The Journal of Machine Learning Research
Print-Archive Edition

Volume 18 (papers from 2017)
Issues 1–73



Microtome Publishing
Brookline, Massachusetts
www.mtome.com

The Journal of Machine Learning Research
Print-Archive Edition
Volume 18 (papers from 2017)
Issues 1–73

The Journal of Machine Learning Research (JMLR) is an open access journal. All articles published in JMLR are freely available via electronic distribution. This Print-Archive Edition is published annually as a means of archiving the contents of the journal in perpetuity. The contents of this edition are articles published electronically in JMLR in 2017.

JMLR is abstracted in ACM Computing Reviews, INSPEC, and Psychological Abstracts/PsycINFO.

JMLR is a publication of Journal of Machine Learning Research, Inc. For further information regarding JMLR, including open access to articles, visit http://www.jmlr.org/.

JMLR Print-Archive Edition is a publication of Microtome Publishing under agreement with Journal of Machine Learning Research, Inc. For further information regarding the Print-Archive Edition, including subscription and distribution information and background on open-access print archiving, visit Microtome Publishing at http://www.mtome.com/.

Collection copyright © 2017 The Journal of Machine Learning Research, Inc. and Microtome Publishing.
Individual articles copyright © 2017 by their respective authors and distributed under a Creative Commons Attribution 4.0 International License.

ISSN 1532-4435 (print)
ISSN 1533-7928 (online)

JMLR Editorial Board

Editor-in-Chief
Bernhard Schölkopf, MPI for Intelligent Systems, Germany

Editor-in-Chief
Kevin Murphy, Google Research, USA

Managing Editor
Aron Culotta, Illinois Institute of Technology, USA

Production Editor
Charles Sutton, University of Edinburgh, UK

JMLR Web Master
Chiyuan Zhang, Massachusetts Institute of Technology, USA

JMLR Action Editors
Ryan Adams, Princeton University, USA. **Shivani Agarwal**, University of Pennsylvania, USA. **Edoardo M. Airoldi**, Harvard University, USA. **Anima Anandkumar**, California Institute of Technology, USA. **Peter Auer**, University of Leoben, Austria. **David Barber**, University College London, UK. **Samy Bengio**, Google Research, USA. **Yoshua Bengio**, Université de Montréal, Canada. **Jeff Bilmes**, University of Washington, USA. **Karsten Borgwardt**, ETH Zurich, Switzerland. **Léon Bottou**, Facebook AI Research. **François Caron**, University of Oxford, United Kingdom. **Miguel A. Carreira-Perpinan**, University of California, Merced, USA. **Alexander Clark**, King's College London, UK. **Corinna Cortes**, Google Research, USA. **Koby Crammer**, Technion, Israel. **Sanjoy Dasgupta**, University of California, San Diego, USA. **Inderjit S. Dhillon**, University of Texas, Austin, USA. **Jennifer Dy**, Northeastern University, USA. **Gal Elidan**, Hebrew University, Israel. **Charles Elkan**, University of California at San Diego, USA. **Barbara Engelhardt**, Princeton University, USA. **Rob Fergus**, New York University, USA. **Kenji Fukumizu**, The Institute of Statistical Mathematics, Japan **Amir Globerson**, Tel Aviv University, Israel **Moises Goldszmidt**, Microsoft Research, USA. **Russ Greiner**, University of Alberta, Canada. **Arthur Gretton**, University College London, UK. **Maya Gupta**, Google Research, USA. **Isabelle Guyon**, ClopiNet, USA. **Moritz Hardt**, Google Research, USA. **Thomas Hofmann**, ETH Zurich, Switzerland **Bert Huang**, Virginia Tech, USA. **Aapo Hyvärinen**, University of Helsinki, Finland **Alex Ihler**, University of California, Irvine, USA. **Tommi Jaakkola**, Massachusetts Institute of Technology, USA. **Samuel Kaski**, Aalto University, Finland **Sathiya Keerthi**, Microsoft Research, USA. **Emtiyaz Khan**, RIKEN Center for Advanced Intelligence, Japan **George Konidaris**, Duke University, USA. **Andreas Krause**, ETH Zurich, Switzerland **Sanjiv Kumar**, Google Research, USA. **Christoph Lampert**, Institute of Science and Technology, Austria **Daniel Lee**, University of Pennsylvania, USA. **Qiang Liu**, Dartmouth College, USA. **Gábor Lugosi**, Pompeu Fabra University, Spain **Michael Mahoney**, University of California at Berkeley, USA. **Shie Mannor**, Technion, Israel **Jon McAuliffe**, University of California at Berkeley, USA. **Robert E. McCulloch**, University of Chicago, USA. **Chris Meek**, Microsoft Research, USA. **Qiaozhu Mei**, University of Michigan, USA. **Vahab Mirrokni**, Google Research, USA. **Mehryar Mohri**, New York University, USA. **Joris Mooij**, University of Amsterdam, Netherlands **Boaz Nadler**, Weizmann Institute of Science, Israel. **Long Nguyen**, University of Michigan, USA. **Sebastian Nowozin**, Microsoft Research, Cambridge, UK **Una-May O'Reilly**, Massachusetts Institute of Technology, USA. **Manfred**

Opper, Technical University of Berlin, Germany **Laurent Orseau**, Google Deepmind, USA. **Luis Ortiz**, University of Michigan - Dearborn, USA. **Jie Peng**, University of California, Davis, USA. **Jan Peters**, Technische Universitaet Darmstadt, Germany **Avi Pfeffer**, Charles River Analytics, USA. **Joelle Pineau**, McGill University, Canada. **Massimiliano Pontil**, Istituto Italiano di Tecnologia (Italy), University College London (UK) **Luc de Raedt**, Katholieke Universiteit Leuven, Belgium **Alexander Rakhlin**, University of Pennsylvania, USA. **Ben Recht**, University of California, Berkeley, USA. **Lorenzo Rosasco**, Massachusetts Institute of Technology, USA. **Saharon Rosset**, Tel Aviv University, Israel **Ruslan Salakhutdinov**, University of Toronto, Canada. **Sujay Sanghavi**, University of Texas, Austin, USA. **Mark Schmidt**, University of British Columbia, Canada. **Marc Schoenauer**, INRIA Saclay, France **John Shawe-Taylor**, University College London, UK **Xiaotong Shen**, University of Minnesota, USA. **David Sontag**, New York University, USA. **Peter Spirtes**, Carnegie Mellon University, USA. **Nathan Srebro**, Toyota Technical Institute at Chicago, USA. **Ingo Steinwart**, University of Stuttgart, Germany **Amos Storkey**, University of Edinburgh, UK **Csaba Szepesvari**, University of Alberta, Canada. **Olivier Teytaud**, INRIA Saclay, France **Ivan Titov**, University of Amsterdam, Netherlands **Ryan Tibshirani**, Carnegie Mellon University **Ryota Tomioka**, Microsoft Research Cambridge, UK **Koji Tsuda**, National Institute of Advanced Industrial Science and Technology, Japan **Zhuowen Tu**, University of California at San Diego, USA. **Nicolas Vayatis**, ENS Cachan, France **S V N Vishwanathan**, Purdue University, USA. **Manfred Warmuth**, University of California at Santa Cruz, USA. **Kilian Weinberger**, Cornell University, USA. **David Wipf**, Microsoft Research Asia, China **Daniela Witten**, University of Washington. **Stefan Wrobel**, Fraunhofer IAIS and University of Bonn, Germany **Eric Xing**, Carnegie Mellon University, USA. **Tong Zhang**, Baidu Inc, China **Zhihua Zhang**, Peking University, China **Hui Zou**, University of Minnesota, USA.

JMLR MLOSS Editors

Alexandre Gramfort, INRIA, Université Paris-Saclay, France **Antti Honkela**, University of Helsinki, Finland **Balázs Kégl**, CNRS / Université Paris-Saclay, France **Cheng Soon Ong**, Australian National University, Australia

JMLR Editorial Board

Naoki Abe, IBM TJ Watson Research Center, USA **Yasemin Altun**, Google Inc, Switzerland **Jean-Yves Audibert**, CERTIS, France **Jonathan Baxter**, Australia National University, Australia **Richard K. Belew**, University of California at San Diego, USA **Kristin Bennett**, Rensselaer Polytechnic Institute, USA **Christopher M. Bishop**, Microsoft Research, Cambridge, UK **Lashon Booker**, The Mitre Corporation, USA **Henrik Boström**, Stockholm University/KTH, Sweden **Craig Boutilier**, Google Research, USA **John Patrick Cunningham**, Columbia University, USA **Nello Cristianini**, University of Bristol, UK **Peter Dayan**, University College, London, UK **Dennis DeCoste**, eBay Research, USA **Thomas Dietterich**, Oregon State University, USA **Saso Dzeroski**, Jozef Stefan Institute, Slovenia **Ran El-Yaniv**, Technion, Israel **Peter Flach**, Bristol University, UK **Dan Geiger**, Technion, Israel **Claudio Gentile**, Università degli Studi dell'Insubria, Italy **Sally Goldman**, Google Research, USA **Thore Graepel**, Google DeepMind and University College London, UK **Tom Griffiths**, University of California at Berkeley, USA **Carlos Guestrin**, University of Washington, USA **Stefan Harmeling**, University of Düsseldorf, Germany **David Heckerman**, Microsoft Research, USA **Katherine Heller**, Duke University, USA **Philipp Hennig**, MPI for Intelligent Systems, Germany **Larry Hunter**, University of Colorado, USA **Jens Kober**, Delft University of Technology, Netherlands **Risi Kondor**, University of Chicago, USA **Aryeh Kontorovich**, Ben-Gurion University of the Negev, Israel **Samory Kpotufe**,

Princeton University, USA **John Lafferty**, University of Chicago, USA **Erik Learned-Miller**, University of Massachusetts, Amherst, USA **Fei Fei Li**, Stanford University, USA **Yi Lin**, University of Wisconsin, USA **Wei-Yin Loh**, University of Wisconsin, USA **Richard Maclin**, University of Minnesota, USA **Sridhar Mahadevan**, University of Massachusetts, Amherst, USA **Vikash Mansingkha**, Massachusetts Institute of Technology, USA **Yishay Mansour**, Tel-Aviv University, Israel **Jon McAuliffe**, University of California, Berkeley, USA **Andrew McCallum**, University of Massachusetts, Amherst, USA **Raymond J. Mooney**, University of Texas, Austin, USA **Klaus-Robert Muller**, Technical University of Berlin, Germany **Kevin Murphy**, Google, USA **Guillaume Obozinski**, Ecole des Ponts - ParisTech, France **Pascal Poupart**, University of Waterloo, Canada **Konrad Rieck**, University of Göttingen, Germany **Cynthia Rudin**, Massachusetts Institute of Technology, USA **Suchi Saria**, Johns Hopkins University, USA **Robert Schapire**, Princeton University, USA **Fei Sha**, University of Southern California, USA **Shai Shalev-Shwartz**, Hebrew University of Jerusalem, Israel **Padhraic Smyth**, University of California, Irvine, USA **Bharath Sriperumbudur**, Pennsylvania State University, USA **Alexander Statnikov**, New York University, USA **Jean-Philippe Vert**, Mines ParisTech, France **Martin J. Wainwright**, University of California at Berkeley, USA **Chris Watkins**, Royal Holloway, University of London, UK **Max Welling**, University of Amsterdam, Netherlands **Chris Williams**, University of Edinburgh, UK **Alice Zheng**, GraphLab, USA

JMLR Advisory Board

Shun-Ichi Amari, RIKEN Brain Science Institute, Japan **Andrew Barto**, University of Massachusetts at Amherst, USA **Thomas Dietterich**, Oregon State University, USA **Jerome Friedman**, Stanford University, USA **Stuart Geman**, Brown University, USA **Geoffrey Hinton**, University of Toronto, Canada **Michael Jordan**, University of California at Berkeley at USA **Leslie Pack Kaelbling**, Massachusetts Institute of Technology, USA **Michael Kearns**, University of Pennsylvania, USA **Steven Minton**, InferLink, USA **Tom Mitchell**, Carnegie Mellon University, USA **Stephen Muggleton**, Imperial College London, UK **Kevin Murphy**, Google, USA **Nils Nilsson**, Stanford University, USA **Tomaso Poggio**, Massachusetts Institute of Technology, USA **Ross Quinlan**, Rulequest Research Pty Ltd, Australia **Stuart Russell**, University of California at Berkeley, USA **Lawrence Saul**, University of California at San Diego, USA **Terrence Sejnowski**, Salk Institute for Biological Studies, USA **Richard Sutton**, University of Alberta, Canada **Leslie Valiant**, Harvard University, USA

Journal of Machine Learning Research

Volume 18 (papers from 2017)
Volume year 2018

Part A

18(1):1–29 **Averaged Collapsed Variational Bayes Inference**
Katsuhiko Ishiguro, Issei Sato, Naonori Ueda

18(2):1–45 **Scalable Influence Maximization for Multiple Products in Continuous-Time Diffusion Networks**
Nan Du, Yingyu Liang, Maria-Florina Balcan, Manuel Gomez-Rodriguez, Hongyuan Zha, Le Song

18(3):1–35 **Local algorithms for interactive clustering**
Pranjal Awasthi, Maria Florina Balcan, Konstantin Voevodski

18(4):1–5 **SnapVX: A Network-Based Convex Optimization Solver**
David Hallac, Christopher Wong, Steven Diamond, Abhijit Sharang, Rok Sosič, Stephen Boyd, Jure Leskovec

18(5):1–30 **Communication-efficient Sparse Regression**
Jason D. Lee, Qiang Liu, Yuekai Sun, Jonathan E. Taylor

18(6):1–36 **Improving Variational Methods via Pairwise Linear Response Identities**
Jack Raymond, Federico Ricci-Tersenghi

18(7):1–37 **Distributed Sequence Memory of Multidimensional Inputs in Recurrent Networks**
Adam S. Charles, Dong Yin, Christopher J. Rozell

18(8):1–35 **Persistence Images: A Stable Vector Representation of Persistent Homology**
Henry Adams, Tegan Emerson, Michael Kirby, Rachel Neville, Chris Peterson, Patrick Shipman, Sofya Chepushtanova, Eric Hanson, Francis Motta, Lori Ziegelmeier

18(9):1–57 **Spectral Clustering Based on Local PCA**
Ery Arias-Castro, Gilad Lerman, Teng Zhang

18(10):1–33 **On Perturbed Proximal Gradient Algorithms**
Yves F. Atchadé, Gersende Fort, Eric Moulines

18(11):1–39 **Differential Privacy for Bayesian Inference through Posterior Sampling**
Christos Dimitrakakis, Blaine Nelson, Zuhe Zhang, Aikaterini Mitrokotsa, Benjamin I. P. Rubinstein

18(12):1–5 **Refinery: An Open Source Topic Modeling Web Platform**
Daeil Kim, Benjamin F. Swanson, Michael C. Hughes, Erik B. Sudderth

18(13):1–43 **Using Conceptors to Manage Neural Long-Term Memories for Temporal Patterns**
Herbert Jaeger

18(14):1–45 **Automatic Differentiation Variational Inference**
Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, David M. Blei

18(15):1–35 **Empirical Evaluation of Resampling Procedures for Optimising SVM Hyperparameters**
Jacques Wainer, Gavin Cawley

18(16):1–49 **A Unified Formulation and Fast Accelerated Proximal Gradient Method for Classification**
Naoki Ito, Akiko Takeda, Kim-Chuan Toh

18(17):1–5 **Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning**
Guillaume Lemaître, Fernando Nogueira, Christos K. Aridas

18(18):1–65 **Information-Geometric Optimization Algorithms: A Unifying Picture via Invariance Principles**
Yann Ollivier, Ludovic Arnold, Anne Auger, Nikolaus Hansen

18(19):1–53 **Breaking the Curse of Dimensionality with Convex Neural Networks**
Francis Bach

18(20):1–32 **Memory Efficient Kernel Approximation**
Si Si, Cho-Jui Hsieh, Inderjit S. Dhillon

18(21):1–38 **On the Equivalence between Kernel Quadrature Rules and Random Feature Expansions**
Francis Bach

18(22):1–40 **Analyzing Tensor Power Method Dynamics in Overcomplete Regime**
Animashree Anandkumar, Rong Ge, Majid Janzamin

18(23):1–5 **JSAT: Java Statistical Analysis Tool, a Library for Machine Learning**
Edward Raff

18(24):1–29 **Identifying a Minimal Class of Models for High–dimensional Data**
Daniel Nevo, Ya'acov Ritov

18(25):1–5 **Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA**
Lars Kotthoff, Chris Thornton, Holger H. Hoos, Frank Hutter, Kevin Leyton-Brown

18(26):1–5 **POMDPs.jl: A Framework for Sequential Decision Making under Uncertainty**
Maxim Egorov, Zachary N. Sunberg, Edward Balaban, Tim A. Wheeler, Jayesh K. Gupta, Mykel J. Kochenderfer

18(27):1–32 **Generalized Pólya Urn for Time-Varying Pitman-Yor Processes**
François Caron, Willie Neiswanger, Frank Wood, Arnaud Doucet, Manuel Davy

18(28):1–39 **Particle Gibbs Split-Merge Sampling for Bayesian Inference in Mixture Models**
Alexandre Bouchard-Côté, Arnaud Doucet, Andrew Roth

18(29):1–53 **Certifiably Optimal Low Rank Factor Analysis**
Dimitris Bertsimas, Martin S. Copenhaver, Rahul Mazumder

18(30):1–52 **Group Sparse Optimization via lp,q Regularization**
Yaohua Hu, Chong Li, Kaiwen Meng, Jing Qin, Xiaoqi Yang

18(31):1–32 **Preference-based Teaching**
Ziyuan Gao, Christoph Ries, Hans U. Simon, Sandra Zilles

18(32):1–40 **Nonparametric Risk Bounds for Time-Series Forecasting**
Daniel J. McDonald, Cosma Rohilla Shalizi, Mark Schervish

18(33):1–39 **Online Bayesian Passive-Aggressive Learning**
Tianlin Shi, Jun Zhu

18(34):1–41 **Asymptotic Analysis of Objectives Based on Fisher Information in Active Learning**
Jamshid Sourati, Murat Akcakaya, Todd K. Leen, Deniz Erdogmus, Jennifer G. Dy

18(35):1–39 **A Spectral Algorithm for Inference in Hidden semi-Markov Models**
Igor Melnyk, Arindam Banerjee

18(36):1–30 **Simplifying Probabilistic Expressions in Causal Inference**
Santtu Tikka, Juha Karvanen

18(37):1–22 **Nearly optimal classification for semimetrics**
Lee-Ad Gottlieb, Aryeh Kontorovich, Pinhas Nisnevitch

18(38):1–39 **Bridging Supervised Learning and Test-Based Co-optimization**
Elena Popovici

18(39):1–5 **GFA: Exploratory Analysis of Multiple Data Sources with Group Factor Analysis**
Eemeli Leppäaho, Muhammad Ammad-ud-din, Samuel Kaski

18(40):1–6 **GPflow: A Gaussian Process Library using TensorFlow**
Alexander G. de G. Matthews, Mark van der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas, Pablo León-Villagrá, Zoubin Ghahramani, James Hensman

18(41):1–49 **COEVOLVE: A Joint Point Process Model for Information Diffusion and Network Evolution**
Mehrdad Farajtabar, Yichen Wang, Manuel Gomez-Rodriguez, Shuang Li, Hongyuan Zha, Le Song

18(42):1–60 **Learning Local Dependence In Ordered Data**
Guo Yu, Jacob Bien

18(43):1–29 **Bayesian Learning of Dynamic Multilayer Networks**
Daniele Durante, Nabanita Mukherjee, Rebecca C. Steorts

18(44):1–29 **Time-Accuracy Tradeoffs in Kernel Prediction: Controlling Prediction Quality**
Samory Kpotufe, Nakul Verma

18(45):1–21 **Asymptotic behavior of Support Vector Machine for spiked population model**
Hanwen Huang

18(46):1–22 **Distributed Semi-supervised Learning with Kernel Ridge Regression**
Xiangyu Chang, Shao-Bo Lin, Ding-Xuan Zhou

18(47):1–43 **On Markov chain Monte Carlo methods for tall data**
Rémi Bardenet, Arnaud Doucet, Chris Holmes

18(48):1–33 **Explaining the Success of AdaBoost and Random Forests as Interpolating Classifiers**
Abraham J. Wyner, Matthew Olson, Justin Bleich, David Mease

18(49):1–47 **Clustering from General Pairwise Observations with Applications to Time-varying Graphs**
Shiau Hong Lim, Yudong Chen, Huan Xu

18(50):1–41 **Uniform Hypergraph Partitioning: Provable Tensor Methods and Sampling Techniques**
Debarghya Ghoshdastidar, Ambedkar Dukkipati

18(51):1–24 **Reconstructing Undirected Graphs from Eigenspaces**
Yohann De Castro, Thibault Espinasse, Paul Rochet

18(52):1–11 **An Optimal Algorithm for Bandit and Zero-Order Convex Optimization with Two-Point Feedback**
Ohad Shamir

18(53):1–31 **Perishability of Data: Dynamic Pricing under Varying-Coefficient Models**
Adel Javanmard

18(54):1–24 **Two New Approaches to Compressed Sensing Exhibiting Both Robust Sparse Recovery and the Grouping Effect**
Mehmet Eren Ahsen, Niharika Challapalli, Mathukumalli Vidyasagar

18(55):1–35 **On the Consistency of Ordinal Regression Methods**
Fabian Pedregosa, Francis Bach, Alexandre Gramfort

18(56):1–26 **Statistical Inference with Unnormalized Discrete Models and Localized Homogeneous Divergences**
Takashi Takenouchi, Takafumi Kanamori

18(57):1–59 **Density Estimation in Infinite Dimensional Exponential Families**
Bharath Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Aapo Hyvärinen, Revant Kumar

18(58):1–52 **Lens Depth Function and k-Relative Neighborhood Graph: Versatile Tools for Ordinal Data Analysis**
Matthäus Kleindessner, Ulrike von Luxburg

18(59):1–39 **Joint Label Inference in Networks**
Deepayan Chakrabarti, Stanislav Funiak, Jonathan Chang, Sofus A. Macskassy

18(60):1–45 **Achieving Optimal Misclassification Proportion in Stochastic Block Models**
Chao Gao, Zongming Ma, Anderson Y. Zhang, Harrison H. Zhou

18(61):1–33 **On the Propagation of Low-Rate Measurement Error to Subgraph Counts in Large Networks**
Prakash Balachandran, Eric D. Kolaczyk, Weston D. Viles

18(62):1–58 **Dense Distributions from Sparse Samples: Improved Gibbs Sampling Parameter Estimators for LDA**
Yannis Papanikolaou, James R. Foulds, Timothy N. Rubin, Grigorios Tsoumakas

18(63):1–29 **Fundamental Conditions for Low-CP-Rank Tensor Completion**
Morteza Ashraphijuo, Xiaodong Wang

18(64):1–34 **Parallel Symmetric Class Expression Learning**
An C. Tran, Jens Dietrich, Hans W. Guesgen, Stephen Marsland

18(65):1–35 **Learning Partial Policies to Speedup MDP Tree Search via Reduction to I.I.D. Learning**
Jervis Pinto, Alan Fern

18(66):1–42 **Hierarchically Compositional Kernels for Scalable Nonparametric Learning**
Jie Chen, Haim Avron, Vikas Sindhwani

18(67):1–29 **Sharp Oracle Inequalities for Square Root Regularization**
Benjamin Stucky, Sara van de Geer

18(68):1–43 **Soft Margin Support Vector Classification as Buffered Probability Minimization**
Matthew Norton, Alexander Mafusalov, Stan Uryasev

18(69):1–29 **Variational Particle Approximations**
Ardavan Saeedi, Tejas D. Kulkarni, Vikash K. Mansinghka, Samuel J. Gershman

18(70):1–37 **A Bayesian Framework for Learning Rule Sets for Interpretable Classification**
Tong Wang, Cynthia Rudin, Finale Doshi-Velez, Yimin Liu, Erica Klampfl, Perry MacNeille

18(71):1–46 **A Robust-Equitable Measure for Feature Ranking and Selection**
A. Adam Ding, Jennifer G. Dy, Yi Li, Yale Chang

18(72):1–32 **Multiscale Strategies for Computing Optimal Transport**
Samuel Gerber, Mauro Maggioni

18(73):1–46 **Non-parametric Policy Search with Limited Information Loss**
Herke van Hoof, Gerhard Neumann, Jan Peters

Part B

18(74):1–40 **Tests of Mutual or Serial Independence of Random Vectors with Applications**
Martin Bilodeau, Aurélien Guetsop Nangue

18(75):1–34 **Recovering PCA and Sparse PCA via Hybrid-(l1,l2) Sparse Sampling of Data Elements**
Abhisek Kundu, Petros Drineas, Malik Magdon-Ismail

18(76):1–61 **Quantifying the Informativeness of Similarity Measurements**
Austin J. Brockmeier, Tingting Mu, Sophia Ananiadou, John Y. Goulermas

18(77):1–36 **Time for a Change: a Tutorial for Comparing Multiple Classifiers Through Bayesian Analysis**
Alessio Benavoli, Giorgio Corani, Janez Demšar, Marco Zaffalon

18(78):1–44 **Relational Reinforcement Learning for Planning with Exogenous Effects**
David Martínez, Guillem Alenyà, Tony Ribeiro, Katsumi Inoue, Carme Torras

18(79):1–31 **Bayesian Tensor Regression**
Rajarshi Guhaniyogi, Shaan Qamar, David B. Dunson

18(80):1–50 **Robust Discriminative Clustering with Sparse Regularizers**
Nicolas Flammarion, Balamurugan Palaniappan, Francis Bach

18(81):1–36 **Making Decision Trees Feasible in Ultrahigh Feature and Label Dimensions**
Weiwei Liu, Ivor W. Tsang

18(82):1–37 **Learning Scalable Deep Kernels with Recurrent Structure**
Maruan Al-Shedivat, Andrew Gordon Wilson, Yunus Saatchi, Zhiting Hu, Eric P. Xing

18(83):1–52 **Convolutional Neural Networks Analyzed via Convolutional Sparse Coding**
Vardan Papyan, Yaniv Romano, Michael Elad

18(84):1–42 **Stochastic Primal-Dual Coordinate Method for Regularized Empirical Risk Minimization**
Yuchen Zhang, Lin Xiao

18(85):1–21 **Angle-based Multicategory Distance-weighted SVM**
Hui Sun, Bruce A. Craig, Lingsong Zhang

18(86):1–47 **Minimax Estimation of Kernel Mean Embeddings**
Ilya Tolstikhin, Bharath K. Sriperumbudur, Krikamol Muandet

18(87):1–28 **The Impact of Random Models on Clustering Similarity**
Alexander J. Gates, Yong-Yeol Ahn

18(88):1–35 **Hierarchical Clustering via Spreading Metrics**
Aurko Roy, Sebastian Pokutta

18(89):1–5 **The MADP Toolbox: An Open Source Library for Planning and Learning in (Multi-)Agent Systems**
Frans A. Oliehoek, Matthijs T. J. Spaan, Bas Terwijn, Philipp Robbel, João V. Messias

18(90):1–50 **A survey of Algorithms and Analysis for Adaptive Online Learning**
H. Brendan McMahan

18(91):1–35 **A distributed block coordinate descent method for training l1 regularized linear classifiers**
Dhruv Mahajan, S. Sathiya Keerthi, S. Sundararajan

18(92):1–31 **Distributed Learning with Regularized Least Squares**
Shao-Bo Lin, Xin Guo, Ding-Xuan Zhou

18(93):1–67 **Identifying Unreliable and Adversarial Workers in Crowdsourced Labeling Tasks**
Srikanth Jagabathula, Lakshminarayanan Subramanian, Ashwin Venkataraman

18(94):1–38 **An Easy-to-hard Learning Paradigm for Multiple Classes and Multiple Labels**
Weiwei Liu, Ivor W. Tsang, Klaus-Robert Müller

18(95):1–32 **Fisher Consistency for Prior Probability Shift**
Dirk Tasche

18(96):1–5 **openXBOW – Introducing the Passau Open-Source Crossmodal Bag-of-Words Toolkit**
Maximilian Schmitt, Björn Schuller

18(97):1–47 **Optimal Rates for Multi-pass Stochastic Gradient Methods**
Junhong Lin, Lorenzo Rosasco

18(98):1–29 **Rank Determination for Low-Rank Data Completion**
Morteza Ashraphijuo, Xiaodong Wang, Vaneet Aggarwal

18(99):1–32 **Bayesian Network Learning via Topological Order**
Young Woong Park, Diego Klabjan

18(100):1–37 **Stability of Controllers for Gaussian Process Dynamics**
Julia Vinogradska, Bastian Bischoff, Duy Nguyen-Tuong, Jan Peters

18(101):1–51 **Harder, Better, Faster, Stronger Convergence Rates for Least-Squares Regression**
Aymeric Dieuleveut, Nicolas Flammarion, Francis Bach

18(102):1–28 **Confidence Sets with Expected Sizes for Multiclass Classification**
Christophe Denis, Mohamed Hebiri

18(103):1–50 **Online Learning to Rank with Top-k Feedback**
Sougata Chaudhuri, Ambuj Tewari

18(104):1–72 **A Unifying Framework for Gaussian Process Pseudo-Point Approximations using Power Expectation Propagation**
Thang D. Bui, Josiah Yan, Richard E. Turner

18(105):1–23 **Accelerating Stochastic Composition Optimization**
Mengdi Wang, Ji Liu, Ethan X. Fang

18(106):1–37 **Distributed Bayesian Learning with Stochastic Natural Gradient Expectation Propagation and the Posterior Server**
Leonard Hasenclever, Stefan Webb, Thibaut Lienart, Sebastian Vollmer, Balaji Lakshminarayanan, Charles Blundell, Yee Whye Teh

18(107):1–28 **Optimal Dictionary for Least Squares Representation**
Mohammed Rayyan Sheriff, Debasish Chatterjee

18(108):1–37 **Computational Limits of A Distributed Algorithm for Smoothing Spline**
Zuofeng Shang, Guang Cheng

18(109):1–67 **Hinge-Loss Markov Random Fields and Probabilistic Soft Logic**
Stephen H. Bach, Matthias Broecheler, Bert Huang, Lise Getoor

18(110):1–49 **Clustering with Hidden Markov Model on Variable Blocks**
Lin Lin, Jia Li

18(111):1–55 **Approximation Vector Machines for Large-scale Online Learning**
Trung Le, Tu Dinh Nguyen, Vu Nguyen, Dinh Phung

18(112):1–29 **Efficient Sampling from Time-Varying Log-Concave Distributions**
Hariharan Narayanan, Alexander Rakhlin

18(113):1–24 **Document Neural Autoregressive Distribution Estimation**
Stanislas Lauly, Yin Zheng, Alexandre Allauzen, Hugo Larochelle

18(114):1–26 **Target Curricula via Selection of Minimum Feature Sets: a Case Study in Boolean Networks**
Shannon Fenn, Pablo Moscato

18(115):1–52 **A General Distributed Dual Coordinate Optimization Framework for Regularized Loss Minimization**
Shun Zheng, Jialei Wang, Fen Xia, Wei Xu, Tong Zhang

18(116):1–40 **Second-Order Stochastic Optimization for Machine Learning in Linear Time**
Naman Agarwal, Brian Bullins, Elad Hazan

18(117):1–49 **Regularized Estimation and Testing for High-Dimensional Multi-Block Vector-Autoregressive Models**
Jiahe Lin, George Michailidis

18(118):1–25 **Learning Theory of Distributed Regression with Bias Corrected Regularization Kernel Network**
Zheng-Chu Guo, Lei Shi, Qiang Wu

18(119):1–59 **Probabilistic Line Searches for Stochastic Optimization**
Maren Mahsereci, Philipp Hennig

18(120):1–49 **Learning Instrumental Variables with Structural and Non-Gaussianity Assumptions**
Ricardo Silva, Shohei Shimizu

18(121):1–34 **Classification of Time Sequences using Graphs of Temporal Constraints**
Mathieu Guillame-Bert, Artur Dubrawski

18(122):1–43 **Distributed Stochastic Variance Reduced Gradient Methods by Sampling Extra Data with Replacement**
Jason D. Lee, Qihang Lin, Tengyu Ma, Tianbao Yang

18(123):1–41 **Kernel Partial Least Squares for Stationary Data**
Marco Singer, Tatyana Krivobokova, Axel Munk

18(124):1–40 **Robust and Scalable Bayes via a Median of Subset Posterior Measures**
Stanislav Minsker, Sanvesh Srivastava, Lizhen Lin, David B. Dunson

18(125):1–53 **Statistical and Computational Guarantees for the Baum-Welch Algorithm**
Fanny Yang, Sivaraman Balakrishnan, Martin J. Wainwright

18(126):1–45 **Online but Accurate Inference for Latent Variable Models with Local Gibbs Sampling**
Christophe Dupuy, Francis Bach

18(127):1–45 **Poisson Random Fields for Dynamic Feature Models**
Valerio Perrone, Paul A. Jenkins, Dario Spanò, Yee Whye Teh

18(128):1–33 **Gap Safe Screening Rules for Sparsity Enforcing Penalties**
Eugene Ndiaye, Olivier Fercoq, Alexandre Gramfort, Joseph Salmon

18(129):1–31 **Minimax Filter: Learning to Preserve Privacy from Inference Attacks**
Jihun Hamm

18(130):1–38 **Knowledge Graph Completion via Complex Tensor Factorization**
Théo Trouillon, Christopher R. Dance, Éric Gaussier, Johannes Welbl, Sebastian Riedel, Guillaume Bouchard

18(131):1–36 **Stabilized Sparse Online Learning for Sparse Data**
Yuting Ma, Tian Zheng

18(132):1–31 **Active-set Methods for Submodular Minimization Problems**
K. S. Sesh Kumar, Francis Bach

18(133):1–33 **A Bayesian Mixed-Effects Model to Learn Trajectories of Changes from Repeated Manifold-Valued Observations**
Jean-Baptiste Schiratti, Stéphanie Allassonnière, Olivier Colliot, Stanley Durrleman

18(134):1–35 **Stochastic Gradient Descent as Approximate Bayesian Inference**
Stephan Mandt, Matthew D. Hoffman, David M. Blei

18(135):1–37 **STORE: Sparse Tensor Response Regression and Neuroimaging Analysis**
Will Wei Sun, Lexin Li

18(136):1–46 **A Survey of Preference-Based Reinforcement Learning Methods**
Christian Wirth, Riad Akrour, Gerhard Neumann, Johannes Fürnkranz

18(137):1–50 **Generalized SURE for optimal shrinkage of singular values in low-rank matrix denoising**
Jérémie Bigot, Charles Deledalle, Delphine Féral

18(138):1–35 **Dimension Estimation Using Random Connection Models**
Paulo Serra, Michel Mandjes

18(139):1–58 **Bayesian Inference for Spatio-temporal Spike-and-Slab Priors**
Michael Riis Andersen, Aki Vehtari, Ole Winther, Lars Kai Hansen

18(140):1–30 **Adaptive Randomized Dimension Reduction on Massive Data**
Gregory Darnell, Stoyan Georgiev, Sayan Mukherjee, Barbara E Engelhardt

18(141):1–35 **A Nonconvex Approach for Phase Retrieval: Reshaped Wirtinger Flow and Incremental Algorithms**
Huishuai Zhang, Yingbin Liang, Yuejie Chi

18(142):1–39 **Consistency, Breakdown Robustness, and Algorithms for Robust Improper Maximum Likelihood Clustering**
Pietro Coretto, Christian Hennig

18(143):1–41 **On Computationally Tractable Selection of Experiments in Measurement-Constrained Regression Models**
Yining Wang, Adams Wei Yu, Aarti Singh

18(144):1–46 **Generalized Conditional Gradient for Sparse Estimation**
Yaoliang Yu, Xinhua Zhang, Dale Schuurmans

18(145):1–31 **Following the Leader and Fast Rates in Online Linear Prediction: Curved Constraint Sets and Other Regularities**
Ruitong Huang, Tor Lattimore, András György, Csaba Szepesvári

18(146):1–48 **Regularization and the small-ball method II: complexity dependent error rates**
Guillaume Lecué, Shahar Mendelson

18(147):1–25 **Matrix Completion with Noisy Entries and Outliers**
Raymond K. W. Wong, Thomas C. M. Lee

18(148):1–29 **Faithfulness of Probability Distributions and Graphs**
Kayvan Sadeghi

18(149):1–49 **Community Extraction in Multilayer Networks with Heterogeneous Community Structure**
James D. Wilson, John Palowitch, Shankar Bhamidi, Andrew B. Nobel

Journal of Machine Learning Research 18 (2017) 1-29 Submitted 6/14; Revised 10/16; Published 1/17

Averaged Collapsed Variational Bayes Inference

Katsuhiko Ishiguro K.ISHIGURO.JP@IEEE.ORG
NTT Communication Science Laboratories
NTT Corporation
Kyoto 619-0237, Japan

Issei Sato SATO@K.U-TOKYO.AC.JP
Graduate School of Frontier Sciences
The University of Tokyo
Tokyo 113-0033, Japan

Naonori Ueda UEDA.NAONORI@LAB.NTT.CO.JP
NTT Communication Science Laboratories
NTT Corporation
Kyoto 619-0237, Japan

Editor: David Blei

Abstract

This paper presents the Averaged CVB (ACVB) inference and offers convergence-guaranteed and practically useful fast Collapsed Variational Bayes (CVB) inferences. CVB inferences yield more precise inferences of Bayesian probabilistic models than Variational Bayes (VB) inferences. However, their convergence aspect is fairly unknown and has not been scrutinized. To make CVB more useful, we study their convergence behaviors in a empirical and practical approach. We develop a convergence-guaranteed algorithm for any CVB-based inference called ACVB, which enables automatic convergence detection and frees non-expert practitioners from the difficult and costly manual monitoring of inference processes. In experiments, ACVB inferences are comparable to or better than those of existing inference methods and deterministic, fast, and provide easier convergence detection. These features are especially convenient for practitioners who want precise Bayesian inference with assured convergence.

Keywords: nonparametric Bayes, collapsed variational Bayes inference, averaged CVB

1. Introduction

Bayesian probabilistic models are powerful because they are capable of expressing complex structures underlying data using various latent variables by formulating the inherent uncertainty of the data generation and collection process as stochastic perturbations. To fully utilize such Bayesian probabilistic models, we rely on Bayesian inferences that compute the posterior distributions of the model given the data. Bayesian inferences infer the shapes of the posterior distribution, in contrast to the point estimate inferences such as Maximum Likelihood (ML) inferences and Maximum a Posterior (MAP) inference that approximate a complicated parameter distribution by a single parameter (set).

Two Bayesian inference algorithms are frequently used for Bayesian probabilistic models: the Gibbs sampler and variational Bayes (cf. Bishop, 2006; Murphy, 2012). The former guarantees asymptotic convergence to the true posteriors of random variables given infinitely many stochastic samples. Variational Bayes (VB) solutions (cf. Attias, 2000; Blei et al., 2016) often enjoy faster convergence with deterministic iterative computations and massively parallel computation thanks to the factorization. The VB approaches also allow easy and automatic detection of convergence. However, VB yields only local optimal solutions due to its use of approximated posteriors.

We can improve these inference methods by developing collapsed estimators, which integrate some parameters out from inferences. Collapsed Gibbs samplers are one of the best inference solutions since they achieve faster convergence and better estimation than the original Gibbs samplers. Recently, collapsed variational Bayes (CVB) solutions have been intensively studied, especially for topic models such as latent Dirichlet allocation (LDA) (Teh et al., 2007; Asuncion et al., 2009; Sato and Nakagawa, 2012) and HDP-LDA (Sato et al., 2012). The seminal paper by Teh and others examined a 2nd-order Taylor approximation of variational expectation (Teh et al., 2007). A simpler 0th-order approximated CVB (CVB0) solution has also been developed as an optimal solution in the sense of minimized α-divergence (Sato and Nakagawa, 2012). These papers report that CVB and CVB0 yield better inference results than VB solutions and even slightly better than exact collapsed Gibbs in data modeling (Kurihara et al., 2007; Teh et al., 2007; Asuncion et al., 2009), link prediction, and neighborhood search (Sato et al., 2012).

In this paper, we are interested in the convergence issue of CVB inferences. The convergence behavior of CVB inferences remains difficult to analyze theoretically, but basically there is no guarantee of convergence for general CVB inferences. Interestingly, this problem has not discussed in the literature with one exception (Foulds et al., 2013), where the authors studied the convergence of CVB on LDA. Unfortunately, their proposal is an online stochastic approximation of MAP which is only valid for LDA. The convergence issue of CVB inference is, however, a more general and problematic issue for practitioners who are unfamiliar with, but still want to tackle state-of-the-art machine learning techniques to various models, not limited to LDA. Since there is no theoretically sound way of determining and detecting convergence of CVB inferences, users must manually determine the convergence of the CVB inferences: a daunting task for non-experts. In that sense, CVB is less attractive than naive VB and EM algorithms, whose convergences are guaranteed and easy to detect automatically. These reasons motivate us to study the convergence behaviors of CVB inferences. Even though the problem remains difficult in theory, we take an empirical and a practical approach to it. We first monitor the naive variational lower bound and the pseudo leave-one-out (LOO) training log likelihood, and then empirically show that the latter may serve as convergence metrics. Next, we develop a simple and effective technique that assures CVB convergence for general Bayesian probabilistic models. Our proposed annealing technique, called **Averaged CVB (ACVB)**, guarantees CVB convergence and allows automatic convergence detection. ACVB has two advantages. First, ACVB posterior updates offer assured convergence due to a simple annealing mechanism. Second, fixed points of the CVB algorithm are equivalent to the converged solution of ACVB, if the original CVB algorithm has fixed points. Our formulation is applicable to any model and is equally valid for CVB as well as CVB0. A convergence-guaranteed ACVB will be the preferred choice for practitioners who want to apply state-of-the-art inference to their problems. In Table 1, we summarize the existing CVB works and this paper, based on applied models and the convergence issue.

We validate our proposed idea on two popular Bayesian probabilistic models. As a simpler model family, we choose LDA (Blei et al., 2003), which is a finite mixture model for a typical Bag-of-Words document data where an observation is governed by a single latent variable. As a more complex model family, we choose the Infinite Relational Model (IRM, Kemp et al., 2006), which

©2017 Ishiguro, Sato and Ueda.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/14-249.html.

Paper	Applied model	Convergence	Comments
Teh et al. (2007)	LDA	-	The seminal paper
Asuncion et al. (2009)	LDA	-	Introduces CVB0
Sato and Nakagawa (2012)	LDA	-	Optimality analysis by α-divergence
Foulds et al. (2013)	LDA	partially	Stochastic approx. MAP rather than CVB0, only valid for LDA
Kurihara et al. (2007)	DPM	-	First attempt at DPM
Teh et al. (2008)	HDP	-	First attempt at HDP
Sato et al. (2012)	HDP	-	Approx. solution
Bleier (2013)	HDP	-	Stochastic approx.
Wan (2013)	HMM	-	First attempt at HMM
Wang and Blunsom (2013)	PCFG	-	First attempt at PCFG
Konishi et al. (2014)	IRM	-	First attempt at IRM
This paper	LDA, IRM	✓	Convergence assurance for any models

Table 1: CVB-related studies summary: in terms of applied models and convergence

is an infinite mixture model for a relational (network) data where an observation link is governed by two latent variables.

In experiments using several real-world relational datasets, we observe that the Averaged CVB0 (ACVB0) inferences offer good data modeling performances, outperform naive VB inferences in many cases, and often show significantly better performances than the 2nd-order CVB and its averaged version. We also observe that ACVB0 typically converges quickly in terms of CPU time, compared to the 2nd-order CVBs and ACVBs. The ACVB0 achieves competitive results with the 0-th order CVB0 inference, which is known to be one of the best Bayesian inference methods. In addition, the ACVB0 guarantees the convergence of the algorithm while the CVB0 does not. Based on these findings, we conclude that ACVB0 inference is convenient and appealing for practitioners because it shows good modeling performance, assures automatic convergence and has relatively fast computation.

The contributions of this paper are summarized as follows:

1. We empirically study the convergence behaviors of CVB inferences and propose a simple but effective annealing technique called Averaged Collapsed Variational Bayes (ACVB) inference that assures the convergence of CVB inferences for all models.

2. We confirm that CVB0 with the above averaging technique (ACVB0) inference offers competitive modeling performances compared to the CVB0 inference, which is one of the best Bayesian inference solutions. We report that the ACVB0 and CVB0 solutions i) outperform naive VBs in most cases, ii) are often significantly better than the 2nd-order counterparts, and iii) are in general computationally faster than the 2nd-order ACVB and CVB solutions.

The rest of this paper is organized as follows. In the 2nd section, we first introduce Bayesian probabilistic models used in experimental validation. Then we briefly review the variational Bayes and the collapsed variational Bayes inferences. Then we present the convergence issue of the CVB inferences using two sections. In section 3, we first empirically show that we can monitor the convergence of the CVB inference by a handy measurement: the pseudo leave-one-out log-likelihood. In section 4, we propose a simple but effective variant of the CVB, the averaged collapsed variational Bayes (ACVB) inference that ensure the convergence of the inference process. The 5th section is devoted to experimental evaluations, and the final section concludes the paper.

2. Background

2.1 Generative Models

In experimental validations, we chose two types of different Bayesian probabilistic models. In this section we first briefly explain them.

2.1.1 LDA

Latent Dirichlet allocation(LDA) (Blei et al., 2003) is a popular Bayesian probabilistic model for topic modeling of Bag-of-Words (BoW) style document data collections. In this paper, we employ LDA as a representative of a model family where an observed word (sample) is governed by a single latent variable.

Assume the observed BoW data collection consists of D documents, where each $d(\in \{1, 2, \ldots, D\})$-th document has N_d tokens. A token may choose a value (word) from the set of unique words whose cardinality is V. Then a probabilistic generative process of LDA is written as follows:

$$\boldsymbol{\beta}_k \mid \boldsymbol{\beta}_0 \sim \text{Dirichlet}(\boldsymbol{\beta}_0), \tag{1}$$

$$\boldsymbol{\theta}_d \mid \boldsymbol{\alpha} \sim \text{Dirichlet}(\boldsymbol{\alpha}), \tag{2}$$

$$z_{d,i} \mid \boldsymbol{\theta}_d \sim \text{Discrete}(\boldsymbol{\theta}_d), \tag{3}$$

$$x_{d,i} \mid \boldsymbol{\beta}_k, z_{d,i} \sim \text{Discrete}\left(\boldsymbol{\beta}_{z_{d,i}}\right). \tag{4}$$

Topic models including LDA are characterized by the notion of topics. A topic is represented as a V-dimensional vector whose v-th attribute indicates the mixing ratio of a v-th word, $\boldsymbol{\beta}_k$, in Equation (1). We assume the number of topics is given as a hyperparameter, and denote the number of topics by K and thus $k \in \{1, 2, \ldots, K\}$. A document is formulated as a mixture of topics. The mixing proportion of the K topics for $d \in \{1, 2, \ldots, D\}$th document is $\boldsymbol{\theta}_d$, in Equation (2).

$z_{d,i}$ in Equation (3) denotes a topic assignment of the dth document's ith observed word. Because $\boldsymbol{\theta}$ is a K-dimensional vector, $z_{d,i} = k \in \{1, 2, \ldots, K\}$. Throughout our paper, we interchangeably choose the 1-of-K representation of Z, where $z_{d,i} = k$ is equivalently represented by $z_{d,i,k} = 1, z_{d,i,l\neq k} = 0$. We generate the observed i-th observation (token) of the d-th document from a V-dimensional Discrete distribution, as in Equation (4). $x_{d,i} = v$ means the i-th token is the v-th symbol (word) out of V vocabularies.

2.1.2 IRM

As a slightly complex model family, we also employ the Infinite Relational Model (Kemp et al., 2006), which is an infinite mixture model for a relational (network) data where an observation link is governed by two latent variables: the cluster assignments of the from-node and the to-node.

IRM is an application of the Dirichlet Process Mixture (DPM) (Sethuraman, 1994; Ferguson, 1973; Blackwell and MacQueen, 1973) for relational data. First, assume a binary two-place relation on the two sets (domains) of objects, namely, $D_1 \times D_2 \to \{0, 1\}$, where $D_1 = \{1, \ldots, i, \ldots, N_1\}$ and $D_2 = \{1, \ldots, j, \ldots, N_2\}$. IRM divides the set of objects into multiple clusters based on the observed relational data matrix of $\boldsymbol{X} = \{x_{i,j} \in \{0, 1\}\}$. Data entry $x_{i,j} \in \{0, 1\}$ denotes the existence of a relation between a row (the first domain) object $i \in \{1, 2, \ldots, N_1\}$ and a column (the second domain) object $j \in \{1, 2, \ldots, N_2\}$. In an online purchase record case, the first domain corresponds to a user list, and object i denotes specific user i. The second domain corresponds to a product item list, and object j denotes specific item j. Data entry $x_{i,j}$ represents the relation between user i and item j: the purchase record.

For such data, we define an IRM as follows:

$$\theta_{k,l}|a_{k,l}, b_{k,l} \sim \text{Beta}\left(a_{k,l}, b_{k,l}\right), \tag{5}$$

$$z_{1,i}|\alpha_1 \sim \text{CRP}\left(\alpha_1\right), \tag{6}$$

$$z_{2,j}|\alpha_2 \sim \text{CRP}\left(\alpha_2\right), \tag{7}$$

$$x_{i,j}|\boldsymbol{Z}_1, \boldsymbol{Z}_2, \{\theta\} \sim \text{Bernoulli}\left(\theta_{z_{1,i}, z_{2,j}}\right). \tag{8}$$

In Equation (5), $\theta_{k,l}$ is the strength of the relation between cluster k in the first domain and cluster l in the second domain. $z_{1,i}$ in Equation (6) and $z_{2,j}$ in Equation (7) denote cluster assignments in the first and the second domains, respectively. Each domain has its own CRP prior, therefore two domains may have different numbers of clusters. We generate observed relational data $x_{i,j}$ following Equation (8), conditioned by cluster assignments $\boldsymbol{Z}_1 = \{z_{1,i}\}_{i=1}^{N_1}$, $\boldsymbol{Z}_2 = \{z_{2,j}\}_{j=1}^{N_2}$ and strengths θ. A typical example of IRM is shown in Figure 1. The IRM infers the appropriate cluster assignment of objects $\boldsymbol{Z}_1 = \{z_{1,i}\}$ and $\boldsymbol{Z}_2 = \{z_{2,j}\}$, given observation relation matrix $\boldsymbol{X} = \{x_{i,j}\}$. We can interpret the clustering as the permutation of object indices to discover the "block" structure (Figure 1 (b)).

As a special case, we can build an IRM for a binary two-place relation between the same domain objects $D \times D \to \{0, 1\}$. The probabilistic generative model of the *single-domain* IRM is described as follows:

$$\theta_{k,l}|a_{k,l}, b_{k,l} \sim \text{Beta}\left(a_{k,l}, b_{k,l}\right), \tag{9}$$

$$z_i|\alpha \sim \text{CRP}\left(\alpha\right), \tag{10}$$

$$x_{i,j}|\boldsymbol{Z}, \{\theta\} \sim \text{Bernoulli}\left(\theta_{z_i, z_j}\right). \tag{11}$$

The generative model clearly shows the difference of a *multi-domain* IRM (Eqs. (5-8)) and *a single-domain* IRM (Eqs. (9-11)). In the latter, there are only N objects in domain D, and they serve as either from-nodes or to-nodes in the network. Object indices i and j point to the same domain. On the other hand, a multi-domain IRM distinguishes the first domain object i from the second domain object j.

Konishi et al. (2014) first introduced CVB algorithms for the single-domain IRM. However, it is not applicable when the number of from-nodes and to-nodes are different. Further, its use is inappropriate if the relations are directional. Thus, the single-domain model has limited applicability,

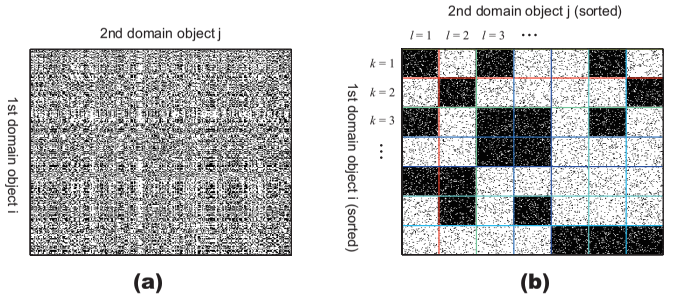


Figure 1: Example of Infinite Relational Models (IRM): (a) input observation $\boldsymbol{X}$. (b) a visualization of inferred clusters $\boldsymbol{Z}$.

unlike the multi-domain IRM. Even though we focus on the multi-domain IRM, all discussions are also valid for a single-domain IRM.

Lastly, we assume two-place relations throughout this paper, but extension that covers higher-order relations is straightforward.

2.2 Variational Bayes (VB) Inference

It is beneficial to quickly derive a VB solution for comparison with a collapsed VB inference. For the VB inference of LDA and IRM, we maximize the VB lower bound, which is defined as:

$$\mathscr{L} = \int q\left(\boldsymbol{Z}, \boldsymbol{\Phi}\right) \log \frac{p\left(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\Phi}\right)}{q\left(\boldsymbol{Z}, \boldsymbol{\phi}\right)} d\boldsymbol{Z} d\boldsymbol{\Phi}, \tag{12}$$

where $\boldsymbol{Z}$ denotes all of the hidden variables (assume $\boldsymbol{Z} = \{\boldsymbol{Z}_1, \boldsymbol{Z}_2\}$ for the IRM case), $\boldsymbol{\Phi}$ denotes all of the associated parameters (e.g., $\boldsymbol{\alpha}_d$ and $\boldsymbol{\beta}_k$ in LDA, $\boldsymbol{\theta}_{k,l}$ in IRM), $\boldsymbol{X}$ denotes all of the observations, and $q(\cdot)$s are *variational* posteriors that approximate the true posteriors. The form of the variational posteriors are chosen to make the inference algorithm efficient. For example, all variational posteriors are assumed to be independent from each other when the mean-field approximation is employed. The lower bound is derived from the following re-formulation of the marginal log likelihood (model evidence) $p\left(\boldsymbol{X}\right)$ by Jensen's inequality (Bishop, 2006; Murphy, 2012):

$$\begin{aligned}
\log p\left(\boldsymbol{X}\right) &= \log \int p\left(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\Phi}\right) d\boldsymbol{Z} d\boldsymbol{\Phi} \\
&= \log \int q\left(\boldsymbol{Z}, \boldsymbol{\Phi}\right) \frac{p\left(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\Phi}\right)}{q\left(\boldsymbol{Z}, \boldsymbol{\Phi}\right)} d\boldsymbol{Z} d\boldsymbol{\Phi} \\
&\geq \int q\left(\boldsymbol{Z}, \boldsymbol{\Phi}\right) \log \frac{p\left(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\Phi}\right)}{q\left(\boldsymbol{Z}, \boldsymbol{\Phi}\right)} d\boldsymbol{Z} d\boldsymbol{\Phi} = \mathscr{L}.
\end{aligned}$$

Thus maximizing the lower bound by identifying good variational posteriors q is reasonable in the sense that it is an approximation of the marginal log likelihood. Maximizing the VB lower bound

is also equivalent to minimizing the Kullback-Leibler divergence between true posteriors p^* and variational posteriors q:

$$\log p(\boldsymbol{X}) = \mathscr{L} - \int q(\mathbf{Z}, \boldsymbol{\Phi}) \log \frac{p(\mathbf{Z}, \boldsymbol{\Phi} \mid \boldsymbol{X})}{q(\mathbf{Z}, \boldsymbol{\Phi})} d\mathbf{Z} d\boldsymbol{\Phi} = \mathscr{L} + \mathrm{KL}(q \mid p^*) .$$

We can readily obtain a general update rule of variational posterior $q(\mathbf{Z})$ and $q(\boldsymbol{\Phi})$ (Bishop, 2006; Murphy, 2012). For example, we have the following update rule for the variational posterior of the ith hidden variable:

$$q(z_i) \propto \exp\left(\mathbb{E}_{q(\mathbf{Z}^{\setminus i}), q(\Phi)}[\log p(\boldsymbol{X}, \mathbf{Z}, \boldsymbol{\Phi})]\right), \tag{13}$$

where $\mathbf{Z}^{\setminus i}$ denotes all of the hidden variables excluding the ith hidden variable. Note that the variational posterior of a hidden variable is dependent on the current values of $q(\Phi)$, i.e., the variational posteriors of all of the associated parameters. The resulting update rules boil down to simple updates of the sufficient statistics of the distributions for hidden variables and parameters for LDA and IRM.

2.3 Collapsed Variational Bayes (CVB) Inference

The general idea of CVB inferences for hierarchical probabilistic models (Kurihara et al., 2007; Teh et al., 2007, 2008; Asuncion et al., 2009; Sato and Nakagawa, 2012; Sato et al., 2012) assumes the variational posteriors of the hidden variables of the model where the *parameters are marginalized out beforehand*. In Equation (12), since parameters $\boldsymbol{\Phi}$ are not marginalized (collapsed) out, we need to compute their variational posteriors as well. The variational posteriors of the parameters impact the inference results and may increase the risk of being trapped at a bad local optimal point.

CVB inference first marginalizes out the parameters in an exact way (as in a collapsed Gibbs sampler). After that, the remaining hidden variables are assumed to be independent from each other. This brings two advantages to CVB. First, the effects of the marginalized parameters are correctly evaluated in CVB while VB approximates them. This means that the variational posteriors computed by CVB will approximate the true posteriors better than those of VB. Second, we can reduce the number of unknown quantities to be inferred because the parameters are already marginalized. This makes the inference faster, more stable, and decreases the risk of being trapped in local optimal solutions. Mathematically, it has been proven that the lower bound of CVB is always tighter than that of the original VB (Teh et al., 2007). This means CVB is always a better approximation of the true posterior than VB.

The following is the formal definition of the CVB lower bound:

$$\mathscr{L}[\mathbf{Z}] = \int q(\mathbf{Z}) \log \frac{p(\boldsymbol{X}, \mathbf{Z})}{q(\mathbf{Z})} d\mathbf{Z}. \tag{14}$$

This is the same formulation as Equation (12) except for the absense of marginalized parameters. Therefore, a general solution is derived in the same manner as in the VB case:

$$q(z_i) \propto \exp\left(\mathbb{E}_{q(\mathbf{Z}^{\setminus i})}[\log p(\boldsymbol{X}, \mathbf{Z})]\right). \tag{15}$$

The CVB inference procedure resembles collapsed Gibbs samplers. We remove one object from the model, recompute the posterior of the object cluster assignment, and return it to the model.

One difference is that CVB computes the soft cluster assignments of $\boldsymbol{Z}$ while the collapsed Gibbs sampler computes hard assignments for each process. We repeat this process on all objects. This one sweep of updates corresponds to one iteration of CVB inference.

One problem is that it is difficult to conduct precise update computations for CVB unlike the original VB, even for such relatively simple Bayesian probabilistic models as LDA and IRM. More specifically, taking expectations over $q(\mathbf{Z}^{\setminus i})$ require intractable discrete combinatorial computations. To remedy this issue, CVB inference approximates these expectations by Taylor expansion. If we denote the expectation of predicate x as $a = \mathbb{E}[x]$, we have:

$$f(x) \approx f(a) + f'(a)(x-a) + \frac{1}{2} f''(a)(x-a)^2 . \tag{16}$$

Taking the expectations of both sides of Equation (16) yields the following equation:

$$\begin{aligned} \mathbb{E}[f(x)] &\approx \mathbb{E}[f(a)] + \mathbb{E}[f'(a)(x-a)] + \frac{1}{2}\mathbb{E}[f''(a)(x-a)^2] \\ &= f(a) + \frac{1}{2}\mathbb{E}[f''(a)(x-a)^2] \\ &= f(\mathbb{E}[x]) + \frac{1}{2} f''(\mathbb{E}[x])\, \mathbb{V}[x] . \end{aligned} \tag{17}$$

The 0th-order term is constant. The 1st-order term is canceled because $x-a$ becomes zero by taking the expectation. $\mathbb{V}$ denotes the posterior variance.

There are two types of approximations in CVB studies. The original **CVB** (Teh et al., 2007) employs 2nd-order Taylor approximation and considers the variance, as in Equation (17). (Asuncion et al., 2009) revealed that the 0th-order Taylor approximation performs quite well in practice for LDA. This is called the **CVB0** solution, which approximates the posterior expectation by

$$\mathbb{E}[f(x)] \approx f(\mathbb{E}[x]). \tag{18}$$

Obviously, the CVB0 solution is simpler than that of the 2nd-order approximation. However it is often superior to the 2nd-order CVB in terms of the perplexity of the learned model (Asuncion et al., 2009; Sato and Nakagawa, 2012; Sato et al., 2012). This may seem counter-intuitive since a 0th-order approximation does not approximate anything. To answer this question, we note that a 0th-order approximation CVB0 is in fact a 1st-order approximation: "CVB1". Recall that in Equation (17) the 1st-order term vanished. This indicates that the 0th-order expansion is equal to the 1st-order expansion. Moreover, it is reasonable that the 1st-order approximation works well in general cases and indeed may outperform higher-order approximations due to uncertainties within the data and imperfections in inference algorithms.

3. Convergence Issue of CVB

3.1 Our Interest: No Assurance for CVB Convergence

It is theoretically guaranteed that each iteration of VB monotonically increases the variational lower bound (Equation (12)) (Attias, 2000; Bishop, 2006). This means VB inference monotonically improves the approximated posterior, and eventually converges to its local optimal solutions. Thus, VB inference yields easy detection of convergence by monitoring the lower bound, and the algorithm automatically halts when it reaches a local optimal point.

Unfortunately, no theoretical guarantee of CVB convergence has been provided so far, probably because we cannot correctly evaluate the posterior expectations over $\boldsymbol{Z}$. What we try to find in CVB solutions is a stationary point of a *Taylor-approximated* CVB lower bound; we are not sure that the procedure actually monotonically improves the *true* lower bound. Moreover, it is unknown whether if the CVB update algorithm has a (algorithmic) fixed point. We are not sure whether the algorithm will even stop after infinitely many iterations. Convergence analysis of CVB inference remains an important open problem in the machine learning field. However, the problem has not been well discussed in the literature though many researchers reported that CVB inference yields better posterior estimations in various cases.

Instead of tackling this problem directly, we study two aspects of CVB convergence in this paper. First we empirically study the convergence behaviors of CVB by monitoring a couple of quantities: a naive VB lower bound and the pseudo leave-one-out log likelihood. We show that the latter is potentially useful for CVB convergence detection. We also propose yet another way to deal with CVB convergence, based on a simple annealing technique. We explain the first aspect in this section and the annealing technique in the next section.

3.2 Assessing Candidate Quantities for CVB Convergence Detection

We cannot correctly compute the true CVB lower bound in Equation (14). Therefore, our first approach is to find some quantities that can serve as proxies of it.

For that purpose, we examine two quantities. The first candidate is an approximation of the naive VB lower bound in Equation (12). The VB lower bound is always a lower bound of the true CVB lower bound. Given the variational posterior of hidden variables $q(\boldsymbol{Z})$, we use this posterior as a proxy of $q(\boldsymbol{Z})$ of the naive VB inference. Computing the variational posteriors of marginalized parameters yields an evaluation of the naive VB lower bound (Equation 12). Of course, this is not equal to the "true" VB lower bound computed by the naive VB inference, but it may be useful for detecting convergence in CVB inferences.

The second one is the pseudo Leave-one-out (LOO) log likelihood of the training data set. The CVB solutions (and the Gibbs sampler) compute the predictive distribution of an object, say, $z_{1,i}$, in a LOO manner. Therefore, we might be able to detect the convergence of CVB inference by watching these predictive distributions. To simplify the explanation, consider a simple model where sample x_i is associated with hidden variable z_i through some probabilistic distributions. A pseudo LOO log likelihood is an approximation of the entire log likelihood (Besag, 1975):

$$\log p\left(\boldsymbol{X} \mid \boldsymbol{Z}\right) \cong \sum_i \log p\left(x_i | \boldsymbol{X}^{\backslash i}, \boldsymbol{Z}\right). \tag{19}$$

We can decompose the above log likelihood as the following simpler sum over samples i:

$$\sum_i \log \sum_k p\left(x_i, z_i = k \mid \boldsymbol{X}^{\backslash i}, \boldsymbol{Z}^{\backslash i}\right). \tag{20}$$

We can easily confirm that $\sum_k p\left(x_i, z_i = k \mid \boldsymbol{X}^{\backslash i}, \boldsymbol{Z}^{\backslash i}\right)$ is in fact a normalization term of a typical collapsed Gibbs posterior by decomposing joint distributions between x_i and z_i. One merit of pseudo LOO log likelihood is its computation cost. It requires no additional cost to compute this quantity during the collapsed Gibbs, since we inevitably compute this normalizer to conduct sampling. Importantly this also holds for the CVB inference; solving Equation (15) typically results in a softened

version of the collapsed Gibbs solution, requiring the same normalizing term. Thus computing the pseudo LOO log likelihood incurs no extra computation cost for CVB inference.

For example, in the case of IRM, we compute the sum of the r.h.s. of Equation (23) (in the appendix) over all the possible values of $z_{1,i} = k \in \{1, 2, \ldots, K\}$ when we update the variational posterior of $q(z_{1,i})$. Let us denote this sum as $C_{1,i}$, which corresponds to $\sum_k p\left(x_i, z_i = k \mid \boldsymbol{X}^{\backslash i}, \boldsymbol{Z}^{\backslash i}\right)$ in Equation (20). In the same manner, we collect $C_{2,j}$ in the second domain updates as the counterpart of C_1. The total log sum $C = \sum_i \log C_{1,i} + \sum_j \log C_{2,j}$ then serves as a "pseudo" log likelihood of the training dataset for CVB.

For LDA and IRM, we synthesize small and large artificial data to test the two quantities. We generated two synthetic Bag-of-Words datasets for LDA. The sizes of these datasets were $D = 500, V = 1000, K = 10$ (smaller **synth 1** data), and $D = 1500, V = 5000, K = 40$ (larger **synth 2** data). Similarly, we generated two synthetic relation datasets for IRM. The sizes and the true numbers of the clusters of these datasets were $N_1 = 100, N_2 = 200, K_1 = 4, K_2 = 5$ (smaller **synth 1** data), and $N_1 = 1,000, N_2 = 1,500, K_1 = 7, K_2 = 6$ (larger **synth 2** data).

Figures 2 and 3 respectively present the evolutions of these two quantities in CVB and CVB0. The hyperparameters are set in the same procedure of the experimental validations (see the Experiment section).

We first notice that the behaviors of the naive VB lower bound (the solid lines) are different in LDA and IRM. This is not desirable behavior as a cue of convergence detection. In the case of LDA (upper panels in the figures), the VB lower bounds first rise to maximum values, and converge at the lower levels. In the case of IRM (lower panels in the figures), on the other hand, the VB lower bound steadily increases and converges at its highest values. It is not surprising that the naive VB lower bound exhibits such strange behaviors. Recall that CVB inference does not directly increase the VB lower bound, which is a looser bound than that of CVB. We also note that the computation load of the lower bound is much heavier than the CVB updates. Thus there is no strong reason to adopt the naive VB lower bound as a convergence monitoring quantity of CVBs.

In contrast, the pseudo LOO training log likelihood (dashed lines) shows similar behaviors in all cases; monotonically increases its value over the iterations (seemingly) and converges at the maximum values. This behavior is preferable as a cue of the convergence than the case of the lower bound. It is also remarkable that pseudo LOO training log likelihood incurs no extra computation loads over the original CVB updates. Since the property of the quantity more or less resembles the model evidence, the pseudo LOO log likelihood is a good choice for convergence detection.

Based on these results, it is preferable to monitor the pseudo LOO log likelihood to asses the convergence of the CVB inferences. However, we stress that there is no theoretical guarantee that the convergence of the pseudo LOO log likelihood is somehow related to the convergence of the CVB inference.

4. Averaged CVB: Convergence Technique for General CVB

In this section, we propose a more direct and convergence-guaranteed technique for general CVB inferences called Averaged CVB (ACVB) and prove that it reaches the fixed point of the original CVB algorithm, if it exists.

Generally it is fruitful to offer an easy convergence detection algorithm for CVB (not restricted to LDA and IRM). For many practitioners, manually determine the convergence of the inference algorithms is difficult. This might be one reason why EM-based algorithms are preferred by prac-

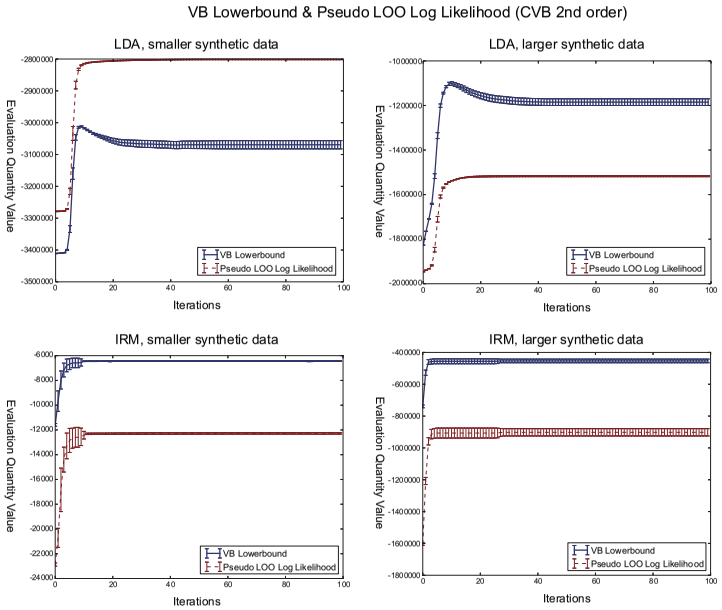


Figure 2: Evolution of two quantities over CVB iterations. Solid lines indicate the evolution of naive VB lower bound. Dashed lines indicate the evolution of pseudo LOO log likelihood on training data. Error bars denote the standard deviations. Upper panels: LDA results on two synthetic data. Lower panels: IRM results on two synthetics data.

titioners: they are convergence guaranteed and the convergence is easy to detect. A convergence-guaranteed ACVB motivates users to use CVB inference, which is more precise than naive VB in theory, with automatic computation termination at a guaranteed convergence.

To the best of our knowledge, a paper by Foulds et al. (Foulds et al., 2013) is the only work that proposes convergence-assured CVB inference. This model, which is based on the Robbins and Monro stochastic approximation (Robbins and Monro, 1951), is only valid for LDA-CVB0. More precisely, the solution presented in the paper is a MAP solution, leveraging the fact that the MAP solution closely resembles the CVB0 solution in the case of LDA. They changed the CVB0 update by ignoring the subtraction of a topic assignment probability vector from sufficient statistics and manually adjusting the Dirichlet parameters, which makes the CVB0 update is equal to the MAP update in LDA. However, this approach is not valid for IRM because the MAP and CVB0 solutions are different. On the contrary, the ACVB is valid for any probabilistic models and indeed for both CVB and CVB0.

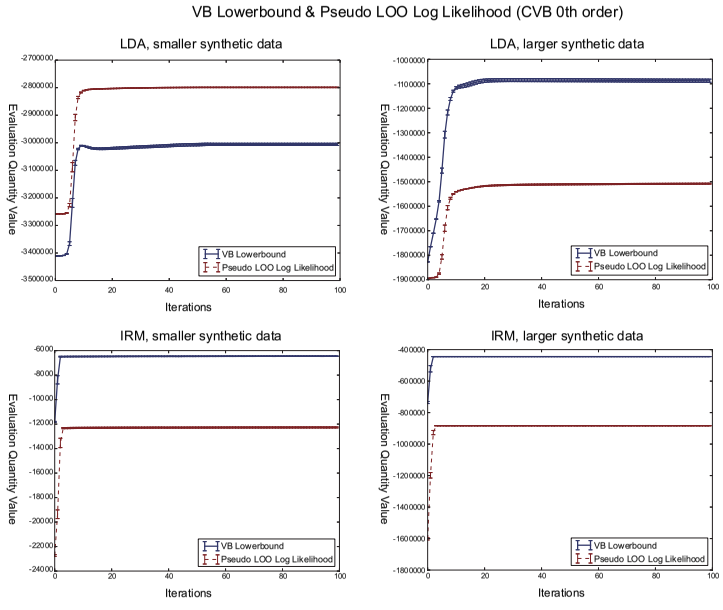


Figure 3: Evolution of two quantities over CVB0 iterations. Solid lines indicate the evolution of naive VB lower bound. Dashed lines indicate the evolution of pseudo LOO log likelihood on training data. Error bars denote the standard deviations. Upper panels: LDA results on two synthetic data. Lower panels: IRM results on two synthetics data.

4.1 Procedure of ACVB

This technique is based on monitoring the changes of $q(\mathbf{Z})$. The rationale is simple: it is reasonable to monitor $q(\mathbf{Z})$ since the CVB solutions are trying to obtain the stationary point of the Taylor-approximated lower bound with respect to $q(\mathbf{Z})$, even though we don't know whether the stationary point exists, as explained before.

Our solution is a simple annealing technique called Averaged CVB (ACVB) which assures the convergence of CVB solutionDfwe s. We emphasize that the ACVB discussion is not limited to LDA and IRM; this technique is applicable to CVB inference on any model. Also, ACVB is equally valid for CVB (2nd order) and CVB0.

After a certain number of iterations for "burn-in", we gradually decrease the portion of the variational posterior changes:

$$\bar{q}^{(s+1)} = \left(1 - \frac{1}{s+1}\right)\bar{q}^{(s)} + \frac{1}{s+1}q^{(s+1)}, \quad \text{or} \quad \bar{q}^{(S)} = \frac{1}{S}\sum_{s=1}^{S} q^{(s)}, \text{“} \tag{21}$$

where s denotes the iterations after completion of the "burn-in" period, $\bar{q}^{(s)}$ denotes the "annealed" variational posterior at the sth iteration, $q^{(s)}$ denotes the variational posterior by CVB inference at the sth iteration, and S is the total number of iterations. After the "burn-in" period, we monitor the ratio of changes of $\bar{q}$ and detect the convergence when the ratio falls below a predefined threshold. As the final result, we use $\bar{q}^{(s)}$, not $q^{(s)}$. During the burn-in period, we monitor the changes of q, which in most cases quickly converges before entering the annealing process.

Hereafter, we respectively denote the (naive) CVB solution and the CVB0 solution, both for ACVB, as **ACVB** and **ACVB0** solutions.

4.2 Properties of ACVB

Concerning the convergence of ACVB, there are three points to note. The first is rather evident but makes ACVB useful for practical CVB inference. ACVB assure convergence, and we can easily detect it by taking the difference of $\bar{q}$ in successive iterations.

Lemma 1 *Averaged variational posterior $\bar{q}^{(s)}$ is convergence-assured: $\forall \epsilon > 0$, $\exists S_0$, s.t. $\forall S > S_0 \Rightarrow \frac{1}{N}\sum_{i=1}^{N}\left|\bar{q}_i^{(S)} - \bar{q}_i^{(S-1)}\right| < \epsilon$.*

Proof Since

$$\frac{1}{S}\sum_{s=1}^{S} q^{(s)} = \left(1 - \frac{1}{S}\right)\frac{1}{S-1}\sum_{s=1}^{S-1} q^{(s)} + \frac{1}{S}q^{(S)},$$

we have

$$\begin{aligned}\left|\frac{1}{S}\sum_{s=1}^{S} q^{(s)} - \frac{1}{S-1}\sum_{s=1}^{S-1} q^{(s)}\right| &= \left|-\frac{1}{S}\frac{1}{S-1}\sum_{s=1}^{S-1} q^{(s)} + \frac{1}{S}q^{(S)}\right| \\ &\leq \frac{1}{S}\frac{1}{S-1}\sum_{s=1}^{S-1}|q^{(s)}| + \frac{1}{S}|q^{(S)}| \\ &\leq \frac{1}{S}\frac{1}{S-1}(S-1) + \frac{1}{S} = \frac{2}{S}.\end{aligned}$$

Thus,

$$\frac{1}{N}\sum_{i=1}^{N}\left|\frac{1}{S}\sum_{s=1}^{S} q_i^{(s)} - \frac{1}{S-1}\sum_{s=1}^{S-1} q_i^{(s)}\right| \leq \frac{2}{S}.$$

If we set $S_0 = \frac{2}{\epsilon}$, then $\forall S > S_0$,

$$\frac{1}{N}\sum_{i=1}^{N}\left|\frac{1}{S}\sum_{s=1}^{S} q_i^{(s)} - \frac{1}{S-1}\sum_{s=1}^{S-1} q_i^{(s)}\right| \leq \frac{2}{S} < \frac{2}{S_0} = \epsilon.$$

This means

$$\frac{1}{N}\sum_{i=1}^{N}\left|\bar{q}_i^{(S)} - \bar{q}_i^{(S-1)}\right| < \epsilon.$$

■

Thus, we can automatically stop the ACVB inference by a stopping rule based on the difference of the ACVB posteriors.

The second point is also noteworthy and validates the use of ACVB in Bayesian inference. We can prove that converged $\bar{q}$ is asymptotically equivalent to the fixed point of the CVB update algorithm, if it exists (note that it is unclear whether the original CVB update algorithm has a fixed point in theory).

Lemma 2 *If variational posterior $q^{(s)}$ converges to a fixed point of the CVB update algorithm, then averaged variational posterior $\bar{q}^{(s)}$ also converges to the fixed point of the CVB algorithm.*

Proof Let q^* be a fixed point of the CVB update algorithm. With this assumption,

$$\lim_{s\to\infty} q^{(s)} = q^* \Leftrightarrow \forall \epsilon > 0, \exists s_0 \text{ s.t. } \forall s > s_0 \Rightarrow |q^{(s)} - q^*| < \epsilon/2.$$

Here, we define

$$\left|\sum_{s=1}^{s_0}(q^{(s)} - q^*)\right| = M > 0,$$

and thus,

$$\lim_{s\to\infty}\frac{M}{s} = 0 \Leftrightarrow \forall \epsilon > 0, \exists s_0' \text{ s.t. } \forall s > s_0' \Rightarrow \frac{M}{s} < \epsilon/2.$$

When $S_0 = \max\{s_0, s_0'\}$, we have

$$\begin{aligned}\forall S > S_0, |\bar{q}^{(S)} - q^*| &= \left|\sum_{s=1}^{S}\frac{1}{S}(q^{(s)} - q^*)\right| \\ &< \frac{M}{S} + \sum_{s=S_0+1}^{S}\left|\frac{1}{S}(q^{(s)} - q^*)\right| \\ &\leq \epsilon/2 + \left|\frac{S - S_0}{S}\right|\epsilon/2 \leq \epsilon/2 + \epsilon/2 = \epsilon.\end{aligned}$$

Therefore,

$$\lim_{s\to\infty}\bar{q}^{(s)} = q^*.$$

■

Note that the literature fails to resolve whether the CVB algorithm (based on Taylor approximation) has an algorithmic fixed point. However, ACVB remains useful because it assures the convergence of the inference process and will find the true solution if CVB has a fixed point. No solutions to the convergence problem of CVB have ever been published, to the best of our knowledge.

As the third point, we note on the required iteration, S, for the convergence of ACVB. The following lemma provides -an insight to set up the maximum number of iterations.

Lemma 3 *The maximum distance between the averaged variational posteriors of consecutive steps is upperbounded by a term proportional to* $\frac{1}{s}$ *where* s *is the number of iterations.*

Proof Consider the averaged variational posterior concerning one hidden variable, z_i. Also consider that $\bar{q}(z_i)$ is a real-valued vector on a simplex (say, on a K dimensional simplex). From the definition, we see:

$$\left\|\bar{q}^{(s+1)}(z_i) - \bar{q}^{(s)}(z_i)\right\| = \frac{1}{s+1}\left\|\bar{q}^{(s)}(z_i) - \bar{q}^{(s+1)}(z_i)\right\| .$$

The maximum L2 distance between two simplex vectors is $\sqrt{2}$. Therefore:

$$\left\|\bar{q}^{(s+1)}(z_i) - \bar{q}^{(s)}(z_i)\right\| = \frac{1}{s+1}\left\|\bar{q}^{(s)}(z_i) - \bar{q}^{(s+1)}(z_i)\right\| \leq \frac{\sqrt{2}}{s+1} .$$

■

Using this lemma, we roughly expect the number of maximum iterations to satisfy a certain threshold of the averaged CVB posterior differences. For example, we have $\left\|\bar{q}^{(s+1)}(z_i) - \bar{q}^{(s)}(z_i)\right\| < 1.4 \times 10^{-3}$ with $s = 1000$ iterations, and $\left\|\bar{q}^{(s+1)}(z_i) - \bar{q}^{(s)}(z_i)\right\| < 1.0 \times 10^{-4}$ with $s = 14000$ iterations. We can use these s as the number of maximum iterations for running the program. In practice we typically need much fewer iterations to achieve the designed threshold.

5. Experiments

This section presents our experimental validations. In summary, we obtained the following results.

1. For almost all the datasets, CVB and ACVB inferences achieved better modeling performance than the naive VB, in both LDA and IRM.
2. CVB0 and ACVB0 inferences often performed significantly better than their 2nd-order counterparts.
3. The computations of the CVB0 and ACVB0 inferences are generally faster than the other VB-based methods.

5.1 Procedure

We compared the performance of the proposed averaged CVB solutions (**ACVB**, **ACVB0**) with naive variational Bayes (**VB**) and CVB solutions (**CVB**, **CVB0**), which are the baseline deterministic inferences. As a reference, we also include comparisons with the collapsed Gibbs samplers (**Gibbs**) with a small number of iterations.

Initialization and hyperparameter choices are important for a fair comparison of inference methods. We employ hyperparameter updates for all solutions: fixed point iterations for VB, CVB, CVB0, ACVB, and ACVB0 and hyper-prior sampling for Gibbs. For LDA, we fixed the initial hyperparameter values based on knowledge from the existing (many) LDA works, especially relying on the result of (Asuncion et al., 2009). For IRM, we tested several initial hyperparameter values and report the results computed using the best hyperparameter setting. All of the hidden variables were initialized in a completely random manner with the uniform distribution to assign soft values

Dataset	D	V	N
20 news group	10,000	13,178	1,046,101
Enron	10,000	15,258	937,113

Table 2: Dataset sizes used in our LDA experiments.

of $p(z_i = k)$. In the case of Gibbs, we performed hard assignments of $z_i = k$ to the most weighted cluster. For the VB, CVB, CVB0, ACVB, and ACVB0 solutions, we normalized the assigned weights. For the LDA experiments, we set the number of topics as $K \in \{50, 100, 200\}$. For the IRM experiments, all of the inferences except Gibbs require a number of truncated clusters a priori. To assess the effect of the truncation level, our experiments examined $K_1 = K_2 = K \in \{20, 40, 60\}$. In practice, we just need to prepare a sufficiently large K to handle data complexity.

We compared the performance of the inference methods by test data perplexity for LDA and by the averaged test data marginal log likelihood for IRM. Given an observation dataset, we excluded roughly 10% of the observations from the inference as held-out test data. After the inference was finished, we computed the perplexity or the marginal log likelihoods of the test data. The test data were randomly sampled for each run. The perplexity and the log likelihood were computed for 20 runs with different initialization and hyperparameter settings.

In the experiments, we set the maximum numbers of inference iterations and iterated the inferences until we reached that maximum. Then we reported the final values of the perplexities (for LDA) or the averaged log likelihood (for IRM) as well the evolutions of these values versus the CPU times.

For the reference Gibbs sampler on LDA, we iterated the sampling procedure $1,000$ times and discarded the first 100 iterations as the burn-in period. For the reference Gibbs sampler on IRM, we iterated the sampling procedure $3,000$ times and discarded the first $1,500$ iterations as the burn-in period.

5.2 Datasets

For our experiments, we prepared several real-world datasets that allowed us to assess the inference performance at several scales and densities.

For the LDA experiments we employed two popular real-world datasets and converted them to the BoW format.

The first dataset is the **20 news group** corpus (Asuncion et al., 2009; Sato and Nakagawa, 2012), including randomly chosen $D = 10,000$ documents with a vocabulary size $V = 13,178$. The second dataset is the **Enron** email corpus (McCallum et al., 2005) including randomly chosen $D = 10,000$ documents with a vocabulary size of $V = 15,258$. Stop words were eliminated. Note the reference materials for detailed information of these datasets. The data sizes and densities are summarized in Table 5.2.

For the IRM experiments, we collected small and large real-world datasets. We explain them more closely since the IRM and the relational data analysis are probably less common than the LDA and the topic model researches for many readers.

The first real-world relational dataset is the Enron e-mail dataset (Klimt and Yang, 2004), which has been used in many studies (Tang et al., 2008; Fu et al., 2009; Ishiguro et al., 2010, 2012). We

Dataset	N_1	N_2	# of "1" entries	Density
Enron Jun.	151	151	257	1.13%
Enron Aug.	151	151	439	1.93%
Enron Oct.	151	151	707	3.10%
Enron Dec.	151	151	377	1.66%
Lastfm UserXUser	1,892	1,892	21,512	0.60%
Lastfm ArtistXTag	6,099	1,088	23,253	0.35%

Table 3: Dataset sizes used in our IRM experiments.

extracted monthly e-mail transactions for 2001. The dataset contains $N = N_1 = N_2 = 151$ company members of Enron. $x_{i,j} = 1(0)$ if there is (not) an e-mail sent from member i to member j. Out of twelve months, we selected the transactions of June (**Enron Jun.**), August (**Enron Aug.**), October (**Enron Oct.**), and December (**Enron Dec.**).

The second real-world relational dataset is the Lastfm dataset[1], which contains several records for the Last.fm music service, including lists of most listened-to musicians, tag assignments for artists, and friend relations among users. We employed the friend relations among $N = N_1 = N_2 = 1892$ users (**Lastfm UserXUser**). $x_{i,j} = 1(0)$ if there is (not) a friend relation from user i to user j. This dataset has ten times more objects and 100 times more matrix entries than the Enron dataset. We also employed the artist-tag relations between $17,632$ artists and $11,946$ tags. Since the relation matrix is too large for Gibbs and naive VB inference, we truncated the number of artists and tags. The original observations were the co-occurrence counts of (artist name, tag) pairs. We binarized the observations as to whether the (artist name, tag) pair counts exceeded; we ignore singletons of (artist name, tag). If the counts exceeded 1, then the observation entries were set to 1; otherwise, they were set to 0. Then all the rows (artists) and columns (tags) that have no "1" entries were removed. The resulting binary matrix consisted of $N_1 = 6,099$ artists and $N_2 = 1,088$ tags (**Lastfm ArtistXTag**). $x_{i,j} = 1(0)$ if artist i is (not) assigned tag word j more than once.

The data sizes and densities are summarized in Table 5.2.

5.3 Results

We start by presenting the numerical performances of the inferences. For LDA, we present the perplexities after 150 iterations in Table 5.3. For IRM, we present the averaged test data log likelihood after 200 iterations in Table 5.3. The results of the best setup are presented for each solution. In addition, we conducted statistical significance tests using t-tests.

First, we review the results of the deterministic (VB-based) inference methods on LDA. CVB0 and ACVB0 inferences always outperformed the naive VB, CVB, and ACVB for LDA[2]. These results are in good accordance with existing CVB researches. The 2nd-order CVB and ACVB inferences often performed worse than the naive VB. (Asuncion et al., 2009) reported that the superiority of CVB inference over naive VB gradually diminishes as the number of topics K increases to roughly more than 100. Our results are probably related to this report.

1. Provided by HetRec2011. http://ir.ii.uam.es/hetrec2011/
2. We note that all of the ACVB0 results on LDA are significantly better than the other VB-based methods, although CVB0 is significantly better than ACVB0 in almost cases.

(A) $K = 50$

Dataset	Gibbs	VB	CVB	CVB0	ACVB	ACVB0
20 news group	1909	2877*	3394*	**2009**	3393*	2016*
Enron	1554	2479*	2923*	**1608**	2922*	1617*

(B): $K = 100$

Dataset	Gibbs	VB	CVB	CVB0	ACVB	ACVB0
20 news group	1718	2937*	3397*	**1791**	3395*	1797*
Enron	1368	2469*	2753*	**1378**	2751*	1387*

(C): $K = 200$

Dataset	Gibbs	VB	CVB	CVB0	ACVB	ACVB0
20 news group	1618	3040*	3100*	**1587**	3097*	1589
Enron	1307	2504*	2384*	**1184**	2382*	1188*

Table 4: Test perplexity on LDA (10% test data) after 150 iterations. Smaller values are better. Boldface indicates the best method within deterministic methods and is significantly better than method(s) marked with $*$ (by t-test, $p = 0.05$). Numbers of Gibbs sampler are computed after $1,000$ iterations. Upper panel: $K = 50$, middle panel: $K = 100$, lower panel: $K = 200$.

CVB0 significantly outperformed ACVB0 in most cases, given the same number of inference iterations. However, ACVB0 provides an easy convergence detection scheme with some theoretical supports, while CVB0 has no such mechanism[3]. In that sense there is no strong reason to choose CVB0 instead of the convergence-guaranteed ACVB0.

Second, we examine the IRM results. For IRM, the CVB and ACVB inferences are significantly better than naive VB in many cases. The ACVB0 inference especially often significantly outperformed the other VB-based methods when K is small. One possible reason is that the averaged posteriors are smoothed versions of the original CVB posteriors. This smoothing may work well for sparse relational data, where inferred solutions tens to be peaky. This result serves as a good reason to adopt ACVB(0) inference: ACVB not only assures the termination of inference algorithm, but also provides better inference solutions than the original CVB in IRM cases.

Third and finally, we compared the performance of the ACVBs and the Collapsed Gibbs sampler. Interestingly, $1,000$ iterations of collapsed Gibbs on LDA performed significantly better than ACVBs, and $3,000$ iterations of the collapsed on IRM did not work as well as expected. We believe this basically depends on the complexity of the model. LDA is a simple probabilistic model. Thus $1,000$ iterations of the collapsed Gibbs are adequate for obtaining good posterior computations. However, IRM is a relatively more complex model than IRM, especially because of the existence of two hidden variables (z_1, z_2) for one observation. This indicates that the (A)CVB inference algorithms on IRM are more likely to be trapped at bad local optima, whereas the collapsed Gibbs sampler yielded stable but not good solutions regardless of the initial Ks. As reported in (Albers et al., 2013), the collapsed Gibbs for IRM requires millions of iterations to obtain better results.

3. although we empirically observed that the pseudo LOO log likelihood may serve as a convergence detector.

(A): $K = 20$

Dataset	Gibbs	VB	CVB	CVB0	ACVB	ACVB0
Enron Jun.	-0.0585	-0.0579	**-0.0559**	-0.0560	-0.0595	-0.0567
Enron Aug.	-0.0830	-0.0786	-0.0762	-0.0777	-0.0809	**-0.0759**
Enron Oct.	-0.1268	-0.1274*	-0.1125	-0.1123	-0.1171	**-0.1110**
Enron Dec.	-0.0740	-0.0726*	-0.0675	-0.0665	-0.0706	**-0.0663**
Lastfm (UserXUser)	-0.0283	-0.0289*	-0.0273*	-0.0271	-0.0271	**-0.0270**
Lastfm (ArtistXTag)	-0.0160	-0.0169*	-0.0163*	-0.0161	-0.0162*	**-0.0160**

(B): $K = 40$

Dataset	Gibbs	VB	CVB	CVB0	ACVB	ACVB0
Enron Jun.	-0.0595	-0.0567	**-0.0549**	-0.0564	-0.0582	-0.0564
Enron Aug.	-0.0838	-0.0808	-0.0770	-0.0753	-0.0819	**-0.0769**
Enron Oct.	-0.1256	-0.1286*	**-0.1129**	-0.1140	-0.1172	-0.1140
Enron Dec.	-0.0750	-0.0722	-0.0694	-0.0682	-0.0731	**-0.0680**
Lastfm (UserXUser)	-0.0280	-0.0292*	-0.0275*	**-0.0269**	-0.0272	**-0.0269**
Lastfm (ArtistXTag)	-0.0161	-0.0172*	-0.0164	-0.0165*	-0.0164	**-0.0163**

(C): $K = 60$

Dataset	Gibbs	VB	CVB	CVB0	ACVB	ACVB0
Enron Jun.	-0.0573	-0.0577	-0.0572	**-0.0558**	-0.0675*	-0.0561
Enron Aug.	-0.0862	-0.0829	-0.0791	-0.0781	-0.0776	**-0.0770**
Enron Oct.	-0.1281	-0.1253*	-0.1122	-0.1144	-0.1162	**-0.1119**
Enron Dec.	-0.0794	-0.0735	-0.0678	-0.0679	-0.0691	**-0.0673**
Lastfm (UserXUser)	-0.0283	-0.0295*	-0.0273*	-0.0269*	**-0.0244**	-0.0268*
Lastfm (ArtistXTag)	-0.0160	-0.0174*	**-0.0164**	**-0.0164**	-0.0166*	-0.0166

Table 5: Marginal test data log likelihood per test data entry (10% test data) on IRM after 200 iterations. Larger values are better. Boldface indicates the best method within deterministic methods and is significantly better than method(s) marked with $*$ (by t-test, $p = 0.05$). Numbers of the Gibbs sampler are computed after 3,000 iterations. Upper panel: $K = 20$, middle panel: $K = 40$, lower panel: $K = 60$.

Thus it is perfectly possible that a collapsed Gibbs outperformed all the VB-based techniques, given more sophisticated sampling techniques and many more iterations.

Our results demonstrate one point of contention. The ACVB and CVB inferences are significantly better than naive VB inference for almost all the IRM cases. This apparently conflicts with a former IRM-CVB study (Konishi et al., 2014) for single-domain IRMs. They reported that CVB inference is inferior to VB for sparse relational data. However, in our experiments, VB never outperformed ACVB or CVB. Our speculation is that this is due to the model structure of the multi-domain IRM (the focus of this paper) and the single-domain IRM in Konishi et al. (2014). In the observation process of a multi-domain IRM (Equation 8), two hidden variables from two independent domains interact (i.e., $z_{1,i}$ and $z_{2,j}$), whereas two from the same domain do not (e.g., $z_{1,i}$ and $z_{1,i'}$). On the

other hand, in the case of a single-domain IRM (Equation 11), two hidden variables from the same (and the sole) domain interact (i.e. z_i and z_j). Given that LDA has no such variable interactions, we believe the multi-domain IRM is one step closer to LDA than the single-domain IRM in terms of interaction complication. Therefore, perhaps the behaviors of the (A)CVB and VB inferences in the multi-domain IRM resembles the LDA cases: CVB is superior to VB.

Figures 4 and 5 present examples of the clustering attained for the **Synth2** and **Lastfm UserXUser** data in $K = 60$. All of the object indices in the figures are sorted so that the objects are grouped into blocks. The horizontal and vertical color lines respectively indicate the borders of the object clusters for first domain k and second domain l. We show the MAP assignments and assign an object to the cluster with the highest posterior probability.

Finally, we show a few plots of performance measures over CPU time. Figure 6 illustrates the time evolution of the test data perplexity of LDA on different data sets. Figure 7 illustrates the time evolution of the test data likelihood of IRM on different data sets. We refrain from presenting the CPU time evolutions of our naive implementations of the collapsed Gibbs samplers, since there is a number of very efficient sampling methods for LDA (Li et al., 2014).

We observe that the ACVB0 and CVB0 inferences generally proceed faster than the other inference methods. Combined with assured convergence and the good numerical performances, we conclude that the ACVB0 solution is a good inference choice for practical usages.

6. Conclusion

In this paper, we proposed an Averaged Collapsed Variational Bayes (ACVB) inference, which is a convergence-guaranteed and useful deterministic inference algorithm that is expected to replace naive VB in practical applications.

We studied the convergence aspect of CVB inference in two ways to address its open problem. We started by examining two possible convergence metric candidates for CVB solutions. Next, we proposed a simple and effective annealing technique, Averaged CVB (ACVB), to assure the convergence of CVB solutions. ACVB posterior updates offer assured convergence due to their simple annealing mechanism. Moreover, the fixed point of the original CVB update algorithm is equivalent to the converged solution of ACVB, if the CVB algorithm has a fixed point (an issue unresolved in the literature). ACVB is applicable to any model and is equally valid for CVB and CVB0.

In experiments using several real-world relational data sets, we concluded that the Averaged CVB0 (ACVB0) inference offers the following impressive performances. It outperforms naive VB inference in many cases and often shows significantly better performances than 2nd-order CVB and its averaged version. ACVB0 typically converges faster than the 2nd-order CVB and ACVB. ACVB0 achieves competitive results with 0-th order CVB0 inference, which is one of the best Bayesian inference methods. In addition, ACVB0 guarantees the convergence of the algorithm while CVB0 does not. Based on these findings we conclude that ACVB0 is an appealing Bayesian inference method for practitioners, who needs precise posterior inferences with guaranteed computational convergences.

For future work, we will enhance the inference speed, especially for relatively slower IRMs. One candidate is to stochastically approximate the sample size as in SGD. Recently, (Foulds et al., 2013) proposed such an approximation for LDA. Another approach parallelizes the inference procedure (Hansen et al., 2011; Albers et al., 2013); they examined the parallelization of collapsed

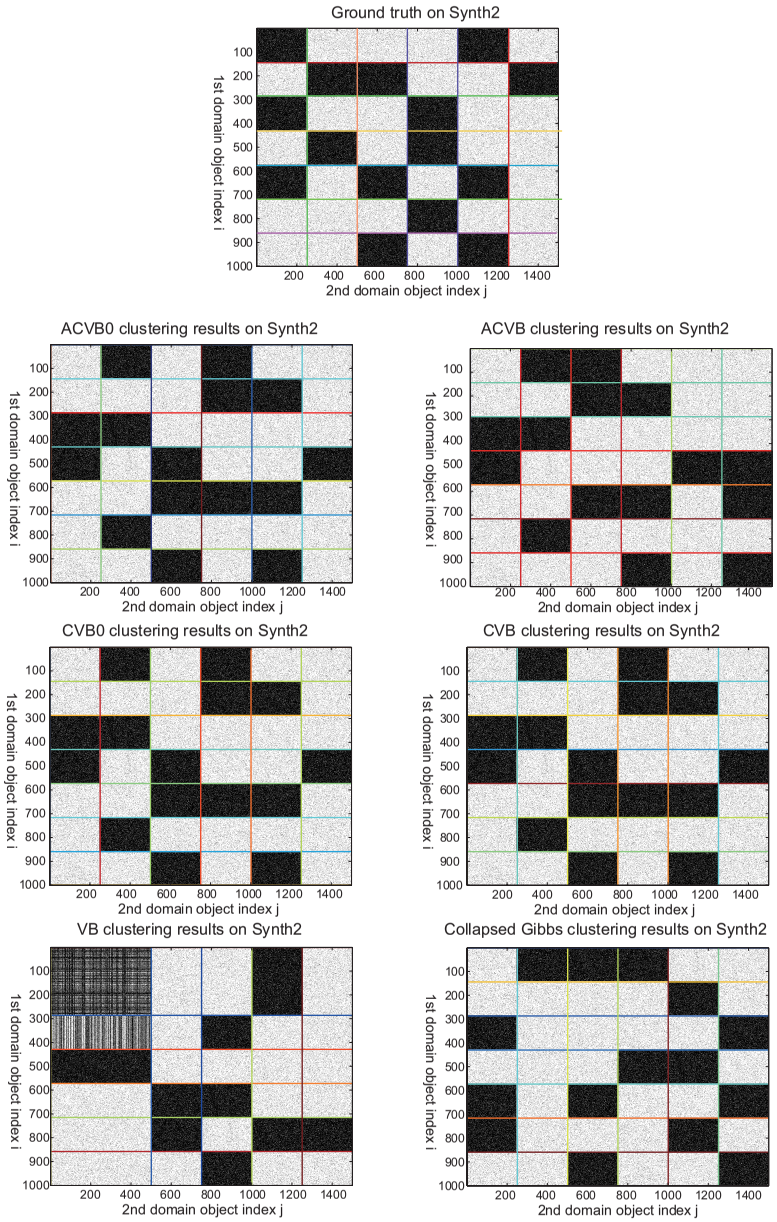


Figure 4: MAP clustering assignments for **Synth2** dataset. All object indices are sorted.

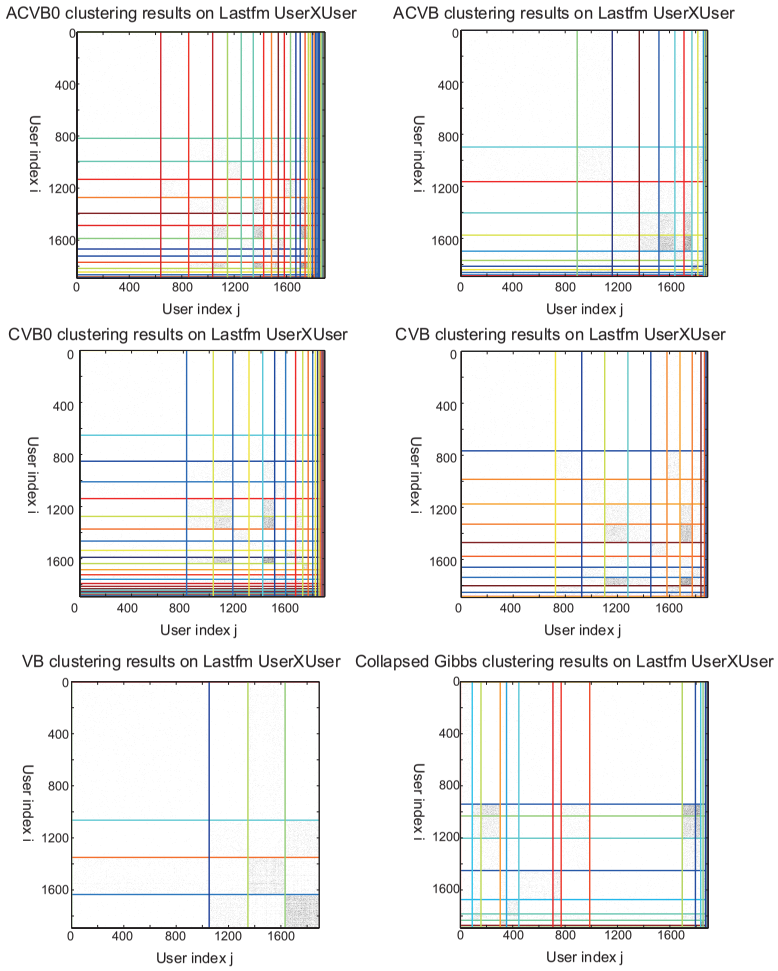


Figure 5: MAP clustering assignments of **Lastfm UserXUser** data set. All object indices are sorted.

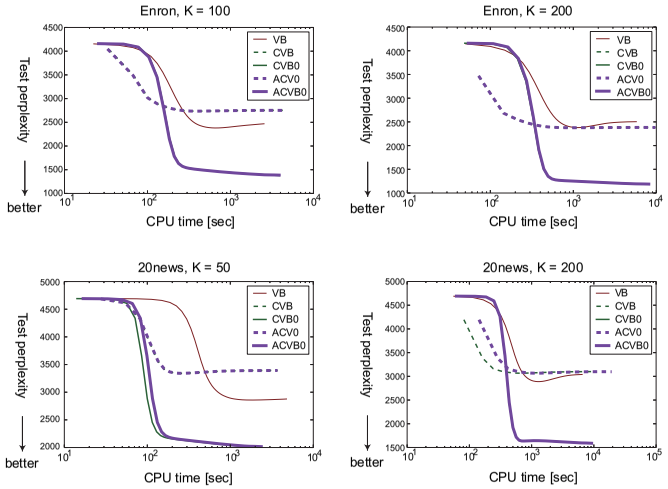


Figure 6: Test data perplexity vs. inference CPU times of LDA for Enron data (upper panels) and news 20 data (lower panels). Horizontal axis denotes CPU times [sec], and vertical axis denotes perplexity.

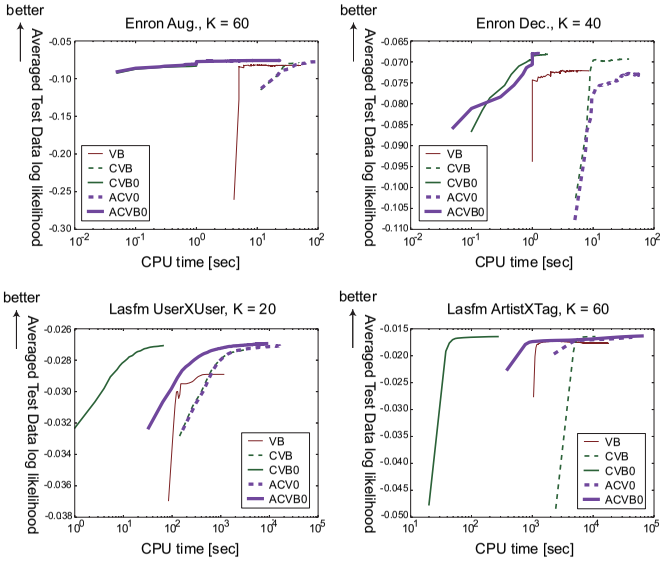


Figure 7: Averaged test data marginal log likelihoods vs. inference CPU times of IRM for Enron data (upper panels) and Lastfm data (lower panels). Horizontal axis denotes CPU times [sec], and vertical axis denotes average test data marginal log likelihoods per relation entry. Presented Gibbs results are those of sampled assignments, not of averaged posteriors.

Gibbs samplers for IRM. It is also important to explore efficient CVB algorithms for more advanced models such as MMSB and its variants (Airoldi et al., 2008; Miller et al., 2009; Griffiths and Ghahramani, 2011). Aside from the representation of multiple cluster assignments, few studies have addressed other issues. For example, (Fu et al., 2009; Ishiguro et al., 2010) focused on the dynamics of network evolution in the context of stochastic blockmodels (MMSB and IRM). Subset IRM (Ishiguro et al., 2012) is another extension of IRM that automatically filters out nodes from the clustering if they are not very informative for grouping. Applying CVB to these models might make it easier for practitioners to examine the depth of various relational data.

Appendix A. CVB inference algorithm for multi-domain IRM

The IRM model in this paper is a multi-domain model where observation $x_{i,j}$ is governed by row latent variable $z_{1,i}$ and column latent variable $z_{2,j}$. Since we distinguish the row and the column objects, there are $N_1 + N_2$ latent variables.

The CVB inference of IRM was first studied in (Konishi et al., 2014). However, their paper focused on a single-domain model, where the observation matrix is a $N \times N$ square matrix and the cluster assignment is only defined on N objects without distinguishing rows and columns. Its CVB scheme is slightly different from the multi-domain IRM, which is our focus in this paper. For readers' convenience, we briefly describe the CVB posteriors for multi-domain IRMs. Since we do not fully explain the actual update rules, readers are referred to (Konishi et al., 2014) to obtain them.

The CVB inference procedure resembles collapsed Gibbs sampling more than ordinary VB inference. We take one object from the model, recompute the posterior of the object cluster assignment, and put it back in the model. One difference is that CVB computes the soft cluster assignments of $\boldsymbol{Z} = \{\boldsymbol{Z}_1, \boldsymbol{Z}_2\}$, while the Gibbs sampler computes hard assignments for each process. We repeat this process on all the objects, and one iteration(sweep) of CVB inference completes.

Next we derive the update rule of the hidden cluster assignment of the first domain $z_{1,i}$. First, we modify the representation of the CVB lower bound. The integral is replaced by the summation because $\boldsymbol{Z}$ is discrete:

$$\begin{aligned}\mathscr{L}\left[z_{1,i}, \boldsymbol{Z}_1^{\setminus(1,i)}, \boldsymbol{Z}_2\right] &= \sum_{z_{1,i}} \sum_{\boldsymbol{Z}_1^{\setminus(1,i)}, \boldsymbol{Z}_2} \left[q(z_{1,i})\, q\left(\boldsymbol{Z}_1^{\setminus(1,i)}\right) q(\boldsymbol{Z}_2) \log \frac{p\left(\boldsymbol{X}^{(1,i)}, \boldsymbol{X}^{\setminus(1,i)}, z_{1,i}, \boldsymbol{Z}_1^{\setminus(1,i)} \boldsymbol{Z}_2\right)}{q(z_{1,i})\, q\left(\boldsymbol{Z}_1^{\setminus(1,i)}\right) q(\boldsymbol{Z}_2)} \right] \\ &= \sum_{z_{1,i}} \mathbb{E}_{\boldsymbol{Z}_1^{\setminus(1,i)}, \boldsymbol{Z}_2} \Big[q(z_{1,i}) \Big\{ \log p\left(\boldsymbol{X}^{(1,i)} | \boldsymbol{X}^{\setminus(1,i)}, z_{1,i}, \boldsymbol{Z}_1^{\setminus(1,i)}, \boldsymbol{Z}_2\right) \\ &\qquad + \log p\left(z_{1,i} | \boldsymbol{Z}_1^{\setminus(1,i)}\right) - \log q(z_{1,i}) \\ &\qquad + (\text{Terms that are not related to } z_{1,i}) \Big\} \Big]. \end{aligned} \tag{22}$$

$\boldsymbol{X}^{(1,i)} = \{x_{i,\cdot}\}$ denotes the set of all the observations concerning object i of the first domain. The remaining observations, the hidden variables excluding $z_{1,i}$, and the statistics computed on these data, are denoted by $\setminus(1,i)$. $\mathbb{E}_x[y]$ indicates the expectations of y on the variational posterior of x.

Taking the derivative of Equation (22) w.r.t. $q(z_{1,i})$ and equating it to zero, we have the following update rule for $q(z_{1,i})$:

$$q(z_{1,i}) \propto \exp\left\{ \mathbb{E}_{\boldsymbol{Z}_1^{\setminus(1,i)}, \boldsymbol{Z}_2}\left[\log p\left(\boldsymbol{X}^{(1,i)} | \boldsymbol{X}^{\setminus(1,i)}, z_{1,i}, \boldsymbol{Z}_1^{\setminus(1,i)}, \boldsymbol{Z}_2\right)\right] + \mathbb{E}_{\boldsymbol{Z}_1^{\setminus(1,i)}, \boldsymbol{Z}_2}\left[\log p\left(z_{1,i} | \boldsymbol{Z}_1^{\setminus(1,i)}\right)\right] \right\}. \tag{23}$$

To evaluate the above equation, we first need to compute the arguments of the expectations. We employ finite truncation of the Stick breaking process (Sethuraman, 1994) as a prior of $\boldsymbol{Z}$. With straightforward computations we have the following (cf. (Konishi et al., 2014)):

$$p\left(z_{1,i} = k | \boldsymbol{Z}_1^{\setminus(1,i)}\right) = \frac{m_{1,k}^{\setminus(1,i)} + 1}{m_{1,k}^{\setminus(1,i)} + M_{1,k}^{\setminus(1,i)} + \alpha_1 + 1} \prod_{k'=1}^{k-1} \frac{M_{1,k'}^{\setminus(1,i)} + \alpha_1}{m_{1,k'}^{\setminus(1,i)} + M_{1,k'}^{\setminus(1,i)} + \alpha_1 + 1}. \tag{24}$$

$$\begin{aligned} &p\left(\boldsymbol{X}^{(1,i)} | z_{1,i} = k, \boldsymbol{Z}_1^{\setminus(1,i)}, \boldsymbol{Z}_2, \boldsymbol{X}^{\setminus(1,i)}\right) \\ &= \prod_{l=1}^{K_2} \frac{\Gamma\left(a_{k,l} + b_{k,l} + n_{k,l}^{\setminus(1,i)} + N_{k,l}^{\setminus(1,i)}\right)}{\Gamma\left(a_{k,l} + n_{k,l}^{\setminus(1,i)}\right)\Gamma\left(b_{k,l} + N_{k,l}^{\setminus(1,i)}\right)} \frac{\Gamma\left(a_{k,l} + n_{k,l}^{\setminus(1,i)} + n_{k,l}^{+(1,i)}\right)\Gamma\left(b_{k,l} + N_{k,l}^{\setminus(1,i)} + N_{k,l}^{+(1,i)}\right)}{\Gamma\left(a_{k,l} + b_{k,l} + n_{k,l}^{\setminus(1,i)} + N_{k,l}^{\setminus(1,i)} + n_{k,l}^{+(1,i)} + N_{k,l}^{+(1,i)}\right)}. \end{aligned} \tag{25}$$

In the above, we introduced the following sufficient statistics:

$$m_{1,k} = \sum_{i=1}^{N_1} \mathbb{I}(z_{1,i} = k) = \sum_{i=1}^{N_1} z_{1,i,k}, \quad m_{1,k}^{\setminus(1,i)} = m_{1,k} - \mathbb{I}(z_{1,i} = k), \tag{26}$$

$$M_{1,k} = \sum_{i=1}^{N_1} \mathbb{I}(z_{1,i} > k) = \sum_{k'=k+1}^{K_1} m_{1,k'}, \quad M_{1,k}^{\setminus(1,i)} = M_{1,k} - \mathbb{I}(z_{1,i} > k), \tag{27}$$

$$n_{k,l} = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} z_{1,i,k} z_{2,j,l} x_{i,j}, \quad n_{k,l}^{\setminus(1,i)} = n_{k,l} - \sum_{j=1}^{N_2} z_{1,i,k} z_{2,j,l} x_{i,j}, \tag{28}$$

$$N_{k,l} = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} z_{1,i,k} z_{2,j,l} (1 - x_{i,j}), \quad N_{k,l}^{\setminus(1,i)} = N_{k,l} - \sum_{j=1}^{N_2} z_{1,i,k} z_{2,j,l} (1 - x_{i,j}) \tag{29}$$

$$n_{k,l}^{+(1,i)} = \sum_{j=1}^{N_2} z_{2,j,l} x_{i,j}, \quad N_{k,l}^{+(1,i)} = \sum_{j=1}^{N_2} z_{2,j,l} (1 - x_{i,j}). \tag{30}$$

The final step is to compute intractable expectations with a help of Taylor-expansion approximation. Since this step is completely identical with (Konishi et al., 2014) thus we refrain from presenting details.

Appendix B. Numerical Performance after Convergence

In the experiment section, we examined the performances of several inference solutions fixing the number of inference iterations. Based on the evolutions in Figure 6, we think the LDA inferences have already almost reached the converged status with 150 iterations. However, the results of IRM (Figure 7) indicate that the inference may take some additional time until convergence. Then it is natural to ask that how the performances of the inferences change when we keep the algorithm iterates until convergence (on IRM).

We need to decide which parameter/statistics to be monitored to detect convergence. For the VB solution, we monitored the VB lower bound. For the CVB and CVB0 solutions, we monitored the pseudo LOO log likelihood of the training data set in Sec. 3. For the ACVB and ACVB0 solutions, we used the annealed posteriors. We determined the solution convergence from the relative changes

(A) $K = 20$

Data Set	Gibbs	VB	CVB	CVB0	ACVB	ACVB0
Enron Jun.	-0.0585	-0.0578	-0.0559	-0.0560	**-0.0550**	-0.0561
Enron Aug.	-0.0830*	-0.0797	-0.0763	-0.0777	-0.0772	**-0.0748**
Enron Oct.	-0.1268*	-0.1275*	-0.1125	-0.1119	-0.1155	**-0.1110**
Enron Dec.	-0.0740*	-0.0727*	-0.0675	-0.0665	-0.0680	**-0.0656**
Lastfm (UserXUser)	-0.0283*	-0.0289*	-0.0273*	-0.0271	-0.0271	**-0.0269**
Lastfm (ArtistXTag)	-0.0160*	-0.0171*	-0.0163*	-0.0161*	-0.0161*	**-0.0158**

(B) $K = 40$

Data Set	Gibbs	VB	CVB	CVB0	ACVB	ACVB0
Enron Jun.	-0.0595	-0.0564	-0.0549	-0.0564	-0.0552	**-0.0534**
Enron Aug.	-0.0838*	-0.0809*	-0.0769	-0.0753	**-0.0736**	-0.0765
Enron Oct.	-0.1256*	-0.1271*	**-0.1128**	-0.1140	-0.1140	-0.1140
Enron Dec.	-0.0750*	-0.0722	-0.0693	-0.0682	-0.0691	**-0.0680**
Lastfm (UserXUser)	-0.0280*	-0.0292*	-0.0273*	**-0.0269**	-0.0272	**-0.0269**
Lastfm (ArtistXTag)	**-0.0161**	-0.0174*	-0.0164*	-0.0165*	-0.0162	**-0.0161**

(C) $K = 60$

Data Set	Gibbs	VB	CVB	CVB0	ACVB	ACVB0
Enron Jun.	-0.0573	-0.0576	-0.0572	-0.0558	-0.0570	**-0.0540**
Enron Aug.	-0.0862*	-0.0826*	-0.0791	-0.0780	**-0.0750**	-0.0759
Enron Oct.	-0.1281*	-0.1275*	-0.1121	-0.1143	-0.1143	**-0.1119**
Enron Dec.	-0.0794*	-0.0736	-0.0676	-0.0679	-0.0678	**-0.0673**
Lastfm (UserXUser)	-0.0283*	-0.0294*	-0.0273*	-0.0269*	**-0.0244**	-0.0268*
Lastfm (ArtistXTag)	**-0.0160**	-0.0177*	-0.0164*	-0.0164*	-0.0163*	-0.0163*

Table 6: Marginal test data log likelihood per test data entry (10% test data) on IRM after convergence. Larger values are better. Boldfaces indicate the best method, which is significantly better than the method(s) marked with $*$ (by t-test, $p = 0.05$). The numbers of the Gibbs sampler is that of after $3,000$ iterations. Upper panel: $K = 20$, middle panel: $K = 40$, lower panel: $K = 60$.

in the monitored quantity; if the changes were smaller than 0.001% of the current value of the quantity, we assumed that the algorithm had converged.

The modeling performances are presented in Table 6. They show the averages of the test data marginal log likelihood after convergence. The results of the best setup are presented for each solution. In addition, we conducted statistical significance tests using t-tests.

Overall, we observe that ACVB and ACVB0 performed slightly better than the experimental results with a fixed number of iterations. This result indicates that the smoothed posteriors of ACVB and ACVB0 are possibly close to good local solutions of the true CVB posteriors.

References

Collapsed variational Bayesian inference for hidden Markov models. In *Proc. AISTATS*, 2013.

Edoardo M. Airoldi, David M. Blei, Stephen E. Fienberg, and Eric P. Xing. Mixed Membership Stochastic Blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2008.

Kristoffer Jon Albers, Andreas Leon Aagard Moth, Morten Mørup, and Mikkel N. Schmidt. Large scale inference in the infinite relational model: Gibbs sampling is not enough. In *Proc. MLSP*, 2013.

Arthur Asuncion, Max Welling, Padhraic Smyth, and Yee Whye Teh. On smoothing and inference for topic models. In *Proc. UAI*, 2009.

Hagai Attias. A variational Bayesian framework for graphical models. In *Proc. NIPS*, 2000.

Jullian Besag. Statistical Analysis of Non-Lattice Data. *Journal of the Royal Statistical Society: Series D*, 24(3):179–195, 1975.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2006.

David Blackwell and James B. MacQueen. Ferguson Distributions via Polya urn schemes. *The Annals of Statistics*, 1(2):353–355, 1973.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *arXiv*, page arXiv:1601.00670v1 [stat.CO], 2016.

Arnim Bleier. Practical collapsed stochastic variational inference for the HDP. In *Proc. NIPS workshop on topic models*, 2013.

Thomas S. Ferguson. A Bayesian Analysis of Some Nonparametric Problems. *The Annals of Statistics*, 1(2):353–355, 1973.

James Foulds, Levi Boyles, Christopher DuBois, Padhraic Symyth, and Max Welling. Stochastic collapsed variational Bayesian inference for latent Dirichlet allocation. In *Proc. KDD*, 2013.

Wenjie Fu, Le Song, and Eric P. Xing. Dynamic mixed membership blockmodel for evolving networks. In *Proc. ICML*, 2009.

Thomas L. Griffiths and Zoubin Ghahramani. The Indian Buffet Process : An Introduction and Review. *Journal of Machine Learning Research*, 12:1185–1224, 2011.

Toke Jansen Hansen, Morten Mørup, and Lars Kai Hanse. Non-parametric co-clustering of large scale sparse nipartite networks on the GPU. In *Proc. MLSP*, 2011.

Katsuhiko Ishiguro, Tomoharu Iwata, Naonori Ueda, and Joshua B. Tenenbaum. Dynamic infinite relational model for time-varying relational data analysis. In *Proc. NIPS*, 2010.

Katushiko Ishiguro, Naonori Ueda, and Hiroshi Sawada. Subset infinite relational models. In *Proc. AISTATS*, 2012.

Charles Kemp, Joshua B. Tenenbaum, Thomas L. Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *Proc. AAAI*, 2006.

Bryan Klimt and Yiming Yang. The Enron corpus : a new dataset for Email classification Research. In *Proc. ECML*, 2004.

Takuya Konishi, Takatomi Kubo, Kazuho Watanabe, and Kazushi Ikeda. Variational Bayesian Inference Algorithms for Infinite Relational Model of Network Data. *IEEE Transactions on Neural Networks and Learning Systems*, 26(9):2176–2181, 2014.

Kenichi Kurihara, Max Welling, and Yee Whye Teh. Collapsed variational Dirichlet process mixture models. In *Proc. IJCAI*, 2007.

Aaron Q. Li, Amr Ahmed, Sujith Ravi, and Alexander J. Smola. Reducing the sampling complexity of topic models. In *Proc. KDD*, 2014.

Andrew McCallum, Andres Corrada-Emmanuel, and Xuerui Wang. Topic and role discovery in social networks. In *Proc. IJCAI*, 2005.

Kurt T. Miller, Thomas L. Griffiths, and Michael I. Jordan. Nonparametric latent feature models for link prediction. In *Proc. NIPS*, 2009.

Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

Issei Sato and Hiroshi Nakagawa. Rethinking collapsed variational Bayes inference for LDA. In *Proc. ICML*, 2012.

Issei Sato, Kenichi Kurihara, and Hiroshi Nakagawa. Practical collapsed variational Bayes inference for hierarchical dirichlet process. In *Proc. KDD*, 2012.

Jayaram Sethuraman. A Constructive definition of Dirichlet process. *Statistica Sinica*, 4:639–650, 1994.

Lei Tang, Huan Liu, Jianping Zhang, and Zohreh Nazeri. Community evolution in dynamic multi-mode networks. In *Proc. KDD*, 2008.

Yee Whye Teh, David Newman, Max Welling, and D. Neaman. A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation. In *Proc. NIPS*, 2007.

Yee Whye Teh, Kenichi Kurihara, and Max Welling. Collapsed variational inference for HDP. In *Proc. NIPS*, 2008.

Pengyu Wang and Phil Blunsom. Collapsed variational Bayesian inference for PCFGs. In *Proc. ACL*, 2013.

Journal of Machine Learning Research 18 (2017) 1-45 Submitted 9/14; Revised 7/16; Published 2/17

Scalable Influence Maximization for Multiple Products in Continuous-Time Diffusion Networks

Nan Du DUNAN@GOOGLE.COM
Google Research, 1600 Amphitheatre Pkwy, Mountain View, CA 94043

Yingyu Liang YINGYUL@CS.PRINCETON.EDU
Department of Computer Science, Princeton University, Princeton, NJ 08540

Maria-Florina Balcan NINAMF@CS.CMU.EDU
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213

Manuel Gomez-Rodriguez MANUELGR@MPI-SWS.ORG
MPI for Software Systems, Kaiserslautern, Germany 67663

Hongyuan Zha ZHA@CC.GATECH.EDU
Le Song LSONG@CC.GATECH.EDU
College of Computing, Georgia Institute of Technology, Atlanta, GA 30332

Editor: Jeff Bilmes

Abstract

A typical viral marketing model identifies influential users in a social network to maximize a single product adoption assuming unlimited user attention, campaign budgets, and time. In reality, multiple products need campaigns, users have limited attention, convincing users incurs costs, and advertisers have limited budgets and expect the adoptions to be maximized soon. Facing these user, monetary, and timing constraints, we formulate the problem as a submodular maximization task in a continuous-time diffusion model under the intersection of one matroid and multiple knapsack constraints. We propose a randomized algorithm estimating the user influence[1] in a network ($|\mathcal{V}|$ nodes, $|\mathcal{E}|$ edges) to an accuracy of ϵ with $n = \mathcal{O}(1/\epsilon^2)$ randomizations and $\tilde{\mathcal{O}}(n|\mathcal{E}| + n|\mathcal{V}|)$ computations. By exploiting the influence estimation algorithm as a subroutine, we develop an adaptive threshold greedy algorithm achieving an approximation factor $k_a/(2 + 2k)$ of the optimal when k_a out of the k knapsack constraints are active. Extensive experiments on networks of millions of nodes demonstrate that the proposed algorithms achieve the state-of-the-art in terms of effectiveness and scalability.

Keywords: Influence Maximization, Influence Estimation, Continuous-time Diffusion Model, Matroid, Knapsack

1. Introduction

Online social networks play an important role in the promotion of new products, the spread of news, the success of political campaigns, and the diffusion of technological innovations. In these contexts, the influence maximization problem (or viral marketing problem) typically has the following flavor: identify a set of influential users in a social network, who, when convinced to adopt a product, shall influence other users in the network and trigger a large cascade of adoptions. This problem has been studied extensively in the literature from both the modeling and the algorithmic aspects (Richardson and Domingos, 2002; Kempe et al., 2003; Leskovec et al., 2007; Chen et al., 2009, 2010a,b, 2011, 2012; Ienco et al., 2010; Goyal et al., 2011a,b; Gomez-Rodriguez and Schölkopf, 2012), where it has been typically assumed that the host (*e.g.*, the owner of an online social platform) faces a single product, endless user attention, unlimited budgets and unbounded time. However, in reality, the host often encounters a much more constrained scenario:

- **Multiple-Item Constraints:** multiple products can spread simultaneously among the same set of social entities. These products may have different characteristics, such as their revenues and speed of spread.
- **Timing Constraints:** the advertisers expect the influence to occur within a certain time window, and different products may have different timing requirements.
- **User Constraints:** users of the social network, each of which can be a potential source, would like to be exposed to only a small number of ads. Furthermore, users may be grouped by their geographical locations, and advertisers may have a target population they want to reach.
- **Product Constraints:** seeking initial adopters entails a cost to the advertiser, who needs to pay to the host and often has a limited amount of money.

For example, Facebook (*i.e.*, the host) needs to allocate ads for various products with different characteristics, *e.g.*, clothes, books, or cosmetics. While some products, such as clothes, aim at influencing within a short time window, some others, such as books, may allow for longer periods. Moreover, Facebook limits the number of ads in each user's sidebar (typically it shows less than five) and, as a consequence, it cannot assign all ads to a few highly influential users. Finally, each advertiser has a limited budget to pay for ads on Facebook and thus each ad can only be displayed to some subset of users. In our work, we incorporate these myriads of practical and important requirements into consideration in the influence maximization problem.

We account for the multi-product and timing constraints by applying product-specific continuous-time diffusion models. Here, we opt for continuous-time diffusion models instead of discrete-time models, which have been mostly used in previous work (Kempe et al., 2003; Chen et al., 2009, 2010a,b, 2011, 2012; Borgs et al., 2012). This is because artificially discretizing the time axis into bins introduces additional errors. One can adjust the additional tuning parameters, like the bin size, to balance the tradeoff between the error and the computational cost, but the parameters are not easy to choose optimally. Extensive experimental comparisons on both synthetic and real-world data have shown that discrete-time models provide less accurate influence estimation than their continuous-time counterparts (Gomez-Rodriguez et al., 2011; Gomez-Rodriguez and Schölkopf, 2012; Gomez-Rodriguez et al., 2013; Du et al., 2013a,b).

However, maximizing influence based on continuous-time diffusion models also entails additional challenges. First, evaluating the objective function of the influence maximization

1. Partial results in the paper on influence estimation have been published in a conference paper: Nan Du, Le Song, Manuel Gomez-Rodriguez, and Hongyuan Zha. Scalable influence estimation in continuous time diffusion networks. In Advances in Neural Information Processing Systems 26, 2013.

©2017 Nan Du, Yingyu Liang, Maria-Florina Balcan, Manuel Gomez-Rodriguez, Hongyuan Zha and Le Song.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/14-400.html.

problem (*i.e.*, the influence estimation problem) in this setting is a difficult graphical model inference problem, *i.e.*, computing the marginal density of continuous variables in loopy graphical models. The exact answer can be computed only for very special cases. For example, Gomez-Rodriguez and Schölkopf (2012) have shown that the problem can be solved exactly when the transmission functions are exponential densities, by using continuous time Markov processes theory. However, the computational complexity of such approach, in general, scales exponentially with the size and density of the network. Moreover, extending the approach to deal with arbitrary transmission functions would require additional nontrivial approximations which would increase even more the computational complexity. Second, it is unclear how to scale up influence estimation and maximization algorithms based on continuous-time diffusion models to millions of nodes. Especially in the maximization case, the influence estimation procedure needs to be called many times for different subsets of selected nodes. *Thus, our first goal is to design a scalable algorithm which can perform influence estimation in the regime of networks with millions of nodes.*

We account for the user and product constraints by restricting the feasible domain over which the maximization is performed. We first show that the overall influence function of multiple products is a submodular function and then realize that the user and product constraints correspond to constraints over the ground set of this submodular function. To the best of our knowledge, previous work has not considered both user and product constraints simultaneously over general unknown different diffusion networks with non-uniform costs. In particular, (Datta et al., 2010) first tried to model both the product and user constraints only with uniform costs and infinite time window, which essentially reduces to a special case of our formulations. Similarly, (Lu et al., 2013) considered the allocation problem of multiple products which may have competitions within the infinite time window. Besides, they all assume that multiple products spread within the same network. In contrast, our formulations generally allow products to have different diffusion networks, which can be unknown in practice. Soma et al. (2014) studied the influence maximization problem for one product subject to one knapsack constraint over a known bipartite graph between marketing channels and potential customers; Ienco et al. (2010) and Sun et al. (2011) considered user constraints but disregarded product constraints during the initial assignment; and, Narayanam and Nanavati (2012) studied the cross-sell phenomenon (the selling of the first product raises the chance of selling the second) and included monetary constraints for all the products. However, no user constraints were considered, and the cost of each user was still uniform for each product. *Thus, our second goal is to design an efficient submodular maximization algorithm which can take into account both user and product constraints simultaneously.*

Overall, this article includes the following major contributions:

- Unlike prior work that considers an a priori described simplistic discrete-time diffusion model, we first **learn** the diffusion networks from data by using continuous-time diffusion models. This allows us to address the timing constraints in a principled way.

- We provide a novel formulation of the influence estimation problem in the continuous-time diffusion model from the perspective of probabilistic graphical models, which allows heterogeneous diffusion dynamics over the edges.

- We propose an efficient randomized algorithm for continuous-time influence estimation, which can scale up to millions of nodes and estimate the influence of each node to an accuracy of ϵ using $n = \mathcal{O}(1/\epsilon^2)$ randomizations.

- We formulate the influence maximization problem with the aforementioned constraints as a submodular maximization under the intersection of matroid constraints and knapsack constraints. The submodular function we use is based on the actual diffusion model learned from the data for the time window constraint. This novel formulation provides us a firm theoretical foundation for designing greedy algorithms with theoretical guarantees.

- We develop an efficient adaptive-threshold greedy algorithm which is linear in the number of products and proportional to $\widetilde{\mathcal{O}}(|\mathcal{V}| + |\mathcal{E}^*|)$, where $|\mathcal{V}|$ is the number of nodes (users) and $|\mathcal{E}^*|$ is the number of edges in the largest diffusion network. We then prove that this algorithm is guaranteed to find a solution with an overall influence of at least $\frac{k_a}{2+2k}$ of the optimal value, when k_a out of the k knapsack constraints are active. This improves over the best known approximation factor achieved by polynomial time algorithms in the combinatorial optimization literature. Moreover, whenever advertising each product to each user entails the same cost, the constraints reduce to an intersection of matroids, and we obtain an approximation factor of $1/3$, which is optimal for such optimization.

- We evaluate our algorithms over large synthetic and real-world datasets and show that our proposed methods significantly improve over previous state-of-the-arts in terms of both the accuracy of the estimated influence and the quality of the selected nodes in maximizing the influence over independently hold-out real testing data.

In the remainder of the paper, we will first tackle the influence estimation problem in section 2. We then formulate different realistic constraints for the influence maximization in section 3 and present the adaptive-thresholding greedy algorithm with its theoretical analysis in section 4; we investigate the performance of the proposed algorithms in both synthetic and real-world datasets in section 5; and finally we conclude in section 6.

2. Influence Estimation

We start by revisiting the continuous-time diffusion model by Gomez-Rodriguez et al. (2011) and then explicitly formulate the influence estimation problem from the perspective of probabilistic graphical models. Because the efficient inference of the influence value for each node is highly non-trivial, we further develop a scalable influence estimation algorithm which is able to handle networks of millions of nodes. The influence estimation procedure will be a key building block for our later influence maximization algorithm.

2.1 Continuous-Time Diffusion Networks

The continuous-time diffusion model associates each edge with a transmission function, that is, a density over the transmission time along the edge, in contrast to previous discrete-time models which associate each edge with a fixed infection probability (Kempe et al., 2003).

Moreover, it also differs from discrete-time models in the sense that events in a cascade are not generated iteratively in rounds, but event timings are sampled directly from the transmission function in the continuous-time model.

Continuous-Time Independent Cascade Model. Given a *directed* contact network, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we use the independent cascade model for modeling a diffusion process (Kempe et al., 2003; Gomez-Rodriguez et al., 2011). The process begins with a set of infected source nodes, $\mathcal{A}$, initially adopting certain contagion (idea, meme or product) at time zero. The contagion is transmitted from the sources along their out-going edges to their direct neighbors. Each transmission through an edge entails *random* waiting times, τ, drawn from different independent pairwise waiting time distributions(one per edge). Then, the infected neighbors transmit the contagion to their respective neighbors, and the process continues. We assume that an infected node remains infected for the entire diffusion process. Thus, if a node i is infected by multiple neighbors, only the neighbor that first infects node i will be the *true* parent. As a result, although the contact network can be an arbitrary directed network, each diffusion process induces a Directed Acyclic Graph (DAG).

Heterogeneous Transmission Functions. Formally, the pairwise transmission function $f_{ji}(t_i|t_j)$ for a directed edge $j \to i$ is the conditional density of node i getting infected at time t_i given that node j was infected at time t_j. We assume it is shift invariant: $f_{ji}(t_i|t_j) = f_{ji}(\tau_{ji})$, where $\tau_{ji} := t_i - t_j$, and causal: $f_{ji}(\tau_{ji}) = 0$ if $\tau_{ji} < 0$. Both parametric transmission functions, such as the exponential and Rayleigh function (Gomez-Rodriguez et al., 2011), and nonparametric functions (Du et al., 2012) can be used and estimated from cascade data.

Shortest-Path Property. The independent cascade model has a useful property we will use later: given a sample of transmission times of all edges, the time t_i taken to infect a node i is the length of the shortest path in $\mathcal{G}$ from the sources to node i, where the edge weights correspond to the associated transmission times.

2.2 Probabilistic Graphical Model for Continuous-Time Diffusion Networks

The continuous-time independent cascade model is essentially a directed graphical model for a set of *dependent* random variables, that is, the infection times t_i of the nodes, where the conditional independence structure is supported on the contact network $\mathcal{G}$. Although the original contact graph $\mathcal{G}$ can contain directed loops, each diffusion process (or a cascade) induces a directed acyclic graph (DAG). For those cascades consistent with a particular DAG, we can model the joint density of t_i using a directed graphical model:

$$p\left(\{t_i\}_{i\in\mathcal{V}}\right) = \prod_{i\in\mathcal{V}} p\left(t_i|\{t_j\}_{j\in\pi_i}\right), \tag{1}$$

where each π_i denotes the collection of parents of node i in the induced DAG, and each term $p(t_i|\{t_j\}_{j\in\pi_i})$ corresponds to a conditional density of t_i given the infection times of the parents of node i. This is true because given the infection times of node i's parents, t_i is independent of other infection times, satisfying the local Markov property of a directed graphical model. We note that the independent cascade model only specifies explicitly the pairwise transmission function of each directed edge, but does not directly define the conditional density $p(t_i|\{t_j\}_{j\in\pi_i})$.

However, these conditional densities can be derived from the pairwise transmission functions based on the Independent-Infection property:

$$p\left(t_i|\{t_j\}_{j\in\pi_i}\right) = \sum\nolimits_{j\in\pi_i} f_{ji}(t_i|t_j) \prod\nolimits_{l\in\pi_i, l\neq j} S(t_i|t_l), \tag{2}$$

which is the sum of the likelihoods that node i is infected by each parent node j. More precisely, each term in the summation can be interpreted as the likelihood $f_{ji}(t_i|t_j)$ of node i being infected at t_i by node j multiplied by the probability $S(t_i|t_l)$ that it has survived from the infection of each other parent node $l \neq j$ until time t_i.

Perhaps surprisingly, the factorization in Equation (1) is the same factorization that can be used for an arbitrary induced DAG consistent with the contact network $\mathcal{G}$. In this case, we only need to replace the definition of π_i (the parent of node i in the DAG) to the set of neighbors of node i with an edge pointing to node i in $\mathcal{G}$. This is not immediately obvious from Equation (1), since the contact network $\mathcal{G}$ can contain directed loops which seems to be in conflict with the conditional independence semantics of directed graphical models. The reason why it is possible to do so is as follows: any fixed set of infection times, $t_1, \ldots, t_d$, induces an ordering of the infection times. If $t_i \leq t_j$ for an edge $j \to i$ in $\mathcal{G}$, $h_{ji}(t_i|t_j) = 0$, and the corresponding term in Equation (2) is zeroed out, making the conditional density consistent with the semantics of directed graphical models.

Instead of directly modeling the infection times t_i, we can focus on the set of mutually *independent* random transmission times $\tau_{ji} = t_i - t_j$. Interestingly, by switching from a node-centric view to an edge-centric view, we obtain a fully factorized joint density of the set of transmission times

$$p\left(\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}\right) = \prod\nolimits_{(j,i)\in\mathcal{E}} f_{ji}(\tau_{ji}), \tag{3}$$

Based on the Shortest-Path property of the independent cascade model, each variable t_i can be viewed as a transformation from the collection of variables $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$. More specifically, let $\mathcal{Q}_i$ be the collection of directed paths in $\mathcal{G}$ from the source nodes to node i, where each path $q \in \mathcal{Q}_i$ contains a sequence of directed edges (j, l). Assuming all source nodes are infected at time zero, then we obtain variable t_i via

$$t_i = g_i\left(\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}|\mathcal{A}\right) = \min_{q\in\mathcal{Q}_i} \sum\nolimits_{(j,l)\in q} \tau_{jl}, \tag{4}$$

where the transformation $g_i(\cdot|\mathcal{A})$ is the value of the shortest-path minimization. As a special case, we can now compute the probability of node i infected before T using a set of independent variables:

$$\Pr\{t_i \leq T|\mathcal{A}\} = \Pr\left\{g_i\left(\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}|\mathcal{A}\right) \leq T\right\}. \tag{5}$$

The significance of the relation is that it allows us to transform a problem involving a sequence of dependent variables $\{t_i\}_{i\in\mathcal{V}}$ to one with independent variables $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$. Furthermore, the two perspectives are connected via the shortest path algorithm in weighted directed graph, a standard well-studied operation in graph analysis.

2.3 Influence Estimation Problem in Continuous-Time Diffusion Networks

Intuitively, given a time window, the wider the spread of infection, the more influential the set of sources. We adopt the definition of influence as the expected number of infected nodes given a set of source nodes and a time window, as in previous work (Gomez-Rodriguez and Schölkopf, 2012). More formally, consider a set of source nodes $\mathcal{A} \subseteq \mathcal{V}, |\mathcal{A}| \leq C$ which get infected at time zero. Then, given a time window T, a node i is infected within the time window if $t_i \leq T$. The expected number of infected nodes (or the influence) given the set of transmission functions $\{f_{ji}\}_{(j,i)\in\mathcal{E}}$ can be computed as

$$\sigma(\mathcal{A}, T) = \mathbb{E}\left[\sum_{i\in\mathcal{V}} \mathbb{I}\left\{t_i \leq T|\mathcal{A}\right\}\right] = \sum_{i\in\mathcal{V}} \Pr\left\{t_i \leq T|\mathcal{A}\right\}, \tag{6}$$

where $\mathbb{I}\{\cdot\}$ is the indicator function and the expectation is taken over the the set of *dependent* variables $\{t_i\}_{i\in\mathcal{V}}$. By construction, $\sigma(\mathcal{A}, T)$ is a non-negative, monotonic nondecreasing submodular function in the set of source nodes shown by Gomez-Rodriguez and Schölkopf (2012).

Essentially, the influence estimation problem in Equation (6) is an inference problem for graphical models, where the probability of event $t_i \leq T$ given sources in $\mathcal{A}$ can be obtained by summing out the possible configuration of other variables $\{t_j\}_{j\neq i}$. That is

$$\Pr\{t_i \leq T|\mathcal{A}\} = \int_0^\infty \cdots \int_{t_i=0}^{T} \cdots \int_0^\infty \left(\prod\nolimits_{j\in\mathcal{V}} p\left(t_j|\{t_l\}_{l\in\pi_j}\right)\right)\left(\prod\nolimits_{j\in\mathcal{V}} dt_j\right), \tag{7}$$

which is, in general, a very challenging problem. First, the corresponding directed graphical models can contain nodes with high in-degree and high out-degree. For example, in Twitter, a user can follow dozens of other users, and another user can have hundreds of "followers". The tree-width corresponding to this directed graphical model can be very high, and we need to perform integration for functions involving many continuous variables. Second, the integral in general can not be evaluated analytically for heterogeneous transmission functions, which means that we need to resort to numerical integration by discretizing the domain $[0, \infty)$. If we use N levels of discretization for each variable, we would need to enumerate $\mathcal{O}(N^{|\pi_i|})$ entries, exponential in the number of parents.

Only in very special cases, can one derive the closed-form equation for computing $\Pr\{t_i \leq T|\mathcal{A}\}$. For instance, Gomez-Rodriguez and Schölkopf (2012) proposed an approach for exponential transmission functions, where the special properties of exponential density are used to map the problem into a continuous time Markov process problem, and the computation can be carried out via a matrix exponential. However, without further heuristic approximation, the computational complexity of the algorithm is exponential in the size and density of the network. The intrinsic complexity of the problem entails the utilization of approximation algorithms, such as mean field algorithms or message passing algorithms. We will design an efficient randomized (or sampling) algorithm in the next section.

2.4 Efficient Influence Estimation in Continuous-Time Diffusion Networks

Our first key observation is that we can transform the influence estimation problem in Equation (6) into a problem with *independent* variables. With the relation in Equation (5),

we can derive the influence function as

$$\begin{aligned}\sigma(\mathcal{A}, T) =& \sum\nolimits_{i\in\mathcal{V}} \Pr\left\{g_i\left(\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}|\mathcal{A}\right) \leq T\right\} \\ =& \mathbb{E}\left[\sum\nolimits_{i\in\mathcal{V}} \mathbb{I}\left\{g_i\left(\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}|\mathcal{A}\right) \leq T\right\}\right],\end{aligned} \tag{8}$$

where the expectation is with respect to the set of independent variables $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$. This equivalent formulation suggests a naive sampling (NS) algorithm for approximating $\sigma(\mathcal{A}, T)$: draw n samples of $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$, run a shortest path algorithm for each sample, and finally average the results (see Appendix A for more details). However, this naive sampling approach has a computational complexity of $O(nC|\mathcal{V}||\mathcal{E}| + nC|\mathcal{V}|^2 \log|\mathcal{V}|)$ due to the repeated calling of the shortest path algorithm. This is quadratic to the network size, and hence not scalable to millions of nodes.

Our second key observation is that for each sample $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$, we are only interested in the neighborhood size of the source nodes, *i.e.*, the summation $\sum_{i\in\mathcal{V}} \mathbb{I}\{\cdot\}$ in Equation (8), rather than in the individual shortest paths. Fortunately, the neighborhood size estimation problem has been studied in the theoretical computer science literature. Here, we adapt a very efficient randomized algorithm by Cohen (1997) to our influence estimation problem. This randomized algorithm has a computational complexity of $O(|\mathcal{E}|\log|\mathcal{V}| + |\mathcal{V}|\log^2|\mathcal{V}|)$ and it estimates the neighborhood sizes for *all* possible single source node locations. Since it needs to run once for each sample of $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$, we obtain an overall influence estimation algorithm with $O(n|\mathcal{E}|\log|\mathcal{V}| + n|\mathcal{V}|\log^2|\mathcal{V}|)$ computation, nearly linear in network size. Next we will revisit Cohen's algorithm for neighborhood estimation.

2.4.1 Randomized Algorithm for Single-Source Neighborhood-Size Estimation

Given a fixed set of edge transmission times $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$ and a source node s, infected at time zero, the neighborhood $\mathcal{N}(s, T)$ of a source node s given a time window T is the set of nodes within distance T from s, *i.e.*,

$$\mathcal{N}(s, T) = \left\{i \mid g_i\left(\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}\right) \leq T,\ i \in \mathcal{V}\right\}. \tag{9}$$

Instead of estimating $\mathcal{N}(s, T)$ directly, the algorithm will assign an exponentially distributed random label r_i to each network node i. Then, it makes use of the fact that the minimum of a set of exponential random variables $\{r_i\}_{i\in\mathcal{N}(s,T)}$ is still an exponential random variable, but with its parameter being equal to the total number of variables, that is, if each $r_i \sim \exp(-r_i)$, then the smallest label within distance T from source s, $r_* := \min_{i\in\mathcal{N}(s,T)} r_i$, will distribute as $r_* \sim \exp\{-|\mathcal{N}(s, T)|r_*\}$. Suppose we randomize over the labeling m times and obtain m such least labels, $\{r_*^u\}_{u=1}^m$. Then the neighborhood size can be estimated as

$$|\mathcal{N}(s, T)| \approx \frac{m-1}{\sum_{u=1}^m r_*^u}. \tag{10}$$

which is shown by Cohen (1997) to be an unbiased estimator of $|\mathcal{N}(s, T)|$. This is an elegant relation since it allows us to transform the counting problem in (9) to a problem of finding the minimum random label r_*. The key question is whether we can compute the least label r_* efficiently, given random labels $\{r_i\}_{i\in\mathcal{V}}$ and any source node s.

Cohen (1997) designed a modified Dijkstra's algorithm (Algorithm 3) to construct a data structure $r_*(s)$, called least label list, for each node s to support such query. Essentially, the algorithm starts with the node i with the smallest label r_i, and then it traverses in breadth-first search fashion along the reverse direction of the graph edges to find all reachable nodes. For each reachable node s, the distance d_* between i and s, and r_i are added to the end of $r_*(s)$. Then the algorithm moves to the node i' with the second smallest label $r_{i'}$, and similarly find all reachable nodes. For each reachable node s, the algorithm will compare the current distance d_* between i' and s with the last recorded distance in $r_*(s)$. If the current distance is smaller, then the current d_* and $r_{i'}$ are added to the end of $r_*(s)$. Then the algorithm move to the node with the third smallest label and so on. The algorithm is summarized in Algorithm 3 in Appendix B.

Algorithm 3 returns a list $r_*(s)$ per node $s \in \mathcal{V}$, which contains information about distance to the smallest reachable labels from s. In particular, each list contains pairs of distance and random labels, (d, r), and these pairs are ordered as

$$\infty > d_{(1)} > d_{(2)} > \ldots > d_{(|r_*(s)|)} = 0 \tag{11}$$

$$r_{(1)} < r_{(2)} < \ldots < r_{(|r_*(s)|)}, \tag{12}$$

where $\{\cdot\}_{(l)}$ denotes the l-th element in the list. (see Appendix B for an example).

If we want to query the smallest reachable random label r_* for a given source s and a time T, we only need to perform a binary search on the list for node s:

$$r_* = r_{(l)}, \text{ where } d_{(l-1)} > T \geq d_{(l)}. \tag{13}$$

Finally, to estimate $|\mathcal{N}(s,T)|$, we generate m *i.i.d.* collections of random labels, run Algorithm 3 on each collection, and obtain m values $\{r_*^u\}_{u=1}^m$, which we use in Equation (10) to estimate $|\mathcal{N}(i,T)|$.

The computational complexity of Algorithm 3 is $O(|\mathcal{E}|\log|\mathcal{V}| + |\mathcal{V}|\log^2|\mathcal{V}|)$, with expected size of each $r_*(s)$ being $O(\log|\mathcal{V}|)$. Then the expected time for querying r_* is $O(\log\log|\mathcal{V}|)$ using binary search. Since we need to generate m set of random labels and run Algorithm 3 m times, the overall computational complexity for estimating the single-source neighborhood size for all $s \in \mathcal{V}$ is $O(m|\mathcal{E}|\log|\mathcal{V}| + m|\mathcal{V}|\log^2|\mathcal{V}| + m|\mathcal{V}|\log\log|\mathcal{V}|)$. For large-scale network, and when $m \ll \min\{|\mathcal{V}|, |\mathcal{E}|\}$, this randomized algorithm can be much more efficient than approaches based on directly calculating the shortest paths.

2.4.2 Constructing Estimation for Multiple-Source Neighborhood Size

When we have a set of sources, $\mathcal{A}$, its neighborhood is the union of the neighborhoods of its constituent sources

$$\mathcal{N}(\mathcal{A},T) = \bigcup\nolimits_{i\in\mathcal{A}} \mathcal{N}(i,T). \tag{14}$$

This is true because each source independently infects its downstream nodes. Furthermore, to calculate the least label list r_* corresponding to $\mathcal{N}(\mathcal{A},T)$, we can simply reuse the least label list $r_*(i)$ of each individual source $i \in \mathcal{A}$. More formally,

$$r_* = \min\nolimits_{i\in\mathcal{A}} \ \min\nolimits_{j\in\mathcal{N}(i,T)} r_j, \tag{15}$$

where the inner minimization can be carried out by querying $r_*(i)$. Similarly, after we obtain m samples of r_*, we can estimate $|\mathcal{N}(\mathcal{A},T)|$ using Equation (10). Importantly, very little additional work is needed when we want to calculate r_* for a set of sources $\mathcal{A}$, and we can reuse work done for a single source. This is very different from a naive sampling approach where the sampling process needs to be done completely anew if we increase the source set. In contrast, using the randomized algorithm, only an additional constant-time minimization over $|\mathcal{A}|$ numbers is needed.

2.4.3 Overall Algorithm

So far, we have achieved efficient neighborhood size estimation of $|\mathcal{N}(\mathcal{A},T)|$ with respect to a given set of transmission times $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$. Next, we will estimate the influence by averaging over multiple sets of samples for $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$. More specifically, the relation from (8)

$$\sigma(\mathcal{A},T) = \mathbb{E}_{\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}}[|\mathcal{N}(\mathcal{A},T)|] = \mathbb{E}_{\{\tau_{ji}\}}\mathbb{E}_{\{r^1,\ldots,r^m\}|\{\tau_{ji}\}}\left[\frac{m-1}{\sum_{u=1}^m r_*^u}\right], \tag{16}$$

suggests the following overall algorithm :

Continuous-Time Influence Estimation (ConTinEst):

1. Sample n sets of random transmission times $\{\tau_{ij}^l\}_{(j,i)\in\mathcal{E}} \sim \prod_{(j,i)\in\mathcal{E}} f_{ji}(\tau_{ji})$.
2. Given a set of $\{\tau_{ij}^l\}_{(j,i)\in\mathcal{E}}$, sample m sets of random labels $\{r_i^u\}_{i\in\mathcal{V}} \sim \prod_{i\in\mathcal{V}} \exp(-r_i)$.
3. Estimate $\sigma(\mathcal{A},T)$ by sample averages $\sigma(\mathcal{A},T) \approx \frac{1}{n}\sum_{l=1}^n \left((m-1)/\sum_{u_l=1}^m r_*^{u_l}\right)$.

What is even more important is that the number of random labels, m, does not need to be very large. Since the estimator for $|\mathcal{N}(A,T)|$ is unbiased (Cohen, 1997), essentially the outer-loop of averaging over n samples of random transmission times further reduces the variance of the estimator in a rate of $\mathcal{O}(1/n)$. In practice, we can use a very small m (*e.g.*, 5 or 10) and still achieve good results, which is also confirmed by our later experiments. Compared to (Chen et al., 2009), the novel application of Cohen's algorithm arises for estimating influence for multiple sources, which drastically reduces the computation by cleverly using the least-label list from single source. Moreover, we have the following theoretical guarantee (see Appendix C for the proof).

Theorem 1 *Draw the following number of samples for the set of random transmission times*

$$n \geq \frac{C\Lambda}{\epsilon^2}\log\left(\frac{2|\mathcal{V}|}{\alpha}\right) \tag{17}$$

where $\Lambda := \max_{\mathcal{A}:|\mathcal{A}|\leq C} 2\sigma(\mathcal{A},T)^2/(m-2) + 2Var(|\mathcal{N}(\mathcal{A},T)|)(m-1)/(m-2) + 2a\epsilon/3$ *and* $|\mathcal{N}(\mathcal{A},T)| \leq |\mathcal{V}|$*, and for each set of random transmission times, draw* m *sets of random labels. Then* $|\widehat{\sigma}(\mathcal{A},T) - \sigma(\mathcal{A},T)| \leq \epsilon$ *uniformly for all* $\mathcal{A}$ *with* $|\mathcal{A}| \leq C$*, with probability at least* $1-\alpha$*.*

The theorem indicates that the minimum number of samples, n, needed to achieve certain accuracy is related to the actual size of the influence $\sigma(\mathcal{A}, T)$, and the variance of the neighborhood size $|\mathcal{N}(\mathcal{A}, T)|$ over the random draw of samples. The number of random labels, m, drawn in the inner loop of the algorithm will monotonically decrease the dependency of n on $\sigma(\mathcal{A}, T)$. It suffices to draw a small number of random labels, as long as the value of $\sigma(\mathcal{A}, T)^2/(m-2)$ matches that of $Var(|\mathcal{N}(\mathcal{A}, T)|)$. Another implication is that influence at larger time window T is harder to estimate, since $\sigma(\mathcal{A}, T)$ will generally be larger and hence require more random samples.

3. Constraints of Practical Importance

By treating our proposed influence estimation algorithm ConTinEst as a building block, we can now tackle the influence maximization problem under various constraints of practical importance. Here, since ConTinEst can estimate the influence value of any source set with respect to any given time window T, the **Timing Constraints** can thus be naturally satisfied. Therefore, in the following sections, we mainly focus on modeling the **Multiple-Item Constraints**, the **User Constraints** and the **Product Constraints**.

3.1 Multiple-Item Constraints

Multiple products can spread simultaneously across the same set of social entities over different diffusion channels. Since these products may have different characteristics, such as the revenue and the speed of spread, and thus may follow different diffusion dynamics, we will use multiple diffusion networks for different types of products.

Suppose we have a set of products $\mathcal{L}$ that propagate on the same set of nodes $\mathcal{V}$. The diffusion network for product i is denoted as $\mathcal{G}_i = (\mathcal{V}, \mathcal{E}_i)$. For each product $i \in \mathcal{L}$, we search for a set of source nodes $\mathcal{R}_i \subseteq \mathcal{V}$ to which we can assign the product i to start its campaign. We can represent the selection of $\mathcal{R}_i$'s using an assignment matrix $A \in \{0,1\}^{|\mathcal{L}|\times|\mathcal{V}|}$ as follows: $A_{ij} = 1$ if $j \in \mathcal{R}_i$ and $A_{ij} = 0$ otherwise. Based on this representation, we define a new ground set $\mathcal{Z} = \mathcal{L} \times \mathcal{V}$ of size $N = |\mathcal{L}| \times |\mathcal{V}|$. Each element of $\mathcal{Z}$ corresponds to the index (i, j) of an entry in the assignment matrix A, and selecting element $z = (i, j)$ means assigning product i to user j (see Figure 1 for an illustration). We also denote $\mathcal{Z}_{*j} = \mathcal{L} \times \{j\}$ and $\mathcal{Z}_{i*} = \{i\} \times \mathcal{V}$ as the j-th column and i-th row of matrix A, respectively. Then, under the above mentioned additional requirements, we would like to find a set of assignments $\mathcal{S} \subseteq \mathcal{Z}$ so as to maximize the following **overall influence**

$$f(S) = \sum_{i\in\mathcal{L}} a_i \sigma_i(\mathcal{R}_i, T_i), \tag{18}$$

where $\sigma_i(\mathcal{R}_i, T_i)$ denote the influence of product i for a given time T_i, $\{a_i > 0\}$ is a set of weights reflecting the different benefits of the products and $\mathcal{R}_i = \{j \in \mathcal{V} : (i, j) \in S\}$. We now show that the overall influence function $f(S)$ in Equation (18) is submodular over the ground set $\mathcal{Z}$.

Lemma 2 *Under the continuous-time independent cascade model, the overall influence $f(S)$ is a normalized monotone submodular function of S.*

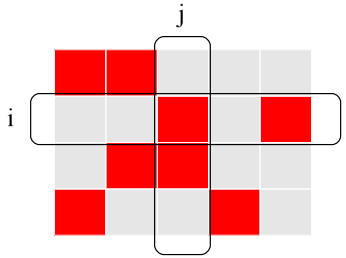


Figure 1: Illustration of the assignment matrix A associated with partition matroid $\mathcal{M}_1$ and group knapsack constraints. If product i is assigned to user j, then $A_{ij} = 1$ (colored in red). The ground set $\mathcal{Z}$ is the set of indices of the entries in A, and selecting an element $(i, j) \in \mathcal{Z}$ means assigning product i to user j. The user constraint means that there are at most u_j elements selected in the j-th column; the product constraint means that the total cost of the elements selected in the i-th row is at most B_i.

Proof By definition, $f(\emptyset) = 0$ and $f(S)$ is monotone. By Theorem 4 in Gomez-Rodriguez and Schölkopf (2012), the component influence function $\sigma_i(\mathcal{R}_i, T_i)$ for product i is submodular in $\mathcal{R}_i \subseteq \mathcal{V}$. Since non-negative linear combinations of submodular functions are still submodular, $f_i(S) := a_i\sigma_i(\mathcal{R}_i, T_i)$ is also submodular in $S \subseteq \mathcal{Z} = \mathcal{L}\times\mathcal{V}$, and $f(S) = \sum_{i\in\mathcal{L}} f_i(S)$ is submodular. ■

3.2 User Constraints

Each social network user can be a potential source and would like to be exposed only to a small number of ads. Furthermore, users may be grouped according to their geographical locations, and advertisers may have a target population they want to reach. Here, we will incorporate these constraints using the matroids which are combinatorial structures that generalize the notion of linear independence in matrices (Schrijver, 2003; Fujishige, 2005). Formulating our constrained influence maximization task with matroids allows us to design a greedy algorithm with provable guarantees.

Formally, suppose that each user j can be assigned to at most u_j products. A matroid can be defined as follows:

Definition 3 *A matroid is a pair, $\mathcal{M} = (\mathcal{Z}, \mathcal{I})$, defined over a finite set (the ground set) $\mathcal{Z}$ and a family of sets (the independent sets) $\mathcal{I}$, that satisfies three axioms:*

1. *Non-emptiness: The empty set $\emptyset \in \mathcal{I}$.*
2. *Heredity: If $Y \in \mathcal{I}$ and $X \subseteq Y$, then $X \in \mathcal{I}$.*
3. *Exchange: If $X \in \mathcal{I}, Y \in \mathcal{I}$ and $|Y| > |X|$, then there exists $z \in Y \setminus X$ such that $X \cup \{z\} \in \mathcal{I}$.*

An important type of matroid is the partition matroid where the ground set $\mathcal{Z}$ is partitioned into disjoint subsets $\mathcal{Z}_1, \mathcal{Z}_2, \ldots, \mathcal{Z}_t$ for some t and

$$\mathcal{I} = \{S \mid S \subseteq \mathcal{Z} \text{ and } |S \cap \mathcal{Z}_i| \leq u_i, \forall i = 1, \ldots, t\}$$

for some given parameters $u_1, \ldots, u_t$. The user constraints can then be formulated as

> Partition matroid $\mathcal{M}_1$: partition the ground set $\mathcal{Z}$ into $\mathcal{Z}_{*j} = \mathcal{L} \times \{j\}$ each of which corresponds to a column of A. Then $\mathcal{M}_1 = \{\mathcal{Z}, \mathcal{I}_1\}$ is
>
> $$\mathcal{I}_1 = \{S | S \subseteq \mathcal{Z} \text{ and } |S \cap \mathcal{Z}_{*j}| \leq u_j, \forall j\}.$$

3.3 Product Constraints

Seeking initial adopters entails a cost to the advertiser, which needs to be paid to the host, while the advertisers of each product have a limited amount of money. Here, we will incorporate these requirements using knapsack constraints which we describe below.

Formally, suppose that each product i has a budget B_i, and assigning item i to user j costs $c_{ij} > 0$. Next, we introduce the following notation to describe product constraints over the ground set $\mathcal{Z}$. For an element $z = (i, j) \in \mathcal{Z}$, define its cost as $c(z) := c_{ij}$. Abusing the notation slightly, we denote the cost of a subset $S \subseteq \mathcal{Z}$ as $c(S) := \sum_{z \in S} c(z)$. Then, in a feasible solution $S \subseteq \mathcal{Z}$, the cost of assigning product i, $c(S \cap \mathcal{Z}_{i*})$, should not be larger than its budget B_i.

Now, without loss of generality, we can assume $B_i = 1$ (by normalizing c_{ij} with B_i), and also $c_{ij} \in (0, 1]$ (by throwing away any element (i, j) with $c_{ij} > 1$), and define

> Group-knapsack: partition the ground set into $\mathcal{Z}_{i*} = \{i\} \times \mathcal{V}$ each of which corresponds to one row of A. Then a feasible solution $S \subseteq \mathcal{Z}$ satisfies
>
> $$c(S \cap \mathcal{Z}_{i*}) \leq 1, \forall i.$$

Importantly, these knapsack constraints have very specific structure: they are on different groups of a partition $\{\mathcal{Z}_{i*}\}$ of the ground set and the submodular function $f(S) = \sum_i a_i \sigma_i(\mathcal{R}_i, T_i)$ is defined over the partition. In consequence, such structures allow us to design an efficient algorithm with improved guarantees over the known results.

3.4 Overall Problem Formulation

Based on the above discussion of various constraints in viral marketing and our design choices for tackling them, we can think of the influence maximization problem as a special case of the following constrained submodular maximization problem with $P = 1$ matroid constraints and $k = |\mathcal{L}|$ knapsack constraints,

$$\begin{aligned} \max_{S \subseteq \mathcal{Z}} \quad & f(S) \\ \text{subject to} \quad & c(S \cap \mathcal{Z}_{i*}) \leq 1, \quad 1 \leq i \leq k, \\ & S \in \bigcap_{p=1}^{P} \mathcal{I}_p, \end{aligned} \tag{19}$$

where, for simplicity, we will denote all the feasible solutions $S \subseteq \mathcal{Z}$ as $\mathcal{F}$. This formulation in general includes the following cases of practical importance :

Uniform User-Cost. An important case of influence maximization, which we denote as the Uniform Cost, is that for each product i, all users have the same cost c_{i*}, *i.e.*, $c_{ij} = c_{i*}$. Equivalently, each product i can be assigned to at most $b_i := \lfloor B_i / c_{i*} \rfloor$ users. Then the product constraints are simplified to

> Partition matroid $\mathcal{M}_2$: for the product constraints with uniform cost, define a matroid $\mathcal{M}_2 = \{\mathcal{Z}, \mathcal{I}_2\}$ where
>
> $$\mathcal{I}_2 = \{S | S \subseteq \mathcal{Z} \text{ and } |S \cap \mathcal{Z}_{i*}| \leq b_i, \forall i\}.$$

In this case, the influence maximization problem defined by Equation (19) becomes the problem with $P = 2$ matroid constraints and no knapsack constraints ($k = 0$). In addition, if we assume only one product needs campaign, the formulation of Equation (19) further reduces to the classic influence maximization problem with the simple cardinality constraint.

User Group Constraint. Our formulation in Equation (19) essentially allows for general matroids which can model more sophisticated real-world constraints, and the proposed formulation, algorithms, and analysis can still hold. For instance, suppose there is a hierarchical community structure on the users, *i.e.*, a tree $\mathcal{T}$ where leaves are the users and the internal nodes are communities consisting of all users underneath, such as customers in different countries around the world. In consequence of marketing strategies, on each community $C \in \mathcal{T}$, there are at most u_C slots for assigning the products. Such constraints are readily modeled by the Laminar Matroid, which generalizes the partition matroid by allowing the set $\{\mathcal{Z}_i\}$ to be a laminar family (*i.e.*, for any $\mathcal{Z}_i \neq \mathcal{Z}_j$, either $\mathcal{Z}_i \subseteq \mathcal{Z}_j$, or $\mathcal{Z}_j \subseteq \mathcal{Z}_i$, or $\mathcal{Z}_i \cap \mathcal{Z}_j = \emptyset$). It can be shown that the community constraints can be captured by the matroid $\mathcal{M} = (\mathcal{Z}, \mathcal{I})$ where $\mathcal{I} = \{S \subseteq \mathcal{Z} : |S \cap C| \leq u_C, \forall C \in \mathcal{T}\}$. In the next section, we first present our algorithm, then provide the analysis for the uniform cost case and finally leverage such analysis for the general case.

4. Influence Maximization

In this section, we first develop a simple, practical and intuitive adaptive-thresholding greedy algorithm to solve the continuous-time influence maximization problem with the aforementioned constraints. Then, we provide a detailed theoretical analysis of its performance.

4.1 Overall Algorithm

There exist algorithms for submodular maximization under multiple knapsack constraints achieving a $1 - \frac{1}{e}$ approximation factor by (Sviridenko, 2004). Thus, one may be tempted to convert the matroid constraint in the problem defined by Equation (19) to $|\mathcal{V}|$ knapsack constraints, so that the problem becomes a submodular maximization problem under $|\mathcal{L}| + |\mathcal{V}|$ knapsack constraints. However, this naive approach is not practical for large-scale scenarios because the running time of such algorithms is exponential in the number of knapsack constraints. Instead, if we opt for algorithms for submodular maximization under k knapsack constraints and P matroids constraints, the best approximation factor achieved by polynomial time algorithms is $\frac{1}{P+2k+1}$ (Badanidiyuru and Vondrák, 2014). However,

Algorithm 1: Density Threshold Enumeration

Input: parameter δ; objective f or its approximation $\widehat{f}$; assignment cost $c(z), z \in \mathcal{Z}$

Set $d = \max\{f(\{z\}) : z \in \mathcal{Z}\}$;

for $\rho \in \left\{\frac{2d}{P+2k+1}, (1+\delta)\frac{2d}{P+2k+1}, \ldots, \frac{2|\mathcal{Z}|d}{P+2k+1}\right\}$ **do**

  Call Algorithm 2 to get S_ρ;

Output: $\operatorname{argmax}_{S_\rho} f(S_\rho)$

this is not good enough yet, since in our problem $k = |\mathcal{L}|$ can be large, though $P = 1$ is small.

Here, we will design an algorithm that achieves a better approximation factor by exploiting the following key observation about the structure of the problem defined by Equation (19): the knapsack constraints are over different groups $\mathcal{Z}_{i*}$ of the whole ground set, and the objective function is a sum of submodular functions over these different groups.

The details of the algorithm, called BudgetMax, are described in Algorithm 1. BudgetMax enumerates different values of a so-called density threshold ρ, runs a subroutine to find a solution for each ρ, which quantifies the cost-effectiveness of assigning a particular product to a specific user, and finally outputs the solution with the maximum objective value. Intuitively, the algorithm restricts the search space to be the set of most cost-effective allocations. The details of the subroutine to find a solution for a fixed density threshold ρ are described in Algorithm 2. Inspired by the lazy evaluation heuristic (Leskovec et al., 2007), the algorithm maintains a working set G and a marginal gain threshold w_t, which geometrically decreases by a factor of $1+\delta$ until it is sufficiently small to be set to zero. At each w_t, the subroutine selects each new element z that satisfies the following properties:

1. It is feasible and the density ratio (the ratio between the marginal gain and the cost) is above the current density threshold;

2. Its marginal gain
$$f(z|G) := f(G \cup \{z\}) - f(G)$$
is above the current marginal gain threshold.

The term "density" comes from the knapsack problem, where the marginal gain is the mass and the cost is the volume. A large density means gaining a lot without paying much. In short, the algorithm considers only high-quality assignments and repeatedly selects feasible ones with marginal gain ranging from large to small.

Remark 1. The traditional lazy evaluation heuristic also keeps a threshold, however, it only uses the threshold to speed up selecting the element with maximum marginal gain. Instead, Algorithm 2 can add multiple elements z from the ground set at each threshold according to the descending order of their marginal gains, and thus reduces the number of rounds from the size of the solution to the number of thresholds $\mathcal{O}(\frac{1}{\delta}\log\frac{N}{\delta})$. This allows us to trade off between the runtime and the approximation ratio (refer to our theoretical guarantees in section 4.2).

Algorithm 2: Adaptive Threshold Greedy for Fixed Density

Input: parameters ρ, δ; objective f or its approximation $\widehat{f}$; assignment cost $c(z), z \in \mathcal{Z}$;set of feasible solutions $\mathcal{F}$; and d from Algorithm 1.

Set $d_\rho = \max\{f(\{z\}) : z \in \mathcal{Z}, f(\{z\}) \geq c(z)\rho\}$;

Set $w_t = \frac{d_\rho}{(1+\delta)^t}$ for $t = 0, \ldots, L = \operatorname{argmin}_i\left[w_i \leq \frac{\delta d}{N}\right]$ and $w_{L+1} = 0$;

Set $G = \emptyset$;

for $t = 0, 1, \ldots, L, L+1$ **do**

  for $z \notin G$ *with* $G \cup \{z\} \in \mathcal{F}$ *and* $f(z|G) \geq c(z)\rho$ **do**

    if $f(z|G) \geq w_t$ **then**

      Set $G \leftarrow G \cup \{z\}$;

Output: $S_\rho = G$

Remark 2. Evaluating the influence of the assigned products f is expensive. Therefore, we will use the randomized algorithm in Section 2.4.3 to compute an estimation $\widehat{f}(\cdot)$ of the quantity $f(\cdot)$.

4.2 Theoretical Guarantees

Although our algorithm is quite intuitive, it is highly non-trivial to obtain the theoretical guarantees. For clarity, we first analyze the simpler case with uniform cost, which then provides the base for analyzing the general case.

4.2.1 Uniform Cost

As shown at the end of Section 3.4, the influence maximization, in this case, corresponds to the problem defined by Equation (19) with $P = 2$ and no knapsack constraints. Thus, we can simply run Algorithm 2 with $\rho = 0$ to obtain a solution G, which is roughly a $\frac{1}{P+1}$-approximation.

Intuition. The algorithm greedily selects feasible elements with sufficiently large marginal gain. However, it is unclear whether our algorithm will find *good* solutions and whether it will be *robust* to noise. Regarding the former, one might wonder whether the algorithm will select just a few elements while many elements in the optimal solution O will become infeasible and will not be selected, in which case the greedy solution G is a poor approximation. Regarding the latter, we only use the estimation $\widehat{f}$ of the influence f (*i.e.*, $|\widehat{f}(S) - f(S)| \leq \epsilon$ for any $S \subseteq \mathcal{Z}$), which introduces additional error to the function value. A crucial question, which has not been addressed before (Badanidiyuru and Vondrák, 2014), is whether the adaptive threshold greedy algorithm is robust to such perturbations.

Fortunately, it turns out that the algorithm will provably select sufficiently many elements of high quality. First, the elements selected in the optimal solution O but not selected in G can be partitioned into $|G|$ groups, each of which is associated with an element in G. Thus, the number of elements in the groups associated with the first t elements in G, by the property of the intersection of matroids, are bounded by Pt. See Figure 2 for an illustration. Second, the marginal gain of each element in G is at least as large as that of any

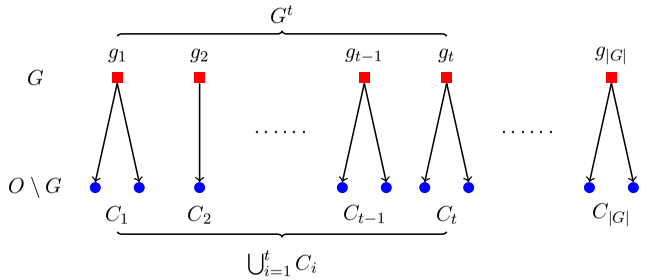


Figure 2: Notation for analyzing Algorithm 2. The elements in the greedy solution G are arranged according to the order in which Algorithm 2 selects them in Step 3. The elements in the optimal solution O but not in the greedy solution G are partitioned into groups $\{C_t\}_{1\leq t\leq |G|}$, where C_t are those elements in $O \setminus G$ that are still feasible before selecting g_t but are infeasible after selecting g_t.

element in the group associated with it (up to some small error). This means that even if the submodular function evaluation is inexact, the quality of the elements in the greedy solution is still good. The two claims together show that the marginal gain of $O \setminus G$ is not much larger than the gain of G, and thus G is a good approximation for the problem.

Formally, suppose we use an inexact evaluation of the influence f such that $|\widehat{f}(S) - f(S)| \leq \epsilon$ for any $S \subseteq \mathcal{Z}$, let product $i \in \mathcal{L}$ spread according to a diffusion network $\mathcal{G}_i = (\mathcal{V}, \mathcal{E}_i)$, and $i^* = \operatorname{argmax}_{i\in\mathcal{L}} |\mathcal{E}_i|$. Then, we have:

Theorem 4 *Suppose $\widehat{f}$ is evaluated up to error $\epsilon = \delta/16$ with* ConTinEst. *For influence maximization with uniform cost, Algorithm 2 (with $\rho = 0$) outputs a solution G with $f(G) \geq \frac{1-2\delta}{3} f(O)$ in expected time $\widetilde{\mathcal{O}}\left(\frac{|\mathcal{E}_{i^*}|+|\mathcal{V}|}{\delta^2} + \frac{|\mathcal{L}||\mathcal{V}|}{\delta^3}\right)$.*

The parameter δ introduces a tradeoff between the approximation guarantee and the runtime: larger δ decreases the approximation ratio but results in fewer influence evaluations. Moreover, the runtime has a linear dependence on the network size and the number of products to propagate (ignoring some small logarithmic terms) and, as a consequence, the algorithm is scalable to large networks.

Analysis. Suppose $G = \{g_1, \ldots, g_{|G|}\}$ in the order of selection, and let $G^t = \{g_1, \ldots, g_t\}$. Let C_t denote all those elements in $O \setminus G$ that satisfy the following: they are still feasible before selecting the t-th element g_t but are infeasible after selecting g_t. Equivalently, C_t are all those elements $j \in O \setminus G$ such that (1) $j \cup G^{t-1}$ does not violate the matroid constraints but (2) $j \cup G^t$ violates the matroid constraints. In other words, we can think of C_t as the optimal elements "blocked" by g_t. Then, we proceed as follows.

By the property of the intersection of matroids, the size of the prefix $\bigcup_{i=1}^t C_t$ is bounded by Pt. As a consequence of this property, for any $Q \subseteq \mathcal{Z}$, the sizes of any two maximal independent subsets T_1 and T_2 of Q can only differ by a multiplicative factor at most P. This can be realized with the following argument. First, note that any element $z \in T_1 \setminus T_2$, $\{z\} \cup T_2$ violates at least one of the matroid constraints since T_2 is maximal. Then, let $\{V_i\}_{1\leq i\leq P}$ denote all elements in $T_1 \setminus T_2$ that violate the i-th matroid, and partition $T_1 \cap T_2$ arbitrarily among these V_i's so that they cover T_1. In this construction, the size of each V_i must be at most $|T_2|$, since otherwise by the Exchange axiom, there would exist $z \in V_i \setminus T_2$ that can be added to T_2, without violating the i-th matroid, leading to a contradiction. Therefore, $|T_1|$ is at most P times $|T_2|$.

Next, we apply the above property as follows. Let Q be the union of G^t and $\bigcup_{i=1}^t C_t$. On one hand, G^t is a maximal independent subset of Q, since no element in $\bigcup_{i=1}^t C_t$ can be added to G^t without violating the matroid constraints. On the other hand, $\bigcup_{i=1}^t C_t$ is an independent subset of Q, since it is part of the optimal solution. Therefore, $\bigcup_{i=1}^t C_t$ has size at most P times $|G^t|$, which is Pt. Note that the properties of matroids are crucial for this analysis, which justifies our formulation using matroids. In summary, we have

Claim 1 $\sum_{i=1}^t |C_i| \leq Pt$, *for* $t = 1, \ldots, |G|$.

Now, we consider the marginal gain of each element in C_t associated with g_t. First, suppose g_t is selected at the threshold $\tau_t > 0$. Then, any $j \in C_t$ has marginal gain bounded by $(1+\delta)\tau_t + 2\epsilon$, since otherwise j would have been selected at a larger threshold before τ_t by the greedy criterion. Second, suppose g_t is selected at the threshold $w_{L+1} = 0$. Then, any $j \in C_t$ has marginal gain approximately bounded by $\frac{\delta}{N} d$. Since the greedy algorithm must pick g_1 with $\widehat{f}(g_1) = d$ and $d \leq f(g_1) + \epsilon$, any $j \in C_t$ has marginal gain bounded by $\frac{\delta}{N} f(G) + \mathcal{O}(\epsilon)$. Putting everything together we have:

Claim 2 *Suppose g_t is selected at the threshold τ_t. Then $f(j|G^{t-1}) \leq (1+\delta)\tau_t + 4\epsilon + \frac{\delta}{N} f(G)$ for any $j \in C_t$.*

Since the evaluation of the marginal gain of g_t should be at least τ_t, this claims essentially indicates that the marginal gain of j is approximately bounded by that of g_t.

Since there are not many elements in C_t (Claim 1) and the marginal gain of each of its elements is not much larger than that of g_t (Claim 2), we can conclude that the marginal gain of $O \setminus G = \bigcup_{i=1}^{|G|} C_t$ is not much larger than that of G, which is just $f(G)$.

Claim 3 *The marginal gain of $O \setminus G$ satisfies*

$$\sum_{j\in O\setminus G} f(j|G) \leq [(1+\delta)P + \delta] f(G) + (6 + 2\delta)\epsilon P|G|.$$

Finally, since by submodularity, $f(O) \leq f(O \cup G) \leq f(G) + \sum_{j\in O\setminus G} f(j|G)$, Claim 3 shows that $f(G)$ is close to $f(O)$ up to a multiplicative factor roughly $(1+P)$ and additive factor $\mathcal{O}(\epsilon P|G|)$. Given that $f(G) > |G|$, it leads to roughly a 1/3-approximation for our influence maximization problem by setting $\epsilon = \delta/16$ when evaluating $\widehat{f}$ with ConTinEst. Combining the above analysis and the runtime of the influence estimation algorithm, we have our final guarantee in Theorem 4. Appendix D.1 presents the complete proofs.

4.2.2 General Case

In this section, we consider the general case, in which users may have different associated costs. Recall that this case corresponds to the problem defined by Equation (19) with $P = 1$ matroid constraints and $k = |\mathcal{L}|$ group-knapsack constraints. Here, we will show that there is a step in Algorithm 1 which outputs a solution S_ρ that is a good approximation.

Intuition. The key idea behind Algorithm 1 and Algorithm 2 is simple: spend the budgets *efficiently* and spend them *as much as possible*. By spending them efficiently, we mean to only select those elements whose density ratio between the marginal gain and the cost is above the threshold ρ. That is, we assign product i to user j only if the assignment leads to large marginal gain without paying too much. By spending the budgets as much as possible, we mean to stop assigning product i only if its budget is almost exhausted or no more assignments are possible without violating the matroid constraints. Here we make use of the special structure of the knapsack constraints on the budgets: each constraint is only related to the assignment of the corresponding product and its budget, so that when the budget of one product is exhausted, it does not affect the assignment of the other products. In the language of submodular optimization, the knapsack constraints are on a partition $\mathcal{Z}_{i*}$ of the ground set and the objective function is a sum of submodular functions over the partition.

However, there seems to be a hidden contradiction between spending the budgets efficiently and spending them as much as possible. On one hand, efficiency means the density ratio should be large, so the threshold ρ should be large; on the other hand, if ρ is large, there are just a few elements that can be considered, and thus the budget might not be exhausted. After all, if we set ρ to be even larger than the maximum possible value, then no element is considered and no gain is achieved. In the other extreme, if we set $\rho = 0$ and consider all the elements, then a few elements with large costs may be selected, exhausting all the budgets and leading to a poor solution.

Fortunately, there exists a suitable threshold ρ that achieves a good tradeoff between the two and leads to a good approximation. On one hand, the threshold is sufficiently small, so that the optimal elements we abandon (*i.e.*, those with low-density ratio) have a total gain at most a fraction of the optimum; on the other hand, it is also sufficiently large, so that the elements selected are of high quality (*i.e.*, of high-density ratio), and we achieve sufficient gain even if the budgets of some items are exhausted.

Theorem 5 *Suppose $\widehat{f}$ is evaluated up to error $\epsilon = \delta/16$ with* ConTinEst. *In Algorithm 1, there exists a ρ such that*

$$f(S_\rho) \geq \frac{\max\{k_a, 1\}}{(2|\mathcal{L}|+2)(1+3\delta)} f(O)$$

where k_a is the number of active knapsack constraints:

$$k_a = |\{i : S_\rho \cup \{z\} \notin \mathcal{F}, \forall z \in \mathcal{Z}_{i*}\}|.$$

The expected running time is $\widetilde{\mathcal{O}}\left(\frac{|\mathcal{E}_{i^*}|+|\mathcal{V}|}{\delta^2} + \frac{|\mathcal{L}||\mathcal{V}|}{\delta^4}\right)$.

Importantly, the approximation factor improves over the best known guarantee $\frac{1}{P+2k+1} = \frac{1}{2|\mathcal{L}|+2}$ for efficiently maximizing submodular functions over P matroids and k general knapsack constraints. Moreover, since the runtime has a linear dependence on the network size, the algorithm easily scales to large networks. As in the uniform cost case, the parameter δ introduces a tradeoff between the approximation and the runtime.

Analysis. The analysis follows the intuition. Pick $\rho = \frac{2f(O)}{P+2k+1}$, where O is the optimal solution, and define

$$O_- := \{z \in O \setminus S_\rho : f(z|S_\rho) < c(z)\rho + 2\epsilon\},$$
$$O_+ := \{z \in O \setminus S_\rho : z \notin O_-\}.$$

Note that, by submodularity, O_- is a superset of the elements in the optimal solution that we abandon due to the density threshold and, by construction, its marginal gain is small:

$$f(O_-|S_\rho) \leq \rho c(O_-) + \mathcal{O}(\epsilon|S_\rho|) \leq k\rho + \mathcal{O}(\epsilon|S_\rho|),$$

where the small additive term $\mathcal{O}(\epsilon|S_\rho|)$ is due to inexact function evaluations. Next, we proceed as follows.

First, if no knapsack constraints are active, then the algorithm runs as if there were no knapsack constraints (but only on elements with density ratio above ρ). Therefore, we can apply the same argument as in the case of uniform cost (refer to the analysis up to Claim 3 in Section 4.2.1); the only caveat is that we apply the argument to O_+ instead of $O \setminus S_\rho$. Formally, similar to Claim 3, the marginal gain of O_+ satisfies

$$f(O_+|S_\rho) \leq [(1+\delta)P + \delta] f(S_\rho) + \mathcal{O}(\epsilon P|S_\rho|),$$

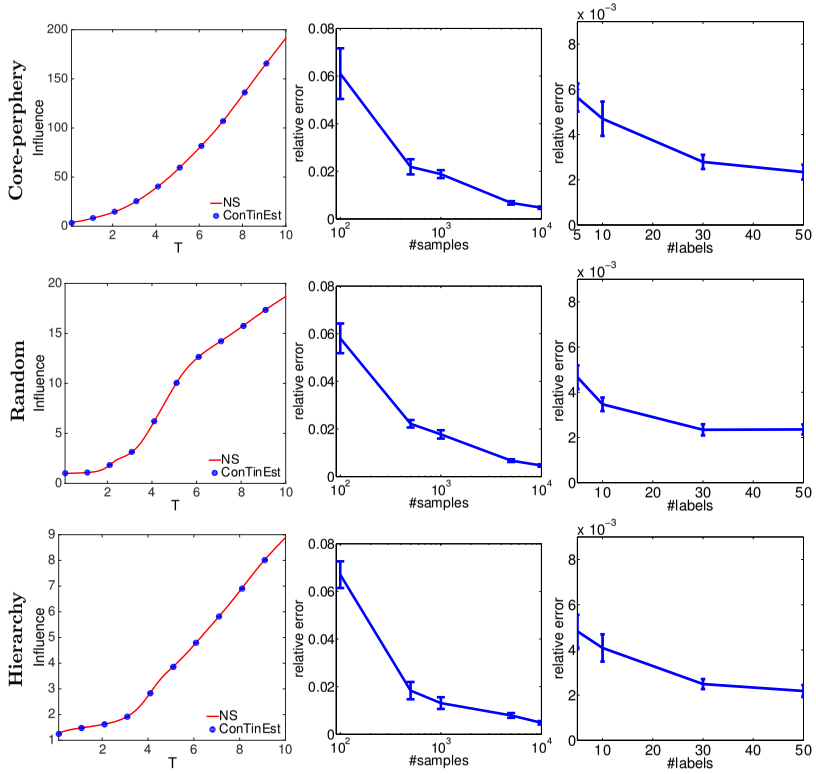
where the small additive term $\mathcal{O}(\epsilon P|S_\rho|)$ is due to inexact function evaluations. Using that $f(O) \leq f(S_\rho) + f(O_-|S_\rho) + f(O_+|S_\rho)$, we can conclude that S_ρ is roughly a $\frac{1}{P+2k+1}$-approximation.

Second, suppose $k_a > 0$ knapsack constraints are active and the algorithm discovers that the budget of product i is exhausted when trying to add element z to the set $G_i = G \cap \mathcal{Z}_{i*}$ of selected elements at that time. Since $c(G_i \cup \{z\}) > 1$ and each of these elements has density above ρ, the gain of $G_i \cup \{z\}$ is above ρ. However, only G_i is included in our final solution, so we need to show that the marginal gain of z is not large compared to that of G_i. To do so, we first realize that the algorithm greedily selects elements with marginal gain above a decreasing threshold w_t. Then, since z is the last element selected and G_i is nonempty (otherwise adding z will not exhaust the budget), the marginal gain of z must be bounded by roughly that of G_i, which is at least roughly $\frac{1}{2}\rho$. Since this holds for all active knapsack constraints, then the solution has value at least $\frac{k_a}{2}\rho$, which is an $\frac{k_a}{P+2k+1}$-approximation.

Finally, combining both cases, and setting $k = |\mathcal{L}|$ and $P = 1$ as in our problem, we have our final guarantee in Theorem 5. Appendix D.2 presents the complete proofs.

5. Experiments on Synthetic and Real Data

In this section, we first evaluate the accuracy of the estimated influence given by ConTinEst and then investigate the performance of influence maximization on synthetic and real networks by incorporating ConTinEst into the framework of BudgetMax. We show that our approach significantly outperforms the state-of-the-art methods in terms of both speed and solution quality.



(a) Influence vs. time (b) Influence vs. #samples (c) Error vs. #labels

Figure 3: Influence estimation for core-periphery, random, and hierarchical networks with 1,024 nodes and 2,048 edges. Column (a) shows estimated influence by NS (near ground truth), and ConTinEst for increasing time window T; Column (b) shows ConTinEst's relative error against number of samples with 5 random labels and $T = 10$; Column (c) reports ConTinEst's relative error against the number of random labels with 10,000 random samples and $T = 10$.

5.1 Experiments on Synthetic Data

We generate three types of Kronecker networks (Leskovec et al., 2010) which are synthetic networks generated by a recursive Kronecker product of a base 2-by-2 parameter matrix with itself to generate self-similar graphs. By tuning the base parameter matrix, we are able to generate the Kronecker networks which can mimic different structural properties of many real networks. In the following, we consider networks of three different types of structures: (*i*) core-periphery networks (parameter matrix: [0.9 0.5; 0.5 0.3]), which mimic the information diffusion traces in real-world networks (Gomez-Rodriguez et al., 2011), (*ii*) random networks ([0.5 0.5; 0.5 0.5]), typically used in physics and graph theory (Easley and Kleinberg, 2010) and (*iii*) hierarchical networks ([0.9 0.1; 0.1 0.9]) (Clauset et al., 2008). Next, we assign a pairwise transmission function for every directed edge in each type of network and set its parameters at random. In our experiments, we use the Weibull distribution from (Aalen et al., 2008),

$$f(t;\alpha,\beta) = \frac{\beta}{\alpha}\left(\frac{t}{\alpha}\right)^{\beta-1} e^{-(t/\alpha)^{\beta}}, t \geq 0, \tag{20}$$

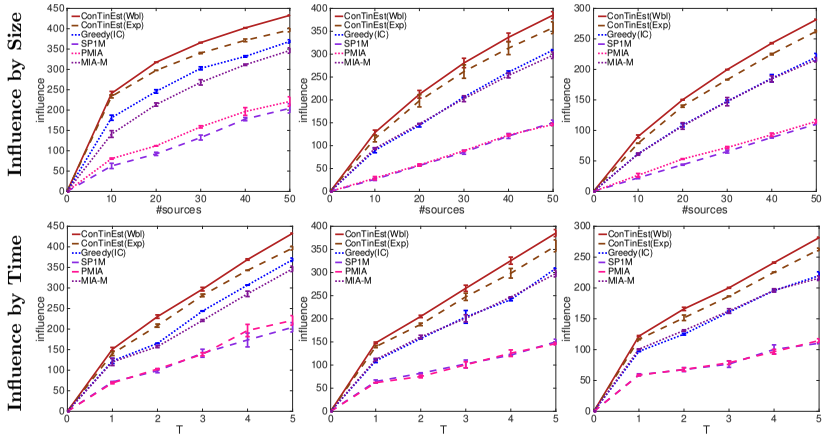
where $\alpha > 0$ is a scale parameter and $\beta > 0$ is a shape parameter. The Weibull distribution (Wbl) has often been used to model lifetime events in survival analysis, providing more flexibility than an exponential distribution. We choose α and β from 0 to 10 uniformly at random for each edge in order to have heterogeneous temporal dynamics. Finally, for each type of Kronecker network, we generate 10 sample networks, each of which has different α and β chosen for every edge.

5.1.1 Influence Estimation

To the best of our knowledge, there is no analytical solution to the influence estimation given Weibull transmission function. Therefore, we compare ConTinEst with the Naive Sampling (NS) approach by considering the highest degree node in a network as the source, and draw 1,000,000 samples for NS to obtain near ground truth. In Figure 3, Column (a) compares ConTinEst with the ground truth provided by NS at different time window T, from 0.1 to 10 in networks of different structures. For ConTinEst, we generate up to 10,000 random samples (or sets of random waiting times), and 5 random labels in the inner loop. In all three networks, estimation provided by ConTinEst fits the ground truth accurately, and the relative error decreases quickly as we increase the number of samples and labels (Column (b) and Column (c)). For 10,000 random samples with 5 random labels, the relative error is smaller than 0.01.

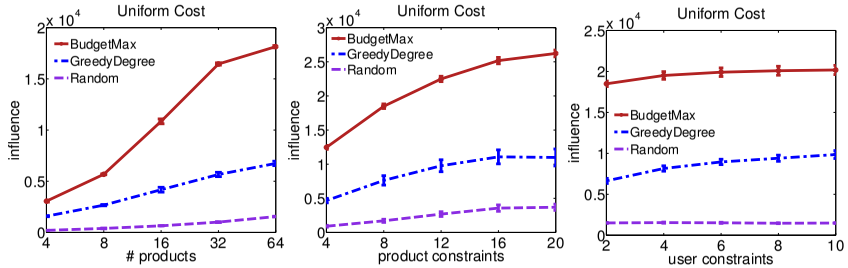
5.1.2 Influence Maximization with Uniform Cost

In this section, we first evaluate the effectiveness of ConTinEst to the classic influence maximization problem where we have only one product to assign with the simple cardinality constraint on the users. We compare to other influence maximization methods developed based on discrete-time diffusion models: traditional greedy by (Kempe et al., 2003), with discrete-time Linear Threshold Model (LT) and Independent Cascade Model (IC) diffusion models, and the heuristic methods SP1M, PMIA and MIA-M by (Chen et al., 2009, 2010a, 2012). For Influmax, since it only supports exponential pairwise transmission functions,



(a) Core-periphery (b) Random (c) Hierarchy

Figure 4: Influence $\sigma(\mathcal{A}, T)$ achieved by varying number of sources $|\mathcal{A}|$ and observation window T on the networks of different structures with 1,024 nodes, 2,048 edges and heterogeneous Weibull transmission functions. Top row: influence against #sources by $T = 5$; Bottom row: influence against the time window T using 50 sources.



(a) By products (b) By product constraints (c) By user constraints

Figure 5: Over the 64 product-specific diffusion networks, each of which has 1,048,576 nodes, the estimated influence (a) for increasing the number of products by fixing the product-constraint at 8 and user-constraint at 2; (b) for increasing product-constraint by user-constraint at 2; and (c) for increasing user-constraint by fixing product-constraint at 8. For all experiments, we have $T = 5$ time window.

we fit an exponential distribution per edge by NetRate (Gomez-Rodriguez et al., 2011). Furthermore, Influmax is not scalable. When the average network density (defined as the average degree per node) of the synthetic networks is ~ 2.0, the run time for Influmax

is more than 24 hours. In consequence, we present the results of ConTinEst using fitted exponential distributions (Exp). For the discrete-time IC model, we learn the infection probability within time window T using Netrapalli's method (Netrapalli and Sanghavi, 2012). The learned pairwise infection probabilities are also served for SP1M and PMIA, which approximately calculate the influence based on the IC model. For the discrete-time LT model, we set the weight of each incoming edge to a node u to the inverse of its in-degree, as in previous work (Kempe et al., 2003), and choose each node's threshold uniformly at random. The top row of Figure 4 compares the expected number of infected nodes against the source set size for different methods. ConTinEst outperforms the rest, and the competitive advantage becomes more dramatic the larger the source set grows. The bottom row of Figure 4 shows the expected number of infected nodes against the time window for 50 selected sources. Again, ConTinEst performs the best for all three types of networks.

Next, using ConTinEst as a subroutine for influence estimation, we evaluate the performance of BudgetMax with the uniform-cost constraints on the users. In our experiments we consider up to 64 products, each of which diffuses over one of the above three different types of Kronecker networks with $\sim$ one million nodes. Further, we randomly select a subset of 512 nodes $\mathcal{V}_S \subseteq \mathcal{V}$ as our candidate target users, who will receive the given products, and evaluate the potential influence of an allocation over the underlying one-million-node networks. For BudgetMax, we set the adaptive threshold δ to 0.01 and the cost per user and product to 1. For ConTinEst, we use 2,048 samples with 5 random labels on each of the product-specific diffusion networks. We repeat our experiments 10 times and report the average performance.

We compare BudgetMax with a nodes' degree-based heuristic, which we refer to as GreedyDegree, where the degree is treated as a natural measure of influence, and a baseline method, which assigns the products to the target nodes randomly. We opt for the nodes' degree-based heuristic since, in practice, large-degree nodes, such as users with millions of followers in Twitter, are often the targeted users who will receive a considerable payment if he (she) agrees to post the adoption of some products (or ads) from merchants. GreedyDegree proceeds as follows. It first sorts the list of all pairs of products i and nodes $j \in \mathcal{V}_S$ in descending order of node-j's degree in the diffusion network associated to product i. Then, starting from the beginning of the list, it considers each pair one by one: if the addition of the current pair to the existing solution does not violate the predefined matroid constraints, it is added to the solution, and otherwise, it is skipped. This process continues until the end of the list is reached. In other words, we greedily assign products to nodes with the largest degree. Due to the large size of the underlying diffusion networks, we do not apply other more expensive node centrality measures such as the clustering coefficient and betweenness.

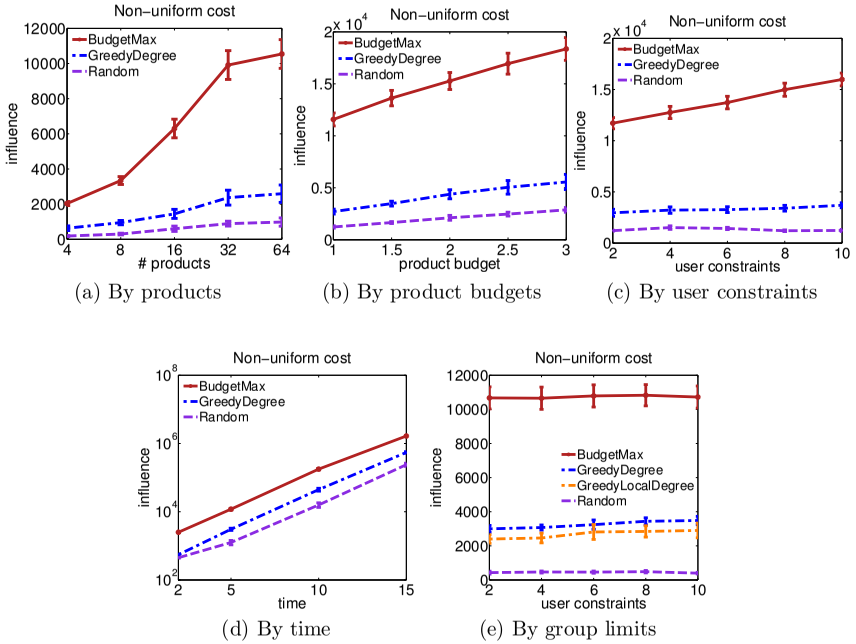
Figure 5 summarizes the results. Panel (a) shows the achieved influence against number of products, fixing the budget per product to 8 and the budget per user to 2. As the number of products increases, on the one hand, more and more nodes become assigned, so the total influence will increase. Yet, on the other hand, the competition among products for a few existing *influential* nodes also increases. GreedyDegree achieves a modest performance, since high degree nodes may have many overlapping children. In contrast, BudgetMax, by taking the submodularity of the problem, the network structure and the diffusion dynamics of the edges into consideration, achieves a superior performance, especially as the number

of product (*i.e.*, the competition) increases. Panel (b) shows the achieved influence against the budget per product, considering 64 products and fixing the budget per user to 2. We find that, as the budget per product increases, the performance of GreedyDegree tends to flatten and the competitive advantage of BudgetMax becomes more dramatic. Finally, Panel (c) shows the achieved influence against the budget per user, considering 64 products and fixing the budget per product to 8. We find that, as the budget per user increases, the influence only increases slowly. This is due to the fixed budget per product, which prevents additional new nodes to be assigned. This meets our intuition: by making a fixed number of people watching more ads per day, we can hardly boost the popularity of the product. Additionally, even though the same node can be assigned to more products, there is hardly ever a node that is the *perfect* source from which all products can efficiently spread.

5.1.3 Influence Maximization with Non-Uniform Cost

In this section, we evaluate the performance of BudgetMax under non-uniform cost constraints, using again ConTinEst as a subroutine for influence estimation. Our designing of user-cost aim to mimic a real scenario, where advertisers pay much more money to celebrities with millions of social network followers than to normal citizens. To do so, we let $c_i \propto d_i^{1/n}$ where c_i is the cost, d_i is the degree, and $n \geq 1$ controls the increasing speed of cost with respect to the degree. In our experiments, we use $n = 3$ and normalize c_i to be within $(0, 1]$. Moreover, we set the product-budget to a base value from 1 to 10 and add a random adjustment drawn from a uniform distribution $U(0, 1)$.

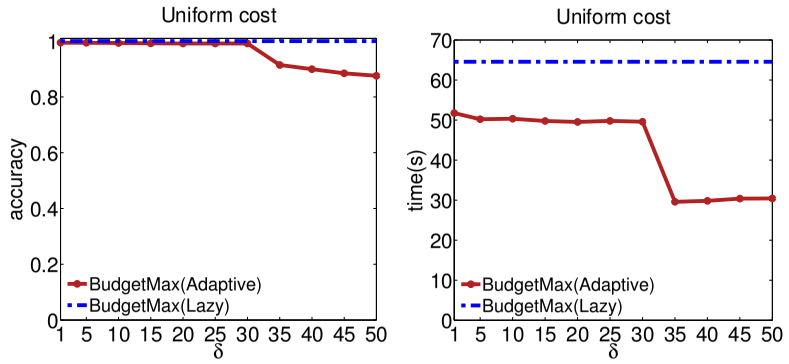
We compare our method to two modified versions of the above mentioned nodes' degree-based heuristic GreedyDegree and to the same baseline method. In the first modified version of the heuristic, which we still refer to as GreedyDegree, takes both the degree and the corresponding cost into consideration. In particular, it sorts the list of all pairs of products i and nodes $j \in \mathcal{V}_S$ in descending order of degree-cost ratio d_j/c_j in the diffusion network associated to product i, instead of simply the node-j's degree, and then proceeds similarly as before. In the second modified version of the heuristic, which we refer as GreedyLocalDegree, we use the same degree-cost ratio but allow the target users to be partitioned into distinct groups (or communities) and pick the most cost-effective pairs within each group locally instead. Figure 6 compares the performance of our method with the competing methods against four factors: (a) the number of products, (b) the budget per product, (c) the budget per user and (d) the time window T, while fixing the other factors. In all cases, BudgetMax significantly outperforms the other methods, and the achieved influence increases monotonically with respect to the factor value, as one may have expected. In addition, in Figure 6(e), we study the effect of the Laminar matroid combined with group knapsack constraints, which is the most general type of constraint we handle in this paper (refer to Section 3.4). The selected target users are further partitioned into K groups randomly, each of which has, $Q_i, i = 1 \dots K$, limit which constrains the maximum allocations allowed in each group. In practical scenarios, each group might correspond to a geographical community or organization. In our experiment, we divide the users into 8 equal-size groups and set $Q_i = 16, i = 1 \dots K$ to indicate that we want a balanced allocation in each group. Figure 6(e) shows the achieved influence against the budget per user for $K = 8$ equally-sized groups and $Q_i = 16, i = 1 \dots K$. In contrast to Figure 6(b), the total



(a) By products (b) By product budgets (c) By user constraints

(d) By time (e) By group limits

Figure 6: Over the 64 product-specific diffusion networks, each of which has a total 1,048,576 nodes, the estimated influence (a) for increasing the number of products by fixing the product-budget at 1.0 and user-constraint at 2; (b) for increasing product-budget by fixing user-constraint at 2; (c) for increasing user-constraint by fixing product-budget at 1.0; (d) for different time window T; and (e) for increasing user-constraint with group-limit 16 by fixing product-budget at 1.0.



(a) δ vs. accuracy (b) δ vs. time

Figure 7: The relative accuracy and the run-time for different threshold parameter δ.

estimated influence does not increase significantly with respect to the budget (*i.e.*, number of slots) per user. This is due to the fixed budget per group, which prevents additional new nodes to be assigned, even though the number of available slots per user increases.

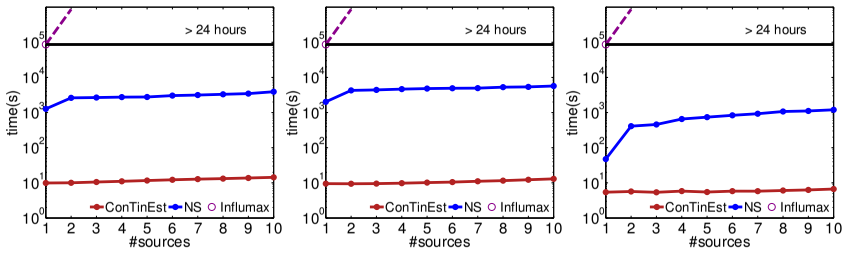
5.1.4 Effects of Adaptive Thresholding

In Figure 7, we investigate the impact that the threshold value δ has on the accuracy and runtime of our adaptive thresholding algorithm and compare it with the lazy evaluation method. Note that the performance and runtime of lazy evaluation do not change with respect to δ because it does not depend on it. Panel (a) shows the achieved influence against the threshold δ. As expected, the larger the δ value, the lower the accuracy. However, our method is relatively robust to the particular choice of δ since its performance is always over a 90-percent relative accuracy even for large δ. Panel (b) shows the runtime against the threshold δ. In this case, the larger the δ value, the lower the runtime. In other words, Figure 7 verifies the intuition that δ is able to trade off the solution quality of the allocation with the runtime time.

5.1.5 Scalability

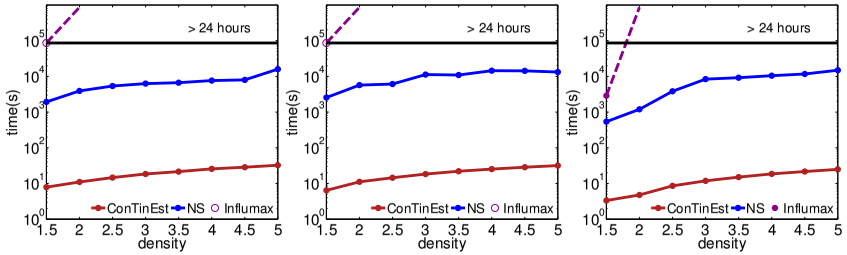
In this section, we start with evaluating the scalability of the proposed algorithms on the classic influence maximization problem where we only have one product with the cardinality constraint on the users.

We compare it to the state-of-the-art method Influmax (Gomez-Rodriguez and Schölkopf, 2012) and the Naive Sampling (NS) method in terms of runtime for the continuous-time influence estimation and maximization. For ConTinEst, we draw 10,000 samples in the outer loop, each having 5 random labels in the inner loop. We plug ConTinEst as a subroutine into the classic greedy algorithm by (Nemhauser et al., 1978). For NS, we also draw 10,000 samples. The first two experiments are carried out in a single 2.4GHz processor.



(a) Core-Periphery (b) Random (c) Hierarchy

Figure 8: Runtime of selecting increasing number of sources on Kronecker networks of 128 nodes and 320 edges with $T = 10$.



(a) Core-Periphery (b) Random (c) Hierarchy

Figure 9: Runtime of selecting 10 sources in networks of 128 nodes with increasing density by $T = 10$.

Figure 8 compares the performance of increasingly selecting sources (from 1 to 10) on small Kronecker networks. When the number of selected sources is 1, different algorithms essentially spend time estimating the influence for each node. ConTinEst outperforms other methods by order of magnitude and for the number of sources larger than 1, it can efficiently reuse computations for estimating influence for individual nodes. Dashed lines mean that a method did not finish in 24 hours, and the estimated run time is plotted.

Next, we compare the run time for selecting 10 sources with increasing densities (or the number of edges) in Figure 9. Again, Influmax and NS are order of magnitude slower due to their respective exponential and quadratic computational complexity in network density. In contrast, the run time of ConTinEst only increases slightly with the increasing density since its computational complexity is linear in the number of edges. We evaluate the speed on large core-periphery networks, ranging from 100 to 1,000,000 nodes with density 1.5 in Figure 10. We report the parallel run time only for ConTinEst and NS (both are implemented by MPI running on 192 cores of 2.4Ghz) since Influmax is not scalable. In contrast to NS, the performance of ConTinEst increases linearly with the network size and can easily scale up to one million nodes.

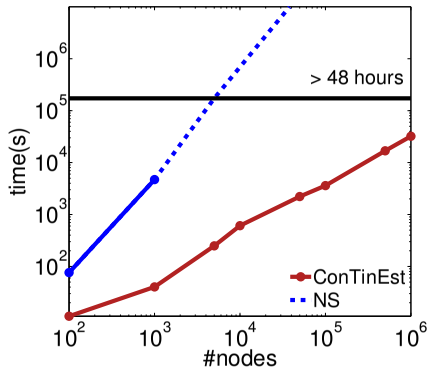
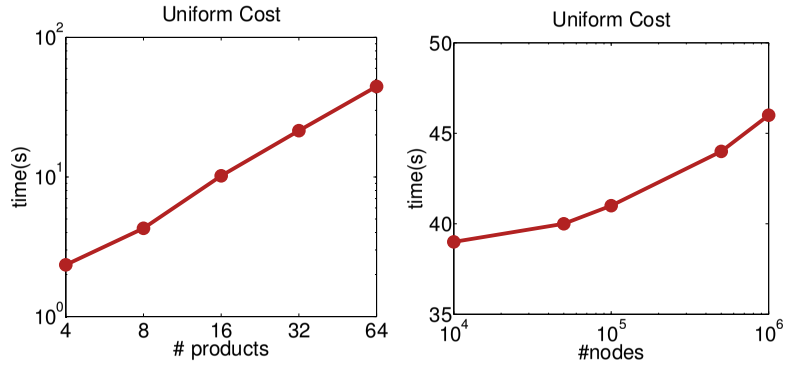


Figure 10: For core-periphery networks by $T = 10$, runtime of selecting 10 sources with increasing network size from 100 to 1,000,000 by fixing 1.5 network density.



(a) Speed by products (b) Speed by nodes

Figure 11: Over the 64 product-specific diffusion networks, each of which has 1,048,576 nodes, the runtime (a) of allocating increasing number of products and (b) of allocating 64 products to 512 users on networks of varying size. or all experiments, we have $T = 5$ time window and fix product-constraint at 8 and user-constraint at 2.

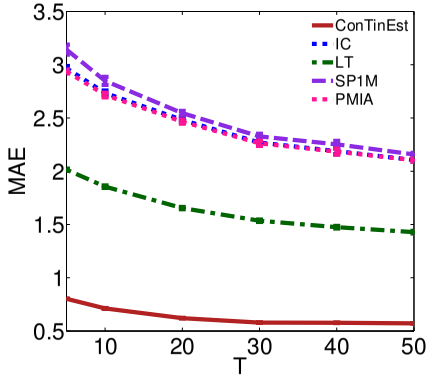


Figure 12: In MemeTracker dataset, comparison of the accuracy of the estimated influence in terms of mean absolute error.

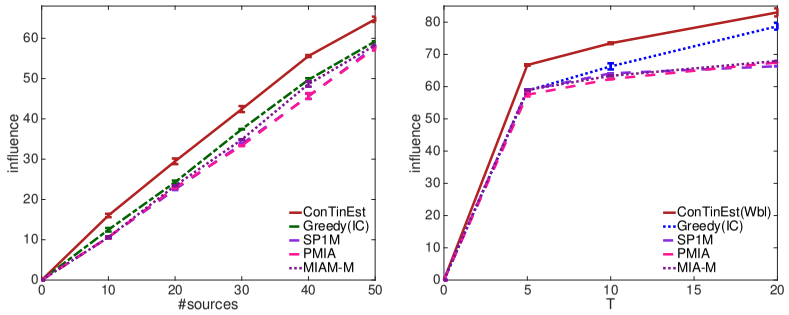
Finally, we investigate the performance of BudgetMax in terms of runtime when using ConTinEst as subroutine to estimate the influence. We can precompute the data structures and store the samples needed to estimate the influence function in advance. Therefore, we focus only on the runtime for the constrained influence maximization algorithm. BudgetMax runs on 64 cores of 2.4Ghz by using OpenMP to accelerate the first round of the optimization. We report the allocation time for increasing number of products in Figure 11(a), which clearly shows a linear time complexity with respect to the size of the ground set. Figure 11(b) evaluates the runtime of allocation by varying the size of the network from 16,384 to 1,048,576 nodes.

5.2 Experiments on Real Data

In this section, we first quantify how well our proposed algorithm can estimate the true influence in the real-world dataset. Then, we evaluate the solution quality of the selected sources for influence maximization under different constraints. We have used the public MemeTracker datasets (Leskovec et al., 2009), which contains more than 172 million news articles and blog posts from 1 million mainstream media sites and blogs.

5.2.1 Influence Estimation

We first trace the flow of information from one site to another by using the hyperlinks among articles and posts as in the work of Gomez-Rodriguez et al. (2011); Du et al. (2012). In detail, we extracted 10,967 hyperlink cascades among top 600 media sites. We then evaluate the accuracy of ConTinEst as follows. First, we repeatedly split all cascades into a 80% training set and a 20% test set at random for five times. On each training set, we learn one continuous-time model, which we use for ConTinEst, and a discrete-time model, which we use for the competitive methods: IC, SP1M, PMIA and MIAM-M. For the continuous-time



(a) Influence vs. #sources (b) Influence vs. time

Figure 13: In MemeTracker dataset, (a) comparison of the influence of the selected nodes by fixing the observation window $T = 5$ and varying the number sources, and (b) comparison of the influence of the selected nodes by fixing the number of sources to 50 and varying the time window.

model, we opt for NetRate (Gomez-Rodriguez et al., 2011) with exponential transmission functions (fixing the shape parameter of the Weibull family to be one) to learn the diffusion networks by maximizing the likelihood of the observed cascades. For the discrete-time model, we learn the infection probabilities using the method by Netrapalli and Sanghavi (2012).

Second, let $\mathcal{C}(u)$ be the set of all cascades where u was the source node. By counting the total number of distinct nodes infected before T in $\mathcal{C}(u)$, we can quantify the real influence of node u up to time T. Thus, we can evaluate the quality of the influence estimation by computing the average (across nodes) Mean Absolute Error (MAE) between the real and the estimated influence on the test set, which we show in Figure 12. Clearly, ConTinEst performs the best statistically. Since the length of real cascades empirically conforms to a power-law distribution, where most cascades are very short (2-4 nodes), the gap of the estimation error is not too large. However, we emphasize that such accuracy improvement is critical for maximizing long-term influence since the estimation error for individuals will accumulate along the spreading paths. Hence, any consistent improvement in influence estimation can lead to significant improvement to the overall influence estimation and maximization task, which is further confirmed in the following sections.

5.2.2 Influence Maximization with Uniform Cost

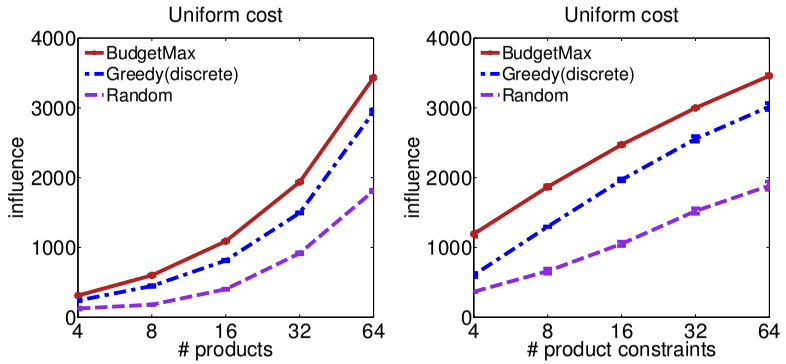
We first apply ConTinEst to the continuous-time influence maximization task with the simple cardinality constraint on the users. We evaluate the influence of the selected nodes in the same spirit as influence estimation: the true influence is calculated as the total number of distinct nodes infected before T based on $\mathcal{C}(u)$ of the selected nodes. Figure 13 shows that the selected sources given by ConTinEst achieve the best performance as we vary the number of selected sources and the observation time window.

Next, we evaluate the performance of BudgetMax on cascades from Memetracker traced from quotes which are short textual phrases spreading through the websites. Because all published documents containing a particular quote are time-stamped, a cascade induced by the same quote is a collection of times when the media site first mentioned it. In detail, we use the public dataset released by Gomez-Rodriguez et al. (2013), which splits the original Memetracker dataset into groups, each associated to a topic or real-world event. Each group consists of cascades built from quotes which were mentioned in posts containing particular keywords. We considered 64 groups, with at least 100,000 cascades, which play the role of products. Therein, we distinguish well-known topics, such as "Apple" and "Occupy Wall-Street", or real-world events, such as the Fukushima nuclear disaster in 2013 and the marriage between Kate Middleton and Prince William in 2011.

We then evaluate the accuracy of BudgetMax in the following way. First, we evenly split each group into a training and a test set and then learn one continuous-time model and a discrete-time model per group using the training sets. As previously, for the continuous-time model, we opt for NetRate (Gomez-Rodriguez et al., 2011) with exponential transmission functions, and for the discrete-time model, we learn the infection probabilities using the method by Netrapalli and Sanghavi (2012), where the step-length is set to one. Second, we run BudgetMax using both the continuous-time model and the discrete-time model. We refer to BudgetMax with the discrete-time model as the Greedy(discrete) method. Since we do have no ground-truth information about cost of each node, we focus our experiments using a uniform cost. Third, once we have found an allocation over the learned networks, we evaluate the performance of the two methods using the cascades in the test set as follows: given a group-node pair (i, j), let $\mathcal{C}(j)$ denote the set of cascades induced by group i that contains node j. Then, we take the average number of nodes coming after j for all the cascades in $\mathcal{C}(j)$ as a proxy of the average influence induced by assigning group i to node j. Finally, the influence of an allocation is just the sum of the average influence of each group-node pair in the solution. In our experiments, we randomly select 128 nodes as our target users.

Figure 14 summarizes the achieved influence against four factors: (a) the number of products, (b) the budget per product, (c) the budget per user and (d) the time window T, while fixing the other factors. In comparison with the Greedy(IC) and a random allocation, BudgetMax finds an allocation that indeed induces the largest diffusion in the test data, with an average 20-percent improvement overall.

In the end, Figure 15 investigates qualitatively the actual allocations of groups (topics or real-world events; in red) and sites (in black). Here, we find examples that intuitively one could expect: "japantoday.com" is assigned to Fukushima Nuclear disaster or "finance.yahoo.com" is assigned to "Occupy Wall-street". Moreover, because we consider several topics and real-world events with different underlying diffusion networks, the selected nodes are not only very popular media sites such as nytimes.com or cnn.com but also several modest sites (Bakshy et al., 2011), often specialized or local, such as freep.com or localnews8.com.



(a) By products (b) By product constraints

(c) By user constraints (d) By time

Figure 14: Over the inferred 64 product-specific diffusion networks, the true influence estimated from separated testing data (a) for increasing the number of products by fixing the product-constraint at 8 and user-constraint at 2; (b) for increasing product-constraint by fixing user-constraint at 2; (c) for increasing user-constraint by fixing product-constraint at 8; (d) for different time window T.

6. Conclusion

We have studied the influence estimation and maximization problems in the continuous-time diffusion model. We first propose a randomized influence estimation algorithm ConTinEst, which can scale up to networks of millions of nodes while significantly improves over previous state of the art methods in terms of the accuracy of the estimated influence. Once we have a subroutine for efficient influence estimation in large networks, we then tackle the problem of maximizing the influence of multiple types of products (or information) in

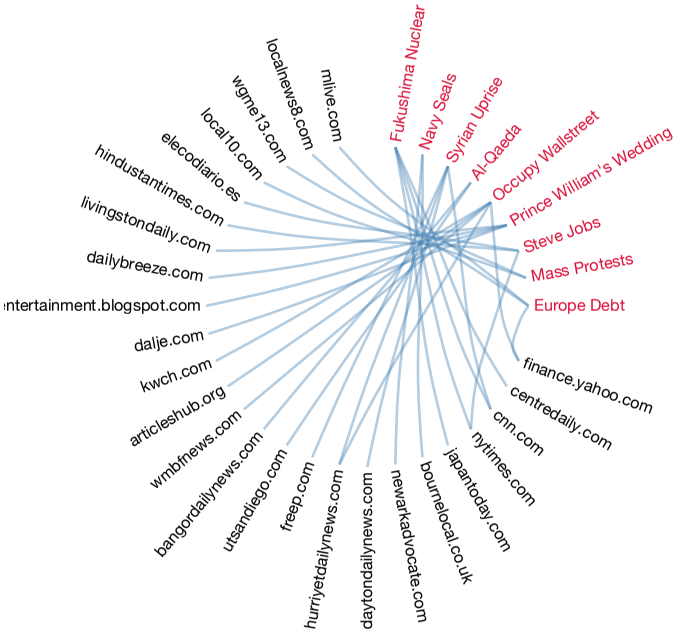


Figure 15: The allocation of memes to media sites.

realistic continuous-time diffusion networks subject to various practical constraints: different products can have different diffusion structures; only influence within a given time window is considered; each user can only be recommended a small number of products; and each product has a limited campaign budget, and assigning it to users incurs costs. We provide a novel formulation as a submodular maximization under an intersection of matroid constraints and group-knapsack constraints, and then design an efficient adaptive threshold greedy algorithm with provable approximation guarantees, which we call BudgetMax. Experimental results show that the proposed algorithm performs remarkably better than other scalable alternatives in both synthetic and real-world datasets. There are also a few interesting open problems. For example, when the influence is estimated using ConTinEst, its error is a random variable. How does this affect the submodularity of the influence function? Is there an influence maximization algorithm that has better tolerance to the random error? These questions are left for future work.

Acknowledgments

Nan Du is supported by the Facebook Graduate Fellowship 2014-2015. Maria-Florina Balcan and Yingyu Liang are supported in part by NSF grant CCF-1101283 and CCF-0953192, AFOSR grant FA9550-09-1-0538, ONR grant N00014-09-1-075, a Microsoft Faculty Fellowship, and a Raytheon Faculty Fellowship. Le Song is supported in part by NSF/NIH BIGDATA 1R01GM108341, ONR N00014-15-1-2340, NSF IIS-1218749, NSF CAREER IIS-1350983, Nvidia and Intel. Hongyuan Zha is supported in part by NSF/NIH BIGDATA 1R01GM108341, NSF DMS-1317424 and NSF CNS-1409635.

References

Odd Aalen, Oernulf Borgan, and Håkon K Gjessing. *Survival and event history analysis: a process point of view*. Springer, 2008.

Ashwinkumar Badanidiyuru and Jan Vondrák. Fast algorithms for maximizing submodular functions. In *SODA*. SIAM, 2014.

Eytan Bakshy, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. Everyone's an influencer: Quantifying influence on twitter. In *International Conference on Web Search and Data Mining (WSDM)*, pages 65–74, 2011.

Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. Influence maximization in social networks: Towards an optimal algorithmic solution. *arXiv preprint arXiv:1212.0884*, 2012.

Wei Chen, Yajun Wang, and Siyu Yang. Efficient influence maximization in social networks. In *Knowledge Discovery and Data Mining (SIGKDD)*, pages 199–208. ACM, 2009.

Wei Chen, Chi Wang, and Yajun Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *Knowledge Discovery and Data Mining (SIGKDD)*, pages 1029–1038. ACM, 2010a.

Wei Chen, Yifei Yuan, and Li Zhang. Scalable influence maximization in social networks under the linear threshold model. In *International Conference on Data Mining series (ICDM)*, pages 88–97. IEEE, 2010b.

Wei Chen, Alex Collins, Rachel Cummings, Te Ke, Zhenming Liu, David Rincon, Xiaorui Sun, Yajun Wang, Wei Wei, and Yifei Yuan. Influence maximization in social networks when negative opinions may emerge and propagate. In *SIAM International Conference on Data Mining (SDM)*, pages 379–390. SIAM, 2011.

Wei Chen, Wei Lu, and Ning Zhang. Time-critical influence maximization in social networks with time-delayed diffusion process. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

Aaron Clauset, Cristopher Moore, and M.E.J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191):98–101, 2008.

Edith Cohen. Size-estimation framework with applications to transitive closure and reachability. *Journal of Computer and System Sciences*, 55(3):441–453, 1997.

Samik Datta, Anirban Majumder, and Nisheeth Shrivastava. Viral marketing for multiple products. In *International Conference on Data Mining series (ICDM)*, pages 118–127, 2010.

Nan Du, Le Song, Alex Smola, and Ming Yuan. Learning networks of heterogeneous influence. In *Neural Information Processing System (NIPS)*, pages 2789–2797, 2012.

Nan Du, Le Song, Manuel Gomez-Rodriguez, and Hongyuhan Zha. Scalable influence estimation in continuous time diffusion networks. In *Neural Information Processing System (NIPS)*, 2013a.

Nan Du, Le Song, Hyenkyun Woo, and Hongyuan Zha. Uncover topic-sensitive information diffusion networks. In *Artificial Intelligence and Statistics (AISTATS)*, 2013b.

David Easley and Jon Kleinberg. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, 2010.

Satoru Fujishige. *Submodular functions and optimization*, volume 58. Elsevier Science Limited, 2005.

Manuel Gomez-Rodriguez and Bernhard Schölkopf. Influence maximization in continuous time diffusion networks. In *International Conference on Machine Learning (ICML)*, pages 313–320, 2012.

Manuel Gomez-Rodriguez, David Balduzzi, and Bernhard Schölkopf. Uncovering the temporal dynamics of diffusion networks. In *International Conference on Machine Learning (ICML)*, 2011.

Manuel Gomez-Rodriguez, Jure Leskovec, and Bernhard Schölkopf. Structure and Dynamics of Information Pathways in On-line Media. In *International Conference on Web Search and Web Data Mining (WSDM)*, 2013.

Amit Goyal, Wei Lu, and Laks V. S. Lakshmanan. Celf++: optimizing the greedy algorithm for influence maximization in social networks. In *World Wide Web Conference (WWW) (Companion Volume)*, pages 47–48, 2011a.

Amit Goyal, Wei Lu, and Laks V. S. Lakshmanan. Simpath: An efficient algorithm for influence maximization under the linear threshold model. In *International Conference on Data Mining series (ICDM)*, pages 211–220, 2011b.

Dino Ienco, Francesco Bonchi, and Carlos Castillo. The meme ranking problem: Maximizing microblogging virality. In *International Conference on Data Mining series (ICDM) Workshops*, 2010.

David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 137–146. ACM, 2003.

Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. Cost-effective outbreak detection in networks. In P. Berkhin, R. Caruana, and X. Wu, editors, *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 420–429. ACM, 2007. URL http://doi.acm.org/10.1145/1281192.1281239.

Jure Leskovec, Lars Backstrom, and Jon Kleinberg. Meme-tracking and the dynamics of the news cycle. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 497–506. ACM, 2009.

Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research (JMLR)*, 11(Feb):985–1042, 2010.

Wei Lu, Francesco Bonchi, Goyal Amit, and Laks V. S. Lakshmanan. The bang for the buck: fair competitive viral marketing from the host perspective. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 928–936, 2013.

Ramasuri Narayanam and Amit A Nanavati. Viral marketing for product cross-sell through social networks. In *Machine Learning and Knowledge Discovery in Databases*. 2012.

George Nemhauser, Laurence Wolsey, and Marshall Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

Praneeth Netrapalli and Sujay Sanghavi. Learning the graph of epidemic cascades. In *SIGMETRICS/PERFORMANCE*, pages 211–222. ACM, 2012. ISBN 978-1-4503-1097-0.

Matthew Richardson and Pedro Domingos. Mining knowledge-sharing sites for viral marketing. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, pages 61–70. ACM, 2002.

Alexander Schrijver. *Combinatorial Optimization*, volume 24 of *Algorithms and Combinatorics*. Springer, 2003.

Tasuku Soma, Naonori Kakimura, Kazuhiro Inaba, and Ken-ichi Kawarabayashi. Optimal budget allocation: Theoretical guarantee and efficient algorithm. In *International Conference on Machine Learning (ICML)*, pages 351–359, 2014.

Tao Sun, Wei Chen, Zhenming Liu, Yajun Wang, Xiaorui Sun, Ming Zhang, and Chin-Yew Lin. Participation maximization based on social influence in online discussion forums. In *International AAAI Conference on Weblogs and Social Media (ICWSM)*, 2011.

Maxim Sviridenko. A note on maximizing a submodular set function subject to knapsack constraint. *Operations Research Letters*, 32:41–43, 2004.

Appendix A. Naive Sampling Algorithm

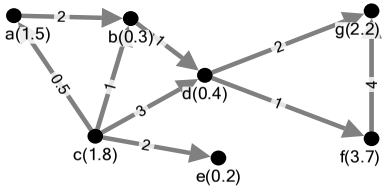
The graphical model perspective described in Section 2.2 suggests a naive sampling (NS) algorithm for approximating $\sigma(\mathcal{A}, T)$:

1. Draw n samples, $\left\{\left\{\tau_{ji}^l\right\}_{(j,i)\in\mathcal{E}}\right\}_{l=1}^n$, *i.i.d.* from the waiting time product distribution $\prod_{(j,i)\in\mathcal{E}} f_{ji}(\tau_{ji})$;
2. For each sample $\left\{\tau_{ji}^l\right\}_{(j,i)\in\mathcal{E}}$ and for each node i, find the shortest path from source nodes to node i; count the number of nodes with $g_i\left(\left\{\tau_{ji}^l\right\}_{(j,i)\in\mathcal{E}}\right) \leq T$;
3. Average the counts across n samples.

Although the naive sampling algorithm can handle arbitrary transmission function, it is not scalable to networks with millions of nodes. We need to compute the shortest path for each node and each sample, which results in a computational complexity of $O(n|\mathcal{E}| + n|\mathcal{V}|\log|\mathcal{V}|)$ for a single source node. The problem is even more pressing in the influence maximization problem, where we need to estimate the influence of source nodes at different location and with increasing number of source nodes. To do this, the algorithm needs to be repeated, adding a multiplicative factor of $C|\mathcal{V}|$ to the computational complexity (C is the number of nodes to select). Then, the algorithm becomes quadratic in the network size. When the network size is in the order of thousands and millions, typical in modern social network analysis, the naive sampling algorithm become prohibitively expensive. Additionally, we may need to draw thousands of samples (n is large), further making the algorithm impractical for large-scale problems.

Appendix B. Least Label List

The notation "argsort$((r_1, \ldots, r_{|\mathcal{V}|}), \text{ascend})$" in line 2 of Algorithm 3 means that we sort the collection of random labels in ascending order and return the argument of the sort as an ordered list.



- Node labeling : $e(0.2) < b(0.3) < d(0.4) < a(1.5) < c(1.8) < g(2.2) < f(3.7)$
- Neighborhoods: $\mathcal{N}(c, 2) = \{a, b, c, e\}$; $\mathcal{N}(c, 3) = \{a, b, c, d, e, f\}$;
- Least-label list: $r_*(c) : (2, 0.2), (1, 0.3), (0.5, 1.5), (0, 1.8)$
- Query: $r_*(c, 0.8) = r(a) = 1.5$

Figure 16: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, edge weights $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$, and node labeling $\{r_i\}_{i\in\mathcal{V}}$ with the associated output from Algorithm 3.

Algorithm 3: Least Label List

Input: a reversed directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with edge weights $\{\tau_{ji}\}_{(j,i)\in\mathcal{E}}$, a node labeling $\{r_i\}_{i\in\mathcal{V}}$

Output: A list $r_*(s)$ for each $s \in \mathcal{V}$

```
for each s ∈ V do d_s ← ∞, r_*(s) ← ∅;
for i in argsort((r_1, ..., r_|V|), ascend) do
    empty heap H ← ∅;
    set all nodes except i as unvisited;
    push (0, i) into heap H;
    while H ≠ ∅ do
        pop (d_*, s) with the minimum d_* from H;
        add (d_*, r_i) to the end of list r_*(s);
        d_s ← d*;
        for each unvisited out-neighbor j of s do
            set j as visited;
            if (d, j) in heap H then
                Pop (d, j) from heap H;
                Push (min {d, d_* + τ_js}, j) into heap H;
            else if d_* + τ_js < d_j then
                Push (d_* + τ_js, j) into heap H;
```

Figure 16 shows an example of the Least-Label-List. The nodes from a to g are assigned to exponentially distributed labels with mean one shown in each parentheses. Given a query distance 0.8 for node c, we can binary-search its Least-label-list $r_*(c)$ to find that node a belongs to this range with the smallest label $r(a) = 1.5$.

Appendix C. Theorem 1

Theorem 1 *Sample the following number of sets of random transmission times*

$$n \geqslant \frac{C\Lambda}{\epsilon^2}\log\left(\frac{2|\mathcal{V}|}{\alpha}\right)$$

where $\Lambda := \max_{\mathcal{A}:|\mathcal{A}|\leq C} 2\sigma(\mathcal{A},T)^2/(m-2) + 2Var(|\mathcal{N}(\mathcal{A},T)|)(m-1)/(m-2) + 2a\epsilon/3$, $|\mathcal{N}(\mathcal{A},T)| \leqslant a$, *and for each set of random transmission times, sample* m *set of random labels. Then we can guarantee that*

$$|\widehat{\sigma}(\mathcal{A},T) - \sigma(\mathcal{A},T)| \leqslant \epsilon$$

simultaneously for all $\mathcal{A}$ *with* $|\mathcal{A}| \leqslant C$, *with probability at least* $1-\alpha$.

Proof Let $S_\tau := |\mathcal{N}(\mathcal{A},T)|$ for a fixed set of $\{\tau_{ji}\}$ and then $\sigma(\mathcal{A},T) = \mathbb{E}_\tau[S_\tau]$. The randomized algorithm with m randomizations produces an unbiased estimator $\widehat{S}_\tau = (m-1)/(\sum_{u=1}^m r_*^u)$ for S_τ, *i.e.*, $\mathbb{E}_{r|\tau}[\widehat{S}_\tau] = S_\tau$, with variance $\mathbb{E}_{r|\tau}[(\widehat{S}_\tau - S_\tau)^2] = S_\tau^2/(m-2)$ (Cohen, 1997).

Then $\widehat{S}_\tau$ is also an unbiased estimator for $\sigma(\mathcal{A},T)$, since $\mathbb{E}_{\tau,r}[\widehat{S}_\tau] = \mathbb{E}_\tau\mathbb{E}_{r|\tau}[\widehat{S}_\tau] = \mathbb{E}_\tau[S_\tau] = \sigma(\mathcal{A},T)$. Its variance is

$$\begin{aligned}
Var(\widehat{S}_\tau) &:= \mathbb{E}_{\tau,r}[(\widehat{S}_\tau - \sigma(\mathcal{A},T))^2] = \mathbb{E}_{\tau,r}[(\widehat{S}_\tau - S_\tau + S_\tau - \sigma(\mathcal{A},T))^2] \\
&= \mathbb{E}_{\tau,r}[(\widehat{S}_\tau - S_\tau)^2] + 2\,\mathbb{E}_{\tau,r}[(\widehat{S}_\tau - S_\tau)(S_\tau - \sigma(\mathcal{A},T))] + \mathbb{E}_{\tau,r}[(S_\tau - \sigma(\mathcal{A},T))^2] \\
&= \mathbb{E}_\tau[S_\tau^2/(m-2)] + 0 + Var(S_\tau) \\
&= \sigma(\mathcal{A},T)^2/(m-2) + Var(S_\tau)(m-1)/(m-2)
\end{aligned}$$

Then using Bernstein's inequality, we have, for our final estimator $\widehat{\sigma}(\mathcal{A},T) = \frac{1}{n}\sum_{l=1}^n \widehat{S}_{\tau^l}$, that

$$\Pr\{|\widehat{\sigma}(\mathcal{A},T) - \sigma(\mathcal{A},T)| \geqslant \epsilon\} \leqslant 2\exp\left(-\frac{n\epsilon^2}{2Var(\widehat{S}_\tau) + 2a\epsilon/3}\right) \tag{21}$$

where $\widehat{S}_\tau < a \leqslant |\mathcal{V}|$.

Setting the right hand side of relation (21) to α, we have that, with probability $1-\alpha$, sampling the following number sets of random transmission times

$$\begin{aligned}
n &\geqslant \frac{2Var(\widehat{S}_\tau) + 2a\epsilon/3}{\epsilon^2}\log\left(\frac{2}{\alpha}\right) \\
&= \frac{2\sigma(\mathcal{A},T)^2/(m-2) + 2Var(S_\tau)(m-1)/(m-2) + 2a\epsilon/3}{\epsilon^2}\log\left(\frac{2}{\alpha}\right)
\end{aligned}$$

we can guarantee that our estimator to have error $|\widehat{\sigma}(\mathcal{A},T) - \sigma(\mathcal{A},T)| \leqslant \epsilon$.

If we want to insure that $|\widehat{\sigma}(\mathcal{A},T) - \sigma(\mathcal{A},T)| \leqslant \epsilon$ simultaneously hold for all $\mathcal{A}$ such that $|\mathcal{A}| \leqslant C \ll |\mathcal{V}|$, we can first use union bound with relation (21). In this case, we have that, with probability $1-\alpha$, sampling the following number sets of random transmission times

$$n \geqslant \frac{C\Lambda}{\epsilon^2}\log\left(\frac{2|\mathcal{V}|}{\alpha}\right)$$

we can guarantee that our estimator to have error $|\widehat{\sigma}(\mathcal{A},T) - \sigma(\mathcal{A},T)| \leqslant \epsilon$ for all $\mathcal{A}$ with $|\mathcal{A}| \leqslant C$. Note that we have define the constant $\Lambda := \max_{\mathcal{A}:|\mathcal{A}|\leq C} 2\sigma(\mathcal{A},T)^2/(m-2) + 2Var(S_\tau)(m-1)/(m-2) + 2a\epsilon/3$. ■

Appendix D. Complete Proofs for Section 4

D.1 Uniform Cost

In this section, we first prove a theorem for the general problem defined by Equation (19), considering a normalized monotonic submodular function $f(S)$ and general P (Theorem 7) and $k = 0$, and then obtain the guarantee for our influence maximization problem (Theorem 4).

Suppose $G = \{g_1, \ldots, g_{|G|}\}$ in the order of selection, and let $G^t = \{g_1, \ldots, g_t\}$. Let C_t denote all those elements in $O \setminus G$ that satisfy the following: they are still feasible before selecting the t-th element g_t but are infeasible after selecting g_t. Formally,

$$C_t = \left\{ z \in O \setminus G : \{z\} \cup G^{t-1} \in \mathcal{F}, \{z\} \cup G^t \notin \mathcal{F} \right\}.$$

In the following, we will prove three claims and then use them to prove the Theorems 7 and 4. Recall that for any $i \in \mathcal{Z}$ and $S \subseteq \mathcal{Z}$, the marginal gain of z with respect to S is denoted as

$$f(z|S) := f(S \cup \{z\}) - f(S)$$

and its approximation is denoted by

$$\widehat{f}(z|S) = \widehat{f}(S \cup \{z\}) - \widehat{f}(S).$$

Also, when $|f(S) - \widehat{f}(S)| \leq \epsilon$ for any $S \subseteq \mathcal{Z}$, we have

$$|\widehat{f}(z|S) - f(z|S)| \leq 2\epsilon$$

for any $z \in \mathcal{Z}$ and $S \subseteq \mathcal{Z}$.

Claim 1. *$\sum_{i=1}^{t} |C_i| \leq Pt$, for $t = 1, \ldots, |G|$.*

Proof We first show the following property about matroids: for any $Q \subseteq \mathcal{Z}$, the sizes of any two maximal independent subsets T_1 and T_2 of Q can only differ by a multiplicative factor at most P. Here, T is a maximal independent subset of Q if and only if:

- $T \subseteq Q$;
- $T \in \mathcal{F} = \bigcap_{i=1}^{P} \mathcal{I}_p$;
- $T \cup \{z\} \notin \mathcal{F}$ for any $z \in Q \setminus T$.

To prove the property, note that for any element $z \in T_1 \setminus T_2$, $\{z\} \cup T_2$ violates at least one of the matroid constraints since T_2 is maximal. Let $\{V_i\}_{1 \leq i \leq P}$ denote all elements in $T_1 \setminus T_2$ that violate the i-th matroid, and then partition $T_1 \cap T_2$ using these V_i's so that they cover T_1. Note that the size of each V_i must be at most that of T_2, since otherwise by the Exchange axiom, there would exist $z \in V_i \setminus T_2$ that can be added to T_2 without violating the i-th matroid, leading to a contradiction. Therefore, $|T_1|$ is at most P times $|T_2|$.

Now we apply the property to prove the claim. Let Q be the union of G^t and $\bigcup_{i=1}^{t} C_t$. On one hand, G^t is a maximal independent subset of Q, since no element in $\bigcup_{i=1}^{t} C_t$ can be added to G^t without violating the matroid constraints. On the other hand, $\bigcup_{i=1}^{t} C_t$ is an independent subset of Q, since it is part of the optimal solution. Therefore, $\bigcup_{i=1}^{t} C_t$ has size at most P times $|G^t|$, which is Pt. ■

Claim 2. *Suppose g_t is selected at the threshold τ_t. Then, $f(j|G^{t-1}) \leq (1+\delta)\tau_t + 4\epsilon + \frac{\delta}{N} f(G), \forall j \in C_t$.*

Proof First, consider $\tau_t > w_{L+1} = 0$. Since g_t is selected at the threshold τ_t, we have that $\widehat{f}(g_t|G^{t-1}) \geq \tau_t$ and thus $f(g_t|G^{t-1}) \geq \tau_t - 2\epsilon$. Any $j \in C_t$ could have been selected at an earlier stage, since adding j to G^{t-1} would not have violated the constraint. However, since $j \notin G^{t-1}$, that means that $\widehat{f}(j|G^{t-1}) \leq (1+\delta)\tau_t$. Then,

$$f(j|G^{t-1}) \leq (1+\delta)\tau_t + 2\epsilon.$$

Second, consider $\tau_t = w_{L+1} = 0$. For each $j \in C_t$, we have $\widehat{f}(j|G) < \frac{\delta}{N} d$. Since the greedy algorithm must pick g_1 with $\widehat{f}(g_1) = d$ and $d \leq f(g_1) + \epsilon$, then

$$f(j|G) < \frac{\delta}{N} f(G) + 4\epsilon.$$

The claim follows by combining the two cases. ■

Claim 3. *The marginal gain of $O \setminus G$ satisfies*

$$\sum_{j \in O \setminus G} f(j|G) \leq [(1+\delta)P + \delta] f(G) + (6 + 2\delta)\epsilon P |G|.$$

Proof Combining Claim 1 and Claim 2, we have:

$$\begin{aligned}\sum_{j \in O \setminus G} f(j|G) = \sum_{t=1}^{|G|} \sum_{j \in C_t} f(j|G) &\leq (1+\delta) \sum_{t=1}^{|G|} |C_t|\tau_t + \delta f(G) + 4\epsilon \sum_{t=1}^{|G|} |C_t| \\ &\leq (1+\delta) \sum_{t=1}^{|G|} |C_t|\tau_t + \delta f(G) + 4\epsilon P |G|.\end{aligned}$$

Further, $\sum_{t=1}^{|G|} |C_t|\tau_t \leq P \sum_{t=1}^{|G|} \tau_t$ by Claim 1 and a technical lemma (Lemma 6). Finally, the claim follows from the fact that $f(G) = \sum_t f(g_t|G^{t-1}) \geq \sum_t (\tau_t - 2\epsilon)$. ■

Lemma 6 *If $\sum_{i=1}^{t} \sigma_{i-1} \leq t$ for $t = 1, \ldots, K$ and $\rho_{i-1} \geq \rho_i$ for $i = 1, \ldots, K-1$ with $\rho_i, \sigma_i \geq 0$, then $\sum_{i=1}^{K} \rho_i \sigma_i \leq \sum_{i=1}^{K} \rho_{i-1}$.*

Proof Consider the linear program

$$\begin{aligned} V \quad = \quad & \max_{\sigma} \sum_{i=1}^{K} \rho_i \sigma_i \\ & \text{s.t.} \quad \sum_{i=1}^{t} \sigma_{i-1} \leq t, \quad t = 1, \ldots, K, \\ & \qquad \sigma_i \geq 0, \quad i = 1, \ldots, K-1 \end{aligned}$$

with dual

$$\begin{aligned} W \quad = \quad & \min_{u} \sum_{i=1}^{K} t u_{t-1} \\ & \text{s.t.} \quad \sum_{t=i}^{K-1} u_t \geq \rho_i, \quad i = 0, \ldots, K-1, \\ & \qquad u_t \geq 0, \quad t = 0, \ldots, K-1. \end{aligned}$$

As $\rho_i \geq \rho_{i+1}$, the solution $u_i = \rho_i - \rho_{i+1}, i = 0, \ldots, K-1$ (where $\rho_K = 0$) is dual feasible with value $\sum_{t=1}^{K} t(\rho_{t-1} - \rho_t) = \sum_{i=1}^{K} \rho_{i-1}$. By weak linear programming duality, $\sum_{i=1}^{K} \rho_i \sigma_i \leq V \leq W \leq \sum_{i=1}^{K} \rho_{i-1}$. ■

Theorem 7 *Suppose we use Algorithm 2 to solve the problem defined by Equation (19) with $k = 0$, using $\rho = 0$ and $\widehat{f}$ to estimate the function f, where $|\widehat{f}(S) - f(S)| \leq \epsilon$ for all $S \subseteq \mathcal{Z}$. It holds that the algorithm returns a greedy solution G with*

$$f(G) \geq \frac{1}{(1+2\delta)(P+1)} f(O) - \frac{4P|G|}{P + c_f}\epsilon$$

where O is the optimal solution, using $\mathcal{O}(\frac{N}{\delta}\log\frac{N}{\delta})$ evaluations of $\widehat{f}$.

Proof By submodularity and Claim 3, we have:

$$f(O) \leq f(O \cup G) \leq f(G) + \sum_{j \in O \setminus G} f(j|G) \leq (1+\delta)(P+1)f(G) + (6+2\delta)\epsilon P|G|,$$

which leads to the bound in the theorem.

Since there are $\mathcal{O}(\frac{1}{\delta}\log\frac{N}{\delta})$ thresholds, and there are $\mathcal{O}(N)$ evaluations at each threshold, the number of evaluations is bounded by $\mathcal{O}(\frac{N}{\delta}\log\frac{N}{\delta})$. ■

Theorem 7 essentially shows $f(G)$ is close to $f(O)$ up to a factor roughly $(1+P)$, which then leads to the following guarantee for our influence maximization problem. Suppose product $i \in \mathcal{L}$ spreads according to diffusion network $\mathcal{G}_i = (\mathcal{V}, \mathcal{E}_i)$, and let $i^* = \text{argmax}_{i \in \mathcal{L}} |\mathcal{E}_i|$.

Theorem 4. *In the influence maximization problem with uniform cost, Algorithm 2 (with $\rho = 0$) is able to output a solution G that satisfies $f(G) \geq \frac{1-2\delta}{3} f(O)$ in expected time $\widetilde{\mathcal{O}}\left(\frac{|\mathcal{E}_{i^*}|+|\mathcal{V}|}{\delta^2} + \frac{|\mathcal{L}||\mathcal{V}|}{\delta^3}\right)$.*

Proof In the influence maximization problem, the number of matroids is $P = 2$. Also note that $|G| \leq f(G) \leq f(O)$, which leads to $4|G|\epsilon \leq 4\epsilon f(O)$. The approximation guarantee then follows from setting $\epsilon \leq \delta/16$ when using ConTinEst (Du et al., 2013a) to estimate the influence.

The runtime is bounded as follows. In Algorithm 2, we need to estimate the marginal gain of adding one more product to the current solution. In ConTinEst (Du et al., 2013a), building the initial data structure takes time

$$\mathcal{O}\left((|\mathcal{E}_{i^*}|\log|\mathcal{V}| + |\mathcal{V}|\log^2|\mathcal{V}|)\frac{1}{\delta^2}\log\frac{|\mathcal{V}|}{\delta}\right)$$

and afterwards each function evaluation takes time

$$\mathcal{O}\left(\frac{1}{\delta^2}\log\frac{|\mathcal{V}|}{\delta}\log\log|\mathcal{V}|\right).$$

As there are $\mathcal{O}\left(\frac{N}{\delta}\log\frac{N}{\delta}\right)$ evaluations where $N = |\mathcal{L}||\mathcal{V}|$, the runtime of our algorithm follows. ■

D.2 General case

As in the previous section, we first prove that a theorem for the problem defined by Equation (19) with general normalized monotonic submodular function $f(S)$ and general P (Theorem 8), and then obtain the guarantee for our influence maximization problem (Theorem 5).

Theorem 8 *Suppose Algorithm 1 uses $\widehat{f}$ to estimate the function f which satisfies $|\widehat{f}(S) - f(S)| \leq \epsilon$ for all $S \subseteq \mathcal{Z}$. Then, there exists a ρ such that*

$$f(S_\rho) \geq \frac{\max\{1, |A_\rho|\}}{(P+2k+1)(1+2\delta)} f(O) - 8\epsilon|S_\rho|$$

where A_ρ is the set of active knapsack constraints:

$$A_\rho = \{i : S_\rho \cup \{z\} \notin \mathcal{F}, \forall z \in \mathcal{Z}_{i*}\}.$$

Proof Consider the optimal solution O and set $\rho^* = \frac{2}{P+2k+1} f(O)$. By submodularity, we have $d \leq f(O) \leq |\mathcal{Z}|d$, so $\rho \in \left[\frac{2d}{P+2k+1}, \frac{2|\mathcal{Z}|d}{P+2k+1}\right]$, and there is a run of Algorithm 2 with ρ such that $\rho^* \in [\rho, (1+\delta)\rho]$. In the following we consider this run.

Case 1 Suppose $|A_\rho| = 0$. The key observation in this case is that since no knapsack constraints are active, the algorithm runs as if there were only matroid constraints. Then, the argument for matroid constraints can be applied. More precisely, let

$$O_+ := \{z \in O \setminus S_\rho : f(z|S_\rho) \geq c(z)\rho + 2\epsilon\}$$

$$O_- := \{z \in O \setminus S_\rho : z \notin O_+\}.$$

Note that all elements in O_+ are feasible. Following the argument of Claim 3 in Theorem 7, we have:

$$f(O_+|S_\rho) \leq ((1+\delta)P + \delta) f(S_\rho) + (4+2\delta)\epsilon P|S_\rho|. \tag{22}$$

Also, by definition, the marginal gain of O_- is:

$$f(O_-|S_\rho) \leq k\rho + 2\epsilon|O_-| \leq k\rho + 2\epsilon P|S_\rho|, \tag{23}$$

where the last inequality follows from the fact that S_ρ is a maximal independent subset, O_- is an independent subset of $O \cup S_\rho$, and the sizes of any two maximal independent subsets in the intersection of P matroids can differ by a factor of at most P. Plugging (22)(23) into $f(O) \leq f(O_+|S_\rho) + f(O_-|S_\rho) + f(S_\rho)$, we obtain the bound

$$f(S_\rho) \geq \frac{f(O)}{(P+2k+1)(1+\delta)} - \frac{(6+2\delta)\epsilon P|S_\rho|}{(P+1)(1+\delta)}.$$

Case 2 Suppose $|A_\rho| > 0$. For any $i \in A_\rho$ (*i.e.*, the i-th knapsack constraint is active), consider the step when i is added to A_ρ. Let $G_i = G \cap \mathcal{Z}_{i*}$, and we have $c(G_i) + c(z) > 1$.

Since every element g we include in G_i satisfies $\widehat{f}(g|G) \geq c(g)\rho$ with respect to the solution G_i when g is added. Then $f(g|G) = f_i(g|G_i) \geq c(g)\rho - 2\epsilon$, and we have:

$$f_i(G_i \cup \{z\}) \geq \rho[c(G_i) + c(z)] - 2\epsilon(|G_i| + 1) > \rho - 2\epsilon(|G_i| + 1). \tag{24}$$

Note that G_i is non-empty since otherwise the knapsack constraint will not be active. Any element in G_i is selected before or at w_t, so $f_i(G_i) \geq w_t - 2\epsilon$. Also, note that z is not selected in previous thresholds before w_t, so $f_i(\{z\}\,|G_i) \leq (1+\delta)w_t + 2\epsilon$ and thus,

$$f_i(\{z\}\,|G_i) \leq (1+\delta)f_i(G_i) + 2\epsilon(2+\delta). \tag{25}$$

Combining Eqs. 24 and 25 into $f_i(G_i \cup \{z\}) = f_i(G_i) + f_i(\{z\}\,|G_i)$ leads to

$$\begin{aligned} f_i(G_i) \geq \frac{\rho}{(2+\delta)} - \frac{2\epsilon(|G_i|+3+\delta)}{(2+\delta)} &\geq \frac{1}{2(1+2\delta)}\rho^* - \frac{2\epsilon(|G_i|+3+\delta)}{(2+\delta)} \\ &\geq \frac{f(O)}{(P+2k+1)(1+2\delta)} - 5\epsilon|G_i|. \end{aligned}$$

Summing up over all $i \in A_\rho$ leads to the desired bound. ■

Suppose item $i \in \mathcal{L}$ spreads according to the diffusion network $\mathcal{G}_i = (\mathcal{V}, \mathcal{E}_i)$. Let $i^* = \text{argmax}_{i\in\mathcal{L}} |\mathcal{E}_i|$. By setting $\epsilon = \delta/16$ in Theorem 8, we have:

Theorem 5. *In Algorithm 1, there exists a ρ such that*

$$f(S_\rho) \geq \frac{\max\{k_a, 1\}}{(2|\mathcal{L}|+2)(1+3\delta)} f(O)$$

where k_a is the number of active knapsack constraints. The expected runtime to obtain the solution is $\widetilde{\mathcal{O}}\left(\frac{|\mathcal{E}_{i^*}|+|\mathcal{V}|}{\delta^2} + \frac{|\mathcal{L}||\mathcal{V}|}{\delta^4}\right)$.

Journal of Machine Learning Research 18 (2017) 1-35 Submitted 3/15; Revised 8/16; Published 1/17

Local algorithms for interactive clustering

Pranjal Awasthi PRANJAL.AWASTHI@RUTGERS.EDU
Department of Computer Science
Rutgers University

Maria Florina Balcan NINAMF@CS.CMU.EDU
School of Computer Science
Carnegie Mellon University

Konstantin Voevodski KVODSKI@GOOGLE.COM
Google, NY, USA

Editor: Le Song

Abstract

We study the design of interactive clustering algorithms. The user supervision that we consider is in the form of cluster split/merge requests; such feedback is easy for users to provide because it only requires a high-level understanding of the clusters. Our algorithms start with any initial clustering and only make local changes in each step; both are desirable properties in many applications. Local changes are desirable because in practice edits of other parts of the clustering are considered churn - changes that are perceived as quality-neutral or quality-negative. We show that in this framework we can still design provably correct algorithms given that our data satisfies natural separability properties. We also show that our framework works well in practice.

1. Introduction

Clustering is usually studied in an unsupervised learning scenario where the goal is to partition the data given pairwise similarity information. Designing provably-good clustering algorithms is challenging because given a similarity function there may be many possible clusterings of the data. Traditional approaches resolve this ambiguity by making assumptions on the data-generation process. For example, there is a large body of work on clustering data that is generated by a mixture of Gaussians (Achlioptas and McSherry, 2005; Kannan et al., 2005; Dasgupta, 1999; Arora and Kannan, 2001; Brubaker and Vempala, 2008; Kalai et al., 2010; Moitra and Valiant, 2010; Belkin and Sinha, 2010), and finding clusters with certain density (Rinaldo and Wasserman, 2010; Chaudhuri and Dasgupta, 2010). But instead of making such assumptions and trying to set the corresponding hyperparameters, we can use limited user (expert) supervision to help the algorithm reach the correct answer.

Interactive clustering algorithms have been facilitated by the availability of cheap crowd-sourcing tools in recent years, which enable collection of relevant user input. In certain applications such as search and document classification, where users are willing to help a clustering algorithm arrive at their own desired answer with a small amount of additional feedback, interactive clustering algorithms are very useful. The works of Balcan and Blum (2008) and Awasthi and Zadeh (2010) provide some initial theoretical results in this new and exciting research area, but their models are not very practical.

©2017 Pranjal Awasthi, Maria Florina Balcan, Konstantin Voevodski.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-085.html.

We observe that in many practical settings we already start with a fairly good clustering computed with semi-automated techniques. For example, consider a massive online news portal that maintains a large collection of news articles. Suppose that the news articles are clustered on the "back-end," and are used to serve several "front-end" applications such as recommendations and article profiles. For such a system, we do not have the freedom to compute arbitrary clusterings and present them to the user, which has been suggested in prior interactive clustering work (Balcan and Blum, 2008; Awasthi and Zadeh, 2010). But it is still feasible to get specific feedback on the current proposed clustering and *locally* edit this clustering. In particular, we may only want to change the "bad" part of the clustering that is revealed by the feedback without changing the rest of the clustering. Our intuition for only considering *local* edits is that in practice changes to other parts of the clustering are considered *churn* - changes that are perceived as quality-neutral or quality-negative. This observation is especially true for large clustering systems that serve many users, which operate on the assumption that if some of the data is broken, it will be pointed out by a user. For such applications it is undesirable to change what the user is not complaining about. Motivated by these observations, in this paper we study the problem of designing *local* algorithms for interactive clustering.

We propose a theoretical interactive clustering model and provide strong experimental evidence supporting its utility in practice. In our model we start with some initial clustering of the data. The algorithm then interacts with the user in stages. In each stage the user provides limited feedback on the proposed clustering in the form of *split* and *merge* requests. The algorithm then makes a *local* edit to the clustering that is consistent with user feedback. Such edits are aimed at improving the problematic part of the clustering pointed out by the user. The goal of the algorithm is to quickly converge (using as few requests as possible) to a clustering that the user is happy with - we call this clustering the target (ground truth) clustering.

In our model the user may request a certain cluster to be *split* if it is *overclustered* (intersects two or more clusters in the target clustering). The user may also request to *merge* two given clusters if they are *underclustered* (both intersect the same target cluster). Note that the user does not tell the algorithm how to perform the split or the merge; such input is unrealistic because it requires a manual analysis of the data points in the corresponding clusters. We also restrict the algorithm to only make *local* changes at each step - in response we may change only the cluster assignments of the points in the corresponding clusters. If the user requests to split a cluster C_i, we may change only the cluster assignments of the points in C_i, and if the user requests to merge C_i and C_j, we may only reassign the points in C_i and C_j.

The split and merge requests in our model are a natural form of feedback. It is easy for users to spot over/underclustering errors and request the corresponding splits/merges (without having to provide any additional information about how to perform the edit). For our model to be practically applicable, we also need to account for noise in the user requests. In particular, if the user requests a merge, only a fraction or a constant number of the points in the two clusters may belong to the same target cluster. Our model (see Section 2) allows for such noisy user responses.

We study the complexity of algorithms in this framework (the number of edits requests needed to find the target clustering) as a function of the error of the initial clustering. We define clustering error in terms of *underclustering* error δ_u and *overclustering* error δ_o (see Section 2). Given that the initial error is often fairly small [1], we would like to develop algorithms whose complexity depends

1. Given 2 different k-clusterings, δ_u and δ_o is at most k^2.

polynomially on δ_u, δ_o and only logarithmically on n, the number of data points. We show that this is indeed possible given that the target clustering satisfies a natural *stability* property (see Section 2). In addition, we develop provably correct algorithms for the well-known correlation-clustering objective (Bansal et al., 2004), which considers pairs of points that are clustered inconsistently with respect to the target clustering (see Section 2).

1.1 Our Results

Our local interactive clustering model is summarized in Section 2.4. This model is then instantiated with specific split/merge algorithms in Section 3 and Section 4. In Section 3 we study the η-merge model. Here we assume that the user may request to split a cluster C_i only if C_i contains points from several ground-truth clusters. The user may request to merge C_i and C_j only if an η-fraction of the points in each C_i and C_j are from the same ground-truth cluster. Note that these restrictions are on the user requests, and not the clustering we are editing. In particular, there may be pairs of clusters C_i and C_j where a smaller fraction of the points are from the same target cluster. But we assume that the user will not request to merge C_i and C_j in such cases because there is not enough evidence to request this merge. Instead, we assume that the user will ask for a split of C_i and/or C_j first, or ask for another merge involving C_i or C_j (where there is more evidence that they need to be merged). This is a realistic restriction because we assume that the merge requests come from high-level observations about the clusters, for example from observing the cluster profiles/summaries.

For this model for $\eta > 0.5$, given an initial clustering with overclustering error δ_o and underclustering error δ_u, we present interactive clustering algorithms that require δ_o split requests and $2(\delta_u + k)\log_{\frac{1}{1-\eta}} n$ merge requests to find the target clustering, where n is the number of points in the data set. For $\eta > 2/3$, given an initial clustering with correlation-clustering error δ_{cc}, we present algorithms that require at most δ_{cc} edit requests to find the target clustering.

In Section 4 we relax the condition on the merges and allow the user to request a merge even if C_i and C_j only have a single point from the same target cluster. We call this the *unrestricted-merge* model. Here the requirement on the accuracy of the user response is much weaker and we need to make further assumptions about the nature of the requests. In particular, we assume that each merge request is chosen uniformly at random from the set of possible merge requests. Under this assumption we present algorithms that with probability at least $1 - \epsilon$ require δ_o split requests and $O(\log \frac{k}{\epsilon}\delta_u^2)$ merge requests to find the target clustering.

Our interactive clustering algorithms take either global or local pairwise similarity data as input. Our most general algorithms use the global average-linkage tree T_{glob} to compute local clustering edits. This tree is constructed from all the data points in the clustering, but it is too large to be directly pruned by users. Still, we can use this global tree to compute accurate local clustering edits. Our split algorithm finds the node in T_{glob} where the corresponding points are first split in two. It is more challenging to design a correct merge procedure, given that we allow "impure" merges, where the clusters in the merge request intersect more than one ground-truth cluster. To perform such merges, in the η-merge model we design an algorithm to extract the "pure" subsets of the two clusters, which must only contain points from the same target cluster. Our algorithm searches for the deepest node in T_{glob} that has enough points from both clusters. In the unrestricted-merge model, we develop another merge algorithm that either merges the two clusters or merges them and splits them. This algorithm always makes progress if the requested merge is "impure," and makes progress on average if it is "pure" (both clusters are subset of the same target cluster).

When the data satisfies stronger assumptions, we present more efficient split and merge algorithms that do not require global pairwise similarity information. These procedures compute the edit by only considering the similarities between the points in the user request.

In Section 5 we demonstrate the effectiveness of our algorithms on real data. We show that for the purposes of splitting known overclustering instances, our split algorithm performs better than well-known algorithms in unsupervised split/merge clustering literature, such as spectral clustering and k-means. We also test our entire interactive clustering framework on the 20 Newsgroup data set, which is known to very challenging for unsupervised (Telgarsky and Dasgupta, 2012; Heller and Ghahramani, 2005; Dasgupta and Hsu, 2008; Dai et al., 2010; Boulis and Ostendorf, 2004; Zhong, 2005) and semi-supervised clustering methods (Basu et al., 2002, 2004). We find that in many scenarios our framework is able find the target Newsgroup clustering after a limited number of edit requests.

1.2 Related work

In this section we give an overview of the related work in the literature.

Interactive Clustering: Interactive clustering models in previous works (Balcan and Blum, 2008; Awasthi and Zadeh, 2010) were inspired by an analogous model for learning under feedback (Angluin, 1998). In this model, the algorithm can propose a hypothesis to the user (in this case, a clustering of the data) and get some feedback regarding the correctness of the current hypothesis. As in our model, the feedback in Balcan and Blum (2008); Awasthi and Zadeh (2010) is in the form of split and merge requests. The goal is to design efficient algorithms that require few user requests. A major limitation in the models of Balcan and Blum (2008) and Awasthi and Zadeh (2010) is that the algorithm is able to choose any arbitrary clustering as the starting point, and can make arbitrary changes to the clustering in each step. Hence these algorithms may propose a series of "bad" clusterings to the user to quickly prune the search space and reach the target clustering. Our interactive clustering model is in the context of an initial clustering; we are restricted to only making local changes to this clustering to correct the errors pointed out by the user. This model is well-motivated by several applications, including the Google application described in Section 5.1.

Active Clustering: Other active clustering frameworks have been proposed, where *active* either refers to selecting which pairwise similarities to consider (not studied here), or requesting user supervision with respect to the ground-truth clustering. For the former problem, Erikkson et al. (2011) study minimizing the number of pairwise similarities needed for an algorithm to compute an accurate clustering. They propose an adaptive algorithm that selects which pairwise similarities to consider. The assumption on the similarity function that they study, which they call *tight clustering condition*, is equivalent to the *strict separation* property studied here. Krishnamurthy et al. (2012) propose a different framework to adaptively select which pairwise similarities to consider. They also study an assumption on the similarity function, which is a generalization of the *strict separation* property that considers the expected pairwise similarities. Their framework recursively splits clusters; they propose a spectral-clustering algorithm to perform the split. They also suggest using the k-means algorithm to perform the split (albeit with no provable guarantees). In our experimental section we compare our proposed split procedures with a similar spectral clustering algorithm and the k-means algorithm (see Section 5.1).

For the latter semi-supervised clustering problem, Nie et al. (2012) develop a clustering algorithm that iteratively extends class labels. It may be used with an empty set of initial labels (unsu-

pervised setting), or with a non-empty set of initial labels (semi-supervised setting). The algorithm minimizes an estimate of Bayes error with respect to the unlabeled instances.

Basu et al. (2002) study the same semi-supervised clustering problem in the context of the k-means objective. Their algorithm has access to the class labels of some seed fraction f of the points in the data set. They propose two variations of a supervised k-means algorithm: one which only uses the known labels to compute the initial clustering (termed *Seeded K-Means*), and another that also enforces that the known labels do not change while the centers/clusters are updated (termed *Constrained K-Means*). They perform experiments on the same 20 Newsgroups data sets that we consider in Section 5.2. The experiments of Basu et al. (2002) show that even though the supervision improves accuracy, even for $f > 0.5$ the output clustering still has many mistakes. They also show that the accuracy depends on how difficult the data set is w.r.t. the separability of the ground-truth clusters, and show improved accuracy on an easier instance (termed *Different-3 Newsgroups*) when compared to a harder one (termed *Same-3 Newsgroups*). In a related work, Basu et al. (2004) study the same problem but with supervision in the form of pairwise *must-link* and *cannot-link* constraints. They propose another supervised k-means algorithm and perform experiments on the same 20 Newsgroups data sets. They are still unable to come close to recovering the ground-truth for a difficult Newsgroups data set (termed *News-sim3*), but have more success with an easier Newsgroup data set (termed *News-diff3*). Ashtiani et al. (2016) also study the k-means objective with supervision in the form of pairwise *must-link* and *cannot-link* constraints, which are modeled as oracle queries. For center-based clustering instances (specified by a Voroni decomposition around a set of centers) that satisfy a separability condition with respect to the cluster centers, they propose an accurate algorithm that requires a limited number of queries; they do not provide any experimental results.

In our experimental section we give a comparison of our results on the 20 Newsgroups data sets with those of Basu et al. (2002) and Basu et al. (2004) (see Section 5.2.4). Our comparison considers the amount of required supervision and accuracy of the final clustering output. We show that only our framework is able to fully recover the ground-truth clustering, albeit we can do this consistently only with some restrictions on the requested merges. On the other hand, Basu et al. (2002) and Basu et al. (2004) do not come close to finding the ground-truth for the harder Newsgroups data sets even when they use a large amount of supervision.

We also note that the supervision considered by Nie et al. (2012), Basu et al. (2002), Basu et al. (2004) and Ashtiani et al. (2016) is harder for the user to provide. Such supervision requires an understanding of the individual data points on the part of the user. The user must study the individual data points to provide instance class labels as in Nie et al. (2012) and Basu et al. (2002), or study pairwise relationships between individual data points to provide must-link and cannot-link constraints as in Basu et al. (2004) and Ashtiani et al. (2016). The supervision in the form of cluster split/merge requests that we consider here is a more realistic form of interaction - it only requires the user to understand the high-level properties of the clusters.

Split/Merge Techniques: Several unsupervised split/merge frameworks have also been proposed (Ding and He, 2002; Lee et al., 2012; Chaudhuri et al., 1992). They focus on designing split/merge algorithms but do not consider any user feedback. Ding and He (2002) propose to use a spectral clustering algorithm to perform the splits. To perform merges, they only consider returning the union of the points in the two clusters. In our experiments, we compare the effectiveness of our split procedures with splits given by spectral clustering (see Section 5.1). With respect to merges, our merge algorithms for $\eta = 1$ are equivalent to the merge proposed by Ding and He (2002), but

in our framework merges may only be initiated by user requests, while Ding and He (2002) propose automatically selecting the next clusters to merge using the max-min-cut criterion (without any user supervision), which they show works well in practice. However, the experiments of Ding and He (2002) also assume that we know the target number of clusters, while our model does not make such assumptions.

Lee et al. (2012) propose a different unsupervised split/merge framework based on optimizing Bayes error. They propose spectral clustering and k-means for computing the splits, which we compare with in our experimental section (see Section 5.1). To perform merges, they again only consider returning the union of the two clusters; the next two clusters to merge are selected by optimizing Bayes error. Unlike Ding and He (2002), they do not assume that we know the number of target clusters. They show good experimental results for video segmentation, but their unsupervised approach may not work for challenging data sets like 20 Newsgroups, where it may not be possible to find the ground truth without user feedback. Chaudhuri et al. (1992) propose another unsupervised split/merge framework for image-segmentation applications. Their split procedure is application-specific because it considers gray-scale values (we cannot compare with it), while their merge is based on edge density.

Data Separability Properties: The data separability (stability) property that we consider in this work is a natural generalization of the "stable marriage" property (see Definition 2), which has been studied in a variety of previous works (Balcan et al., 2008; Bryant and Berry, 2001). It is the weakest among the stability properties that have been studied recently such as *strict separation* and *strict threshold separation* (Balcan et al., 2008; Erikkson et al., 2011; Ackerman et al., 2012; Ackerman and Dasgupta, 2014; Balcan et al., 2014). This property is also known to hold for real-world data. In particular, Voevodski et al. (2012) observed that this property holds for protein sequence data, where similarities are computed with sequence alignment and ground truth clusters correspond to evolutionary-related proteins. Other stronger separability properties have also been considered in the literature (Awasthi and Balcan, 2015).

2. Notation and Preliminaries

Given a data set X of n points we define $\mathcal{C} = \{C_1, C_2, \ldots C_k\}$ to be a k-clustering of X where the C_i's represent the individual clusters. Given two clusterings $\mathcal{C}$ and $\mathcal{C}'$, we next define several notions of clustering error, which are used in our theoretical analysis.

2.1 Clustering Error

Given two clusterings $\mathcal{C}$ and $\mathcal{C}'$, we define the distance between a cluster $C_i \in \mathcal{C}$ and the clustering $\mathcal{C}'$ as:

$$\text{dist}(C_i, \mathcal{C}') = |\{C'_j \in \mathcal{C}' : C'_j \cap C_i \neq \emptyset\}| - 1.$$

This distance is the number of *additional* clusters in $\mathcal{C}'$ that contain points from C_i; it evaluates to 0 when all points in C_i are contained in a single cluster in $\mathcal{C}'$. Naturally, we can then define the distance between $\mathcal{C}$ and $\mathcal{C}'$ as:

$$\text{dist}(\mathcal{C}, \mathcal{C}') = \sum_{C_i \in \mathcal{C}} \text{dist}(C_i, \mathcal{C}').$$

Notice that this notion of clustering distance is asymmetric: $\text{dist}(\mathcal{C},\mathcal{C}') \neq \text{dist}(\mathcal{C}',\mathcal{C})$. Also note that $\text{dist}(\mathcal{C},\mathcal{C}') = 0$ if and only if $\mathcal{C}$ refines $\mathcal{C}'$. Observe that if $\mathcal{C}$ is the ground-truth clustering, and $\mathcal{C}'$ is a proposed clustering, then $\text{dist}(\mathcal{C},\mathcal{C}')$ can be considered an *underclustering error*, and $\text{dist}(\mathcal{C}',\mathcal{C})$ an *overclustering error*.

An underclustering error is an instance of several clusters in a proposed clustering containing points from the same ground-truth cluster; this ground-truth cluster is said to be *underclustered*. Conversely, an overclustering error is an instance of points from several ground-truth clusters contained in the same cluster in a proposed clustering; this proposed cluster is said to be *overclustered*. In the following sections we use $\mathcal{C}^* = \{C_1^*, C_2^*, \ldots C_k^*\}$ to refer to the ground-truth clustering, and use $\mathcal{C}$ to refer to a proposed clustering. We use δ_u to refer to the underclustering error of a proposed clustering, and δ_o to refer to the overclustering error. In other words, we have $\delta_u = \text{dist}(\mathcal{C}^*,\mathcal{C})$ and $\delta_o = \text{dist}(\mathcal{C},\mathcal{C}^*)$. We use δ to denote the sum of the two errors: $\delta = \delta_u + \delta_o$. We call δ the *under/overclustering error*, and use $\delta(\mathcal{C},\mathcal{C}^*)$ to refer to the error of $\mathcal{C}$ with respect to $\mathcal{C}^*$.

We observe that the *under/overclustering error* $\delta(\mathcal{C},\mathcal{C}^*)$ may also be restated in terms of the *bipartite cluster-intersection graph* of Xiang et al. (2012). This bipartite graph $G = (\mathcal{C},\mathcal{C}^*,E)$ describes the relationships between the clusters of $\mathcal{C}$ and $\mathcal{C}^*$, with nodes on one side corresponding to the clusters of $\mathcal{C}$ and nodes on the other side corresponding to the clusters of $\mathcal{C}^*$. The set of (unweighted) edges E corresponds to the intersections between the two sets of clusters: there is an edge if the corresponding clusters intersect on at least one data point (Xiang et al., 2012). Then we can express the *under/overclustering error* $\delta(\mathcal{C},\mathcal{C}^*)$ using vol(G), the sum of the degrees of the nodes of G:

$$\delta(\mathcal{C},\mathcal{C}^*) = \text{vol}(G) - |\mathcal{C}| - |\mathcal{C}^*|.$$

In addition to the *under/overclustering* error defined above, in parts of our analysis we define the distance between two clusterings using the *correlation-clustering* objective function. Given a proposed clustering $\mathcal{C}$, and a ground-truth clustering $\mathcal{C}^*$, we define the correlation-clustering error δ_{cc} as the number of (ordered) pairs of points that are clustered *inconsistently* with $\mathcal{C}^*$:

$$\delta_{cc} = |\{(u,v) \in X \times X : c(u,v) \neq c^*(u,v)\}|,$$

where $c(u,v) = 1$ if u and v are in the same cluster in $\mathcal{C}$, and 0 otherwise; $c^*(u,v) = 1$ if u and v are in the same cluster in $\mathcal{C}^*$, and 0 otherwise. In our analysis we also call each such pair of points a *pairwise correlation-clustering error*.

Note that as before we may divide the correlation-clustering error δ_{cc} into overclustering component δ_{cco} and underclustering component δ_{ccu}:

$$\delta_{cco} = |\{(u,v) \in X \times X : c(u,v) = 1 \text{ and } c^*(u,v) = 0\}|,$$
$$\delta_{ccu} = |\{(u,v) \in X \times X : c(u,v) = 0 \text{ and } c^*(u,v) = 1\}|.$$

Observe that by definition $\delta_{cc} = \delta_{cco} + \delta_{ccu}$.

2.2 Definitions

Our interactive clustering model concerns computing *local* clustering edits, which only change the part of the clustering that the user is complaining about. This intuition is captured by the following definition.

Definition 1 (Local algorithm) *We say that an interactive clustering algorithm is* local *if in each iteration only the cluster assignments of the points involved in the edit request may be changed. If the user requests to split C_i, the algorithm may only reassign the points in C_i. If the user requests to merge C_i and C_j, the algorithm may only reassign the points in $C_i \cup C_j$.*

We next define the properties of a clustering that we study in this work, which describe how separable the ground-truth clusters are.

Definition 2 (Stability) *Given a clustering $\mathcal{C} = \{C_1, C_2, \cdots C_k\}$ over a domain X and a similarly function $S : X \times X \mapsto \Re$, we say that $\mathcal{C}$ satisfies stability with respect to S if for all $i \neq j$, and for all $A \subset C_i$ and $A' \subseteq C_j$, $S(A, C_i \setminus A) > S(A, A')$, where for any two sets A, A', $S(A, A') = E_{x \in A, y \in A'} S(x, y)$.*

In addition to the *stability* property, we also study the stronger *strict separation* and *strict threshold separation* properties, which were first introduced in Balcan et al. (2008). They are defined below. Clearly, we can verify that *strict separation* and *strict threshold separation* imply *stability*.

Definition 3 (Strict separation) *Given a clustering $C = \{C_1, C_2, \cdots C_k\}$ over a domain X and a similarly function $S : X \times X \mapsto \Re$, we say that C satisfies strict separation with respect to S if for all $i \neq j$, $x, y \in C_i$ and $z \in C_j$, $S(x, y) > S(x, z)$.*

Definition 4 (Strict threshold separation) *Given a clustering $C = \{C_1, C_2, \cdots C_k\}$ over a domain X and a similarly function $S : X \times X \mapsto \Re$, we say that C satisfies strict threshold separation with respect to S if there exists a threshold t such that, for all i, $x, y \in C_i$, $S(x, y) > t$, and, for all $i \neq j$, $x \in C_i, y \in C_j$, $S(x, y) \leq t$.*

We model the user as an oracle that provides edit requests. In order for our algorithms to make progress, the oracle requests must be somewhat consistent with the target clustering, which is captured by the following definitions.

Definition 5 (η-merge model) *The oracle may request to split a cluster C_i only if C_i contains points from more than one target cluster. The oracle may request to merge two clusters C_i and C_j only if at least an η-fraction of the points in each C_i and C_j belong to the same target cluster.*

Definition 6 (unrestricted-merge model) *The oracle may request to split a cluster C_i only if C_i contains points from more than one target cluster. The oracle may request to merge two clusters C_i and C_j only if C_i and C_j both intersect the same target cluster.*

Note that the assumptions about the split requests are the same in both models. With respect to the merges, in the η-merge model the oracle may request to merge two clusters if both have a *constant fraction* of points from the same target cluster. In the unrestricted-merge model, the oracle may request to merge two clusters even if both only have *some* points from the same target cluster. In each model, we will refer to the split/merge requests that satisfy these definitions as *valid* edit requests.

Also note that in the η-merge model the restrictions on valid merge requests are with respect to the oracle requests, and not the clustering we are editing. In particular, there may be pairs of clusters C_i and C_j where a smaller fraction of the points are from the same target cluster. But we assume that the oracle will not request to merge C_i and C_j in such cases. Instead, we assume that the oracle will ask for a split of C_i and/or C_j first, or ask for a different (valid) merge involving C_i or C_j.

2.3 Generalized clustering error

We observe that the clustering errors defined in the previous section may be generalized by abstracting their common properties. We define the following properties of what we call *natural* clustering error, which is any integer-valued error that decreases when we locally improve the proposed clustering.

Definition 7 *We say that a clustering error is* natural *if it satisfies the following properties:*

- *If there exists a cluster C_i that contains points from C_j^* and some other ground-truth cluster(s), then splitting this cluster into two clusters $C_{i,1} = C_i \cap C_j^*$ and $C_{i,2} = C_i - C_{i,1}$ must decrease the error.*
- *If there exist two clusters that contain only points from the same target cluster, then merging them into one cluster must decrease the error.*
- *The error is integer-valued.*

We expect a lot of definitions of clustering error to satisfy the above criteria (especially the first two properties), in addition to other domain-specific criteria. Clearly, the under/overclustering error $\delta = \delta_u + \delta_o$ and the correlation-clustering error δ_{cc} are also *natural* clustering errors (Claim 8). Given a *natural* clustering error γ, a proposed clustering $\mathcal{C}$ and the target clustering $\mathcal{C}^*$, we will use $\gamma(\mathcal{C}, \mathcal{C}^*)$ to denote the magnitude of the error of $\mathcal{C}$ with respect to $\mathcal{C}^*$. We can also prove that the under/overclustering error defined in the previous section is the lower-bound on any *natural* clustering error (Theorem 9).

Claim 8 *The under/overclustering error and the correlation clustering error satisfy Definition 7 and hence are natural clustering errors.*

Theorem 9 *For any* natural *clustering error γ, any proposed clustering $\mathcal{C}$, and any target clustering $\mathcal{C}^*$, $\gamma(\mathcal{C}, \mathcal{C}^*) \geq \delta(\mathcal{C}, \mathcal{C}^*)$.*

Proof Given any proposed clustering $\mathcal{C}$, and any target clustering $\mathcal{C}^*$, we may transform $\mathcal{C}$ into $\mathcal{C}^*$ via the following sequence of edits. First, we split all overclustering instances using the following iterative procedure: while there exists a cluster C_i that contains points from C_j^* and some other ground-truth cluster(s), we split it into two clusters $C_{i,1} = C_i \cap C_j^*$ and $C_{i,2} = C_i - C_{i,1}$. Note that this iterative split procedure will require exactly δ_o split edits, where δ_o is the initial overclustering error. Then, when we are left with only "pure" clusters (each intersects exactly one target cluster), we merge all underclustering instances using the following iterative procedure: while there exist two clusters C_i and C_j that contain only points from the same target cluster, merge C_i and C_j. Note that this iterative merge procedure will require exactly δ_u merge edits, where δ_u is the initial underclustering error. Let us use γ to refer to any *natural* clustering error of $\mathcal{C}$ with respect to $\mathcal{C}^*$. By the first property of *natural* clustering error, each split must have decreased γ by at least one. By the second property, each merge must have decreased γ by at least one as well. Given that we performed exactly $\delta = \delta_o + \delta_u$ edits, it follows that initially $\gamma(\mathcal{C}, \mathcal{C}^*)$ must have been at least δ. ■

For additional discussion about comparing clusterings see Meilă (2007). Note that several criteria discussed in Meilă (2007) satisfy our first two properties (given that we replace "must decrease the error" with "must increase the similarity" in Definition 7). In addition, the Rand criteria and the Mirkin criteria discussed in Meilă (2007) are closely related to the correlation clustering error defined here (all three measures are a function of the number of pairs of points that are clustered incorrectly).

2.4 Local interactive clustering

We now give a general overview of our local interactive clustering model. This model is then instantiated by specific split/merge algorithms in Section 3 and Section 4 (depending on the assumptions about the oracle requests and the separability of the data).

Our approach is outlined in Figure 1 - we iteratively keep on making the local edits requested by the oracle. Note that the oracle does not provide the algorithm with any additional information about how to execute the split/merge. The description in Figure 1 also assumes that the requests come one at a time. Batch requests can be handled as well by queuing them in arbitrary order and then executing them one at a time (as long as the clusters in the request have not changed).

Given that the oracle requests must be somewhat consistent with the target clustering (see our assumptions about the validity of such requests in Section 2.2), and that our algorithms must therefore make progress with respect to finding the target clustering (see our analysis in Section 3 and Section 4), we expect the loop to terminate when we reach the target clustering.

Also note that the description in Figure 1 does not say anything about the sequence of oracle requests. The algorithms for the η-merge model in Section 3 are consistent with this description; they make no additional assumptions about this sequence. However, the algorithms in the unrestricted-merge model in Section 4 do make an additional assumption that each merge request is drawn uniformly at random from the set of valid merge requests.

Figure 1: Local interactive clustering model

while (there exists split/merge request from oracle)

- perform corresponding local clustering edit

3. The η-merge model

In this section we describe and analyze our split/merge algorithms in the η-merge model. As a preprocessing step for all our split/merge algorithms, we first run the hierarchical average-linkage algorithm on all the points in the data set to compute the global average-linkage tree, which we denote by T_{glob}. The leaf nodes in this tree contain the individual points, and the root node contains all the points. The tree is computed in a bottom-up fashion: starting with the leafs in each iteration the two most similar nodes are merged, where the similarity between two nodes N_1 and N_2 is the average similarity between the points in N_1 and the points in N_2.

We assign a label "impure" to each cluster in the initial clustering; these labels are used by the merge procedure but are not visible to the user. Given a split or merge request from the oracle, a local clustering edit is computed from the global tree T_{glob} as described in Figure 2 and Figure 3.

To implement Step 1 in Figure 2, we start at the root of T_{glob} and "follow" the points in C_i down one of the branches until we find a node that splits them. In order to implement Step 2 in Figure 3, it suffices to perform a post-order traversal of T_{glob} and return the first node that has "enough" points from both clusters.

Figure 2: Split procedure

Algorithm: SPLIT PROCEDURE

Input: Cluster C_i, global average-linkage tree T_{glob}.

1. Search T_{glob} to find the node N at which the set of points in C_i are first split in two.
2. Let N_1 and N_2 be the children of N. Set $C_{i,1} = N_1 \cap C_i$, $C_{i,2} = N_2 \cap C_i$.
3. Delete C_i and replace it with $C_{i,1}$ and $C_{i,2}$. Mark the two new clusters as "impure".

The split procedure is fairly intuitive: if the average-linkage tree is consistent with the target clustering, it suffices to find the node in the tree where the corresponding points are first split in two. It is more challenging to develop a correct merge procedure: note that Step 2 in Figure 3 is only correct if $\eta > 0.5$, which ensures that if two nodes in the tree have more than an η-fraction of the points from C_i and C_j, one must be an ancestor of the other. If the average-linkage tree is consistent with the ground-truth, then clearly the node equivalent to the corresponding target cluster (that C_i and C_j both intersect) will have enough points from C_i and C_j; therefore the node that we find in Step 2 must be this node or one of its descendants. In addition, because our merge procedure may replace two clusters with three, we require pure/impure labels for the merge requests to terminate: "pure" clusters may only have other points added to them, and retain this label throughout the execution of the algorithm.

Figure 3: Merge procedure

Algorithm: MERGE PROCEDURE

Input: Clusters C_i and C_j, global average-linkage tree T_{glob}.

1. If C_i is marked as "pure" set $\eta_1 = 1$ else set $\eta_1 = \eta$. Similarly set η_2 for C_j.
2. Search T_{glob} for a node of maximal depth N that contains *enough* points from C_i and C_j: $|N \cap C_i| \geq \eta_1 |C_i|$ and $|N \cap C_j| \geq \eta_2 |C_j|$.
3. Replace C_i by $C_i \setminus N$, replace C_j by $C_j \setminus N$.
4. Add a new cluster containing $N \cap (C_i \cup C_j)$, mark it as "pure".

We now state the performance guarantee for these split and merge algorithms.

Theorem 10 *Suppose the target clustering satisfies stability, and the initial clustering has overclustering error δ_o and underclustering error δ_u. In the η-merge model, for any $\eta > 0.5$, the algorithms in Figure 2 and Figure 3 require at most δ_o split requests and $2(\delta_u + k) \log_{\frac{1}{1-\eta}} n$ merge requests to find the target clustering.*

In order to prove the theorem, we must do some preliminary analysis. First, we observe that if the target clustering satisfies stability, then every node of the average-linkage tree must be *laminar* (consistent) with respect to the target clustering. Informally, each node in a hierarchical clustering tree T is said to be *laminar* (consistent) with respect to the clustering $\mathcal{C}$ if for each cluster $C_i \in \mathcal{C}$, the points in C_i are first grouped together in T before they are grouped with points from any other cluster $C_{j \neq i}$. We formally state and prove these observations next.

Definition 11 (Laminar) *A node N is laminar with respect to a clustering $\mathcal{C}$ if for each cluster $C_i \in \mathcal{C}$ we have either $N \cap C_i = \emptyset$, $N \subseteq C_i$, or $C_i \subset N$.*

Lemma 12 *Suppose the ground-truth clustering $\mathcal{C}^*$ over a domain X satisfies stability with respect to a similarity function S. Let T be the average-linkage tree for X constructed with S. Then every node in T is laminar w.r.t. $\mathcal{C}^*$.*

Proof The proof of this statement can be found in Balcan et al. (2008). The intuition is that if there is a node in T that is not laminar w.r.t. C^*, then the average-linkage algorithm, at some step, must have merged $A \subseteq C_i^*$, with $B \subset C_j^*$ for some $i \neq j$. However, this will contradict the stability property for the sets A and/or B. ■

It follows that the split computed by the algorithm in Figure 2 must also be consistent with the target clustering; we call such splits *clean*.

Definition 13 (Clean split) *A partition (split) of a cluster C_i into clusters $C_{i,1}$ and $C_{i,2}$ is said to be* clean *if $C_{i,1}$ and $C_{i,2}$ are non-empty, and for each ground-truth cluster C_j^* such that $C_j^* \cap C_i \neq \emptyset$, either $C_j^* \cap C_i = C_j^* \cap C_{i,1}$ or $C_j^* \cap C_i = C_j^* \cap C_{i,2}$.*

We now prove the following properties regarding the correctness of the split/merge procedures; these properties are then used in the proof of Theorem 10.

Lemma 14 *If the ground-truth clustering satisfies stability and $\eta > 0.5$ then,*

- ***a.*** *The split procedure in Figure 2 always produces a clean split.*
- ***b.*** *The new cluster added by the merge procedure in Figure 3 must be "pure", i.e., it must contain points from a single ground-truth cluster.*

Proof a. For purposes of contradiction, suppose the returned split is not clean: $C_{i,1}$ and $C_{i,2}$ contain points from the same ground-truth cluster C_j^*. It must be the case that C_i contains points from several ground-truth clusters, which implies that w.l.o.g. $C_{i,1}$ contains points from some other ground-truth cluster $C_{l \neq j}^*$. This implies that N_1 is not laminar w.r.t. $\mathcal{C}^*$, which contradicts Lemma 12. **b.** By our assumption, at least $\eta |C_i|$ points from C_i and $\eta |C_j|$ points from C_j are from the same ground-truth cluster C_l^*. Clearly, the node N' in T_{glob} that is equivalent to C_l^* (which contains all the points in C_l^* and no other points) must contain *enough* points from C_i and C_j, and only ascendants and

descendants of N' may contain more than an $\eta > 1/2$ fraction of points from both clusters. Therefore, the node N that we find in Step 2 in Figure 3 must be N' or one of its descendants, and will only contain points from C_l^*. ■

We also prove an additional property of the split algorithm - it always makes progress with respect to reducing overclustering error. This property is then used in the proof of Theorem 10.

Lemma 15 *If the ground-truth clustering satisfies stability, then the split algorithm in Figure 2 must reduce overclustering error by exactly 1.*

Proof Suppose we execute Split(C_1), and call the resulting clusters C_2 and C_3. Call δ_1 the overclustering error before the split, and δ_2 the overclustering error after the split. Let's use k_1 to refer to the number of ground-truth clusters that intersect C_1, and define k_2 and k_3 in the same manner. Due to the *clean split* property, no ground-truth cluster can intersect both C_2 and C_3, therefore it must be the case that $k_2 + k_3 = k_1$. Also, clearly $k_2, k_3 > 0$. Therefore we have:

$$\begin{aligned} \delta_2 &= \delta_1 - (k_1 - 1) + (k_2 - 1) + (k_3 - 1) \\ &= \delta_1 - k_1 + (k_2 + k_3) - 1 \\ &= \delta_1 - 1. \end{aligned}$$

■

Using Lemma 14 and Lemma 15, we can now prove the bounds on the number of split and merge requests stated in Theorem 10.

Proof [Proof of Theorem 10] We first give a bound on the number of splits. Lemma 15 shows that each split reduces the overclustering error by exactly 1. It is easy to verify that merges cannot increase overclustering error. Therefore the total number of splits may be at most δ_o.

We next give the argument about the number of *impure* and *pure* merges. We call a merge *pure* if both clusters have the label "pure", and call it *impure* otherwise. We first argue that we cannot have too many impure merges before each cluster in C is marked pure. Consider the clustering $P = \{C_i \cap C_j^* \mid C_i \text{ is marked "impure" and } C_i \cap C_j^* \neq \emptyset\}$. Clearly, at the start $|P| = \delta_u + k$. A merge does not increase the number of clusters in P, and the splits do not change P at all (because of the *clean split* property). Moreover, each impure merge *depletes* some $P_i \in P$ by moving at least $\eta|P_i|$ of its points to a pure cluster. Clearly, we can then have at most $\log_{1/(1-\eta)} n$ merges depleting each P_i. Since each impure merge must deplete some P_i, it must be the case that we can have at most $(\delta_u + k)\log_{1/(1-\eta)} n$ impure merges in total.

Notice that a pure cluster can only be created by an impure merge, and there can be at most one pure cluster created by each impure merge. Clearly, a pure merge removes exactly one pure cluster. Therefore the number of pure merges may be at most the total number of pure clusters that are created, which is at most the total number of impure merges. Therefore the total number of merges must be less than $2(\delta_u + k)\log_{1/(1-\eta)} n$. ■

We can also restate the run-time bound in Theorem 10 in terms of any *natural* clustering error γ. The following corollary follows from Theorem 10 and Theorem 9.

Corollary 16 *Suppose the target clustering satisfies stability, and the initial clustering has clustering error γ, where γ is any* natural *clustering error as defined in Definition 7. In the η-merge model, for any $\eta > 0.5$, the algorithms in Figure 2 and Figure 3 require at most $O((\gamma + k)\log_{\frac{1}{1-\eta}} n)$ edit requests to find the target clustering.*

3.1 Algorithms for correlation-clustering error

To efficiently find the target clustering with respect to the correlation clustering error, we propose a different merge procedure, which is described in Figure 4.

Figure 4: Merge procedure for the *correlation-clustering* objective

Algorithm: MERGE PROCEDURE

Input: Clusters C_i and C_j, global average-linkage tree T_{glob}

Search T_{glob} for a node of maximal depth N that contains *enough* points from C_i and C_j: $|N \cap C_i| \geq \eta|C_i|$ and $|N \cap C_j| \geq \eta|C_j|$
if $|C_i| \geq |C_j|$ **then**
  Replace C_i by $C_i \cup (N \cap C_j)$
  Replace C_j by $C_j \setminus N$
else
  Replace C_i by $C_i \setminus N$
  replace C_j by $C_j \cup (N \cap C_i)$
end if

Here instead of creating a new "pure" cluster, we add the corresponding points to the larger of the two clusters in the merge request. Notice that this algorithm is much simpler than the merge algorithm in Figure 3, and does not require pure/impure labels. Using this merge procedure and the split procedure presented earlier gives the following performance guarantee.

Theorem 17 *Suppose the target clustering satisfies stability, and the initial clustering has correlation-clustering error δ_{cc}. In the η-merge model, for any $\eta > 2/3$, using the split and merge procedures in Figures 2 and 4 requires at most δ_{cc} edit requests to find the target clustering.*

Proof Recall that we may express the correlation clustering error δ_{cc} as a summation of overclustering error δ_{cco} and underclustering error δ_{ccu} (see Section 2.1). Consider the contributions of individual points to δ_{cco} and δ_{ccu}, which are defined as:

$$\delta_{cco}(u) = |\{v \in X : c(u,v) = 1 \text{ and } c^*(u,v) = 0\}|,$$
$$\delta_{ccu}(u) = |\{v \in X : c(u,v) = 0 \text{ and } c^*(u,v) = 1\}|.$$

We first argue that a split of a cluster C_i must reduce δ_{cc}. Given that the split is *clean*, it is easy to verify that the outcome may not increase $\delta_{ccu}(u)$ for any $u \in C_i$. We can also verify that for each $u \in C_i$, $\delta_{cco}(u)$ must decrease by at least 1. This completes the argument, given that for all other points $w \notin C_i$, $\delta_{cco}(w)$ and $\delta_{ccu}(w)$ remain unchanged.

We now argue that if $\eta > 2/3$, each merge of C_i and C_j must reduce δ_{cc}. Without loss of generality, suppose that $|C_i| \geq |C_j|$, and let us use P to refer to the "pure" subset of C_j that is

moved to C_i. We observe that the outcome must remove at least δ_1 *pairwise correlation-clustering errors* (see Section 2.1), where δ_1 satisfies $\delta_1 \geq 2|P|(\eta|C_i|)$. Similarly, we observe that the outcome may add at most δ_2 pairwise correlation-clustering errors, where δ_2 satisfies:

$$\delta_2 \leq 2|P|((1-\eta)|C_i|) + 2|P|((1-\eta)|C_j|) \leq 4|P|((1-\eta)|C_i|).$$

It follows that for $\eta > 2/3$, δ_1 must exceed δ_2; therefore the number of pairwise correlation-clustering errors must decrease, giving a lower correlation-clustering error. ■

Observe that the runtime bound in Theorem 17 is tight up to a constant: in some instances any *local* algorithm requires at least $\delta_{cc}/2$ edits to find the target clustering. To verify this, suppose the target clustering is composed of n singleton clusters, and the initial clustering contains $n/2$ clusters of size 2. In this instance, the initial correlation clustering error $\delta_{cc} = n$, and the oracle must issue at least $n/2$ split requests before we reach the target clustering (one for each of the $n/2$ initial clusters).

3.2 Algorithms under stronger assumptions

When the data satisfies stronger separability properties we may simplify the presented algorithms and/or obtain better performance guarantees. In particular, if the data satisfies the *strict separation* property (see Definition 3), we may change the split and merge algorithms to use the local average-linkage tree, which is constructed from only the points in the edit request. In addition, if the data satisfies *strict threshold separation* (see Definition 4), we may remove the restriction on η and use a different merge procedure that is correct for any $\eta > 0$.

Theorem 18 *Suppose the target clustering satisfies strict separation, and the initial clustering has overclustering error δ_o and underclustering error δ_u. In the η-merge model, for any $\eta > 0.5$, the algorithms in Figure 5 and Figure 6 require at most δ_o split requests and $2(\delta_u + k)\log_{\frac{1}{1-\eta}} n$ merge requests to find the target clustering.*

Proof Let us use $\mathcal{L}^*$ to refer to the ground-truth clustering of the points in the split/merge request. If the target clustering satisfies strict separation, it is easy to verify that every node in the local average-linkage tree T_{loc} must be laminar (consistent) w.r.t. $\mathcal{L}^*$. We can then use this observation to prove the equivalent of Lemma 14 for strict separation for the split procedure in Figure 5 and the merge procedure in Figure 6. Similarly, we can prove the equivalent of Lemma 15 for strict separation for the split procedure in Figure 5. The analysis in Theorem 10 remains unchanged. ■

Theorem 19 *Suppose the target clustering satisfies strict threshold separation, and the initial clustering has overclustering error δ_o and underclustering error δ_u. In the η-merge model, for any $\eta > 0$, the algorithms in Figure 5 and Figure 7 require at most δ_o split requests and $2(\delta_u + k)\log_{\frac{1}{1-\eta}} n$ merge requests to find the target clustering.*

Proof If the target clustering satisfies strict threshold separation, we can verify that for any $\eta > 0$, the split procedure in Figure 5 and the merge procedure in Figure 7 still have the properties stated in Lemma 14 and Lemma 15. The analysis in Theorem 10 remains unchanged.

To verify that the split procedure always produces a clean split, again let us use $\mathcal{L}^*$ to refer to the ground-truth clustering of the points in the split request. If the target clustering satisfies strict threshold separation, we can again verify that every node in the local average-linkage tree T_{loc} must be laminar (consistent) w.r.t. $\mathcal{L}^*$. It follows that the split procedure always produces a clean split and must reduce overclustering error by exactly 1. Note that clearly this argument does not depend on the setting of η.

We now verify that the new cluster added by the merge procedure in Figure 7 must be "pure" (must contain points from a single target cluster). Observe that due to the strict threshold separation property, in the graph G in Figure 7, all pairs of points from the same target cluster must be connected before any pairs of points from different target clusters. It follows that the first component that contains at least an η-fraction of points from C_i and C_j must be "pure". Note that this argument applies for any $\eta > 0$. ■

Figure 5: Split procedure under stronger assumptions

Algorithm: SPLIT PROCEDURE

Input: Cluster C_i, local average-linkage tree T_{loc}.

1. Let $C_{i,1}$ and $C_{i,2}$ be the children of the root in T_{loc}.
2. Delete C_i and replace it with $C_{i,1}$ and $C_{i,2}$. Mark the two new clusters as "impure".

Figure 6: Merge procedure under strict separation

Algorithm: MERGE PROCEDURE

Input: Clusters C_i and C_j, local average-linkage tree T_{loc}.

1. If C_i is marked as "pure" set $\eta_1 = 1$ else set $\eta_1 = \eta$. Similarly set η_2 for C_j.
2. Search T_{loc} for a node of maximal depth N that contains *enough* points from C_i and C_j: $|N \cap C_i| \geq \eta_1|C_i|$ and $|N \cap C_j| \geq \eta_2|C_j|$.
3. Replace C_i by $C_i \setminus N$, replace C_j by $C_j \setminus N$.
4. Add a new cluster containing $N \cap (C_i \cup C_j)$, mark it as "pure".

As in Corollary 16, we may also restate the run-time bounds in Theorem 18 and Theorem 19 in terms of any natural clustering error γ. The following corollaries follow from Theorem 18, Theorem 19 and Theorem 9.

Corollary 20 *Suppose the target clustering satisfies strict separation, and the initial clustering has clustering error γ, where γ is any* natural *clustering error as defined in Definition 7. In the η-merge model, for any $\eta > 0.5$, the algorithms in Figure 5 and Figure 6 require at most $O((\gamma+k)\log_{\frac{1}{1-\eta}} n)$ edit requests to find the target clustering.*

Figure 7: Merge procedure under strict threshold separation

Algorithm: MERGE PROCEDURE

Input: Clusters C_i and C_j.

1. If C_i is marked as "pure" set $\eta_1 = 1$ else set $\eta_1 = \eta$. Similarly set η_2 for C_j.
2. Let $G = (V, E)$ be a graph where $V = C_i \cup C_j$ and $E = \emptyset$. Set $N = \emptyset$.
3. While true:
   Connect the next-closest pair of points in G;
   Let $\hat{C}_1, \hat{C}_2, \ldots, \hat{C}_m$ be the connected components of G;
   if there exists $\hat{C}_l$ such that $|\hat{C}_l \cap C_i| \geq \eta |C_i|$ and $|\hat{C}_l \cap C_j| \geq \eta |C_j|$ **then**
      $N = \hat{C}_l$;
      break;
   end if
4. Replace C_i by $C_i \setminus N$, replace C_j by $C_j \setminus N$.
5. Add a new cluster containing N, mark it as "pure".

Corollary 21 *Suppose the target clustering satisfies strict threshold separation, and the initial clustering has clustering error γ, where γ is any* natural *clustering error as defined in Definition 7. In the η-merge model, for any $\eta > 0$, the algorithms in Figure 5 and Figure 7 require at most $O((\gamma + k) \log_{\frac{1}{1-\eta}} n)$ edit requests to find the target clustering.*

4. The unrestricted-merge model

In this section we further relax the assumptions about the nature of the oracle requests. As before, the oracle may request to split a cluster if it contains points from two or more target clusters. For merges, now the oracle may request to merge C_i and C_j if both clusters contain only a single point from the same ground-truth cluster. We note that this is a minimal set of assumptions for a local algorithm to make progress, otherwise the oracle may request irrelevant splits or merges that prevent the algorithm from finding the target clustering. For this model we propose the merge algorithm described in Figure 8. The split algorithm remains the same as in Figure 2.

To provably find the ground-truth clustering in this setting we require that each merge request must be chosen uniformly at random from the set of *valid* merge requests. This assumption is consistent with the observation in Awasthi and Zadeh (2010) that in the unrestricted-merge model with arbitrary request sequences, even very simple clustering instances require a prohibitively large number of requests. We do not make additional assumptions about the nature of the split requests; in each iteration any *valid* split may be proposed by the oracle. In this setting our algorithms have the following performance guarantee.

Theorem 22 *Suppose the target clustering satisfies stability, and the initial clustering has overclustering error δ_o and underclustering error δ_u. In the unrestricted-merge model, assuming that each*

Figure 8: Merge procedure for the unrestricted-merge model

Algorithm: MERGE PROCEDURE

Input: Clusters C_i and C_j, global average-linkage tree T_{avg}.

1. Let C'_i, C'_j = Split($C_i \cup C_j$), where the split is performed as in Figure 2.
2. Delete C_i and C_j.
3. If the sets C'_i and C'_j are the same as C_i and C_j, then add $C_i \cup C_j$, otherwise add C'_i and C'_j.

merge request is chosen uniformly at random from the set of valid merge requests, with probability at least $1 - \epsilon$, the algorithms in Figure 2 and Figure 8 require δ_o split requests and $O(\log \frac{k}{\epsilon} \delta_u^2)$ merge requests to find the target clustering.

Theorem 22 is proved in the following lemmas. We first state a lemma regarding the correctness of the Algorithm in Figure 8. We argue that if the algorithm merges C_i and C_j, it must be the case that both C_i and C_j only contain points from the same ground-truth cluster.

Lemma 23 *If the algorithm in Figure 8 merges C_i and C_j in Step 3, it must be the case that $C_i \subset C_l^*$ and $C_j \subset C_l^*$ for some ground-truth cluster C_l^*.*

Proof We prove the contrapositive. Suppose C_i and C_j both contain points from C_l^*, and in addition $C_i \cup C_j$ contains points from some other ground-truth cluster. Let us define $S_1 = C_l^* \cap C_i$ and $S_2 = C_l^* \cap C_j$. Because the clusters C'_i, C'_j result from a *clean* split, it follows that $S_1, S_2 \subset C'_i$ or $S_1, S_2 \subset C'_j$. Without loss of generality, assume $S_1, S_2 \subset C'_i$. Then clearly $C'_i \neq C_i$ and $C'_i \neq C_j$, so C_i and C_j are not merged. ■

The δ_o bound on the number of split requests follows from the observation that each split reduces the overclustering error by exactly 1 (see Lemma 15), and the fact that the merge procedure does not increase overclustering error (see Lemma 24).

Lemma 24 *The merge algorithm in Figure 8 does not increase overclustering error.*

Proof Suppose $C_i \cup C_j$ intersects several ground-truth clusters, and hence we obtain two new clusters C'_i, C'_j. Let us call δ_1 the overclustering error before the merge, and δ_2 the overclustering error after the merge. Let's use k_1 to refer to the number of ground-truth clusters that intersect C_i, k_2 to refer to the number of ground-truth clusters that intersect C_j, and define k'_1 and k'_2 in the same manner. The new clusters C'_i and C'_j result from a "clean" split, therefore no ground-truth cluster may intersect both of them. It follows that $k'_1 + k'_2 < k_1 + k_2$. Therefore we now have:

$$\begin{aligned} \delta_2 &= \delta_1 - (k_1 - 1) - (k_2 - 1) + (k'_1 - 1) + (k'_2 - 1) \\ &= \delta_1 - (k_1 + k_2) + (k'_1 + k'_2) < \delta_1. \end{aligned}$$

If C_i and C_j are both subset of the same ground-truth cluster, then clearly the merge operation has no effect on the overclustering error. ■

Finally, Lemma 25 and Lemma 26 bound the number of impure and pure merges. Here we call a proposed merge *pure* if both clusters are subset of the same ground-truth cluster, and *impure* otherwise.

Lemma 25 *The merge algorithm in Figure 8 requires at most δ_u impure merge requests.*

Proof We argue that the result of each impure merge request must reduce the underclustering error δ_u by at least 1. Recall from Section 2.1 that we may express δ_u as a summation of underclustering error with respect to each target cluster. Suppose the oracle requests to merge C_i and C_j, and C_i' and C_j' are the resulting clusters. Clearly, the local edit has no effect on the underclustering error with respect to target clusters that do not intersect C_i or C_j. In addition, because the new clusters C_i' and C_j' result from a *clean* split, for target clusters that intersect exactly one of C_i, C_j, the underclustering error must stay the same. For target clusters that intersect both C_i and C_j, the underclustering error must decrease by exactly one; the number of such target clusters is at least one. ■

Lemma 26 *Assuming that each merge request is chosen uniformly at random from the set of valid merge requests, the probability that the algorithm in Figure 8 requires more than $O(\log \frac{k}{\epsilon}\delta_u^2)$ pure merge requests is less than ϵ.*

Proof We first consider the pure merge requests involving points from some ground-truth cluster C_i^*, the total number of pure merge requests (involving any ground-truth cluster) can then be bounded with a union-bound. To facilitate our argument, let us assign an identifier to each cluster containing points from C_i^* in the following manner:

1. Maintain a *cluster-id* variable, which is initialized to 1. To assign a "new" identifier to a cluster, set its identifier to *cluster-id*, and increment *cluster-id*.
2. In the initial clustering, assign a *new* identifier to each cluster containing points from C_i^*.
3. When we split a cluster containing points from C_i^*, assign its identifier to the newly-formed cluster containing points from C_i^*.
4. When we merge two clusters that are not both subset of the same target cluster, if one of the clusters contains points from C_i^*, assign its identifier to the newly-formed cluster containing points from C_i^*. If both clusters contain points from C_i^*, assign a *new* identifier to the newly-formed cluster containing points from C_i^*.
5. When we merge two clusters C_1 and C_2, and both contain only points from C_i^*, if the outcome is one new cluster, assign it a *new* identifier. If the outcome is two new clusters, assign them the identifiers of C_1 and C_2.

Observe that when clusters containing points from C_i^* are assigned identifiers in this manner, the maximum value of *cluster-id* is bounded by $O(\delta_i)$, where δ_i denotes the underclustering error

of the initial clustering with respect to C_i^*: $\delta_i = \mathrm{dist}(C_i^*, C)$. To verify this, consider that we assign exactly $\delta_i + 1$ new identifiers in Step 2, and each time we assign a new identifier in Steps 4 and 5, δ_i decreases by one.

We say that a *pure* merge request involving points from C_i^* is *original* if the user has never asked us to merge clusters with the given identifiers, otherwise we say that this merge request is *repeated*. Given that the maximum value of *cluster-id* is bounded by $O(\delta_i)$, the total number of *original* merge requests must be $O(\delta_i^2)$.

We now argue that if a merge request is not original, we can lower bound the probability that it will result in the merging of the two clusters. For a repeated merge request $M_i = \mathrm{Merge}(C_1, C_2)$, let X_i be a random variable defined as follows:

$$X_i = \begin{cases} 1 & \text{if neither } C_1 \text{ nor } C_2 \text{ have been involved in a pure merge request since} \\ & \text{the last time a merge of clusters with these identifiers was proposed.} \\ 0 & \text{otherwise.} \end{cases}$$

Clearly, when $X_i = 1$ it must be the case that C_1 and C_2 are merged. Assuming that each merge request is chosen uniformly at random from the set of valid merge requests, we observe that $\Pr[X_i = 1] > \frac{1}{2\delta_i+1}$. To verify this, observe that in each merge request the probability that the user requests to merge C_1 and C_2 is $\frac{1}{m}$, and the probability that the user requests a pure merge involving C_1 or C_2 and some other cluster is less than $\frac{2\delta_i}{m}$, where m is the total number of valid merge requests; we can then bound the probability that the former happens before the latter.

We can then use a Chernoff bound to argue that after $t = O(\log \frac{k}{\epsilon}\delta_i^2)$ *repeated* merge requests, the probability that $\sum_{i=1}^{t} X_i < \delta_i$ (which must be true if we need more *repeated* merge requests) is less than ϵ/k. Therefore, the probability that we need more than $O(\log \frac{k}{\epsilon}\delta_i^2)$ *repeated* merge requests is less than ϵ/k.

By the union-bound, the probability that we need more than $O(\log \frac{k}{\epsilon}\delta_i^2)$ *repeated* merge requests for *any* ground-truth cluster C_i^* is less than $k \cdot \epsilon/k = \epsilon$. Therefore with probability at least $1-\epsilon$ for all ground-truth clusters we need $\sum_i O(\log \frac{k}{\epsilon}\delta_i^2) = O(\log \frac{k}{\epsilon}\sum_i \delta_i^2) = O(\log \frac{k}{\epsilon}\delta_u^2)$ *repeated* merge requests, where δ_u is the underclustering error of the initial clustering. Also recall that for all ground-truth clusters we need $\sum_i O(\delta_i^2) = O(\delta_u^2)$ *original* merge requests. Adding the two terms together, it follows that with probability at least $1-\epsilon$ we need a total of $O(\log \frac{k}{\epsilon}\delta_u^2)$ pure merge requests. ■

As in the previous section, we also restate the run-time bound in Theorem 22 in terms of any *natural* clustering error γ. The following corollary follows from Theorem 22 and Theorem 9.

Corollary 27 *Suppose the target clustering satisfies stability, and the initial clustering has clustering error γ, where γ is any* natural *clustering error as defined in Definition 7. In the unrestricted-merge model, assuming that each merge request is chosen uniformly at random from the set of valid merge requests, with probability at least $1-\epsilon$, the algorithms in Figure 2 and Figure 8 require $O(\log \frac{k}{\epsilon}\gamma^2)$ edit requests to find the target clustering.*

4.1 Algorithms under stronger assumptions

As in Section 3.2, if the data satisfies *strict separation*, then instead of the split procedure in Figure 2 we can use the procedure in Figure 5, which uses the local average-linkage tree (constructed from

only the points in the user request). Similarly, we can modify the merge procedure in Figure 8 to perform the split as in Figure 5. We can then prove the equivalent of Theorem 22 and Corollary 27 for strict separation for these split/merge algorithms.

5. Experimental Results

We perform two sets of experiments. We first test our proposed split algorithm on the clustering of business listings maintained by Google. We then test the proposed model in its entirety on the 20 Newsgroups data set.

5.1 Clustering business listings

Google maintains a large collection of data records representing businesses, which as of 2013 contained more than 100 millions records. These records are initially clustered using a similarity function. Each cluster is expected to contain records about the same distinct business. Clusters are then summarized; these summaries are served to users online via various front-end applications. Users report bugs such as "you are displaying the name of one business, but the address of another" (caused by overclustering), or "a particular business is shown multiple times" (caused by underclustering). The clusters involved in these requests are often quite large and usually contain records about several businesses. Therefore this setting closely matches our interactive clustering model - users provide high-level feedback requesting cluster merges/splits, where the corresponding data points intersect several ground-truth clusters.

Here we evaluate the effectiveness of our proposed split algorithm in such cases. Correctness of a split is evaluated with respect to the ground-truth labels, which are obtained through manual inspection of the listings in the cluster. For example, from manual inspection it is very easy to see that two Starbucks records with different addresses are in fact different businesses (have different ground-truth labels) even if, say, they have the same corporate phone number. But this observation is not necessarily captured by the noisy similarity function, causing the listings to incorrectly cluster together.

We consider two criteria to determine whether a split is correct or not, and show experimental results for both. We first consider a binary split correct if the two resulting sub-clusters are "clean" according to Definition 13. Note that a clean split is sufficient and necessary for reducing the under/overclustering error in our model. To compute the splits, we use the algorithm in Figure 5, which we refer to as *Clean-Split*. This algorithm is easier to implement and run than the algorithm in Figure 2 because it only requires local pairwise similarities (for the listings in the cluster). But this algorithm is still provably correct under stronger assumptions on the data (see Theorem 18 and Theorem 19), which we believe are still satisfied in this application.

We compare performance with two well-known split algorithms from split/merge clustering literature: the optimal 2-median clustering (*2-Median*), and a "sweep" of the second-smallest eigenvector of the corresponding graph Laplacian matrix. With respect to the spectral split algorithm, let $\{v_1, \ldots, v_n\}$ be the order of the vertices when sorted by their eigenvector entries. We compute the partition $\{v_1, \ldots, v_i\}$ and $\{v_{i+1}, \ldots, v_n\}$ such that its conductance is smallest (*Spectral-Balanced*), and a partition such that the similarity between v_i and v_{i+1} is smallest (*Spectral-Gap*).

We compare the split algorithms on overclustering instances that were discovered during a manual clustering quality evaluation. We consider 20 such instances to ensure that our results are consistent, given that there may be differences in the difficulty of each instance. We have anonymized

Table 1: Number of correct (clean) splits on the Google business listings data sets.

Clean-Split	2-Median	Spectral-Gap	Spectral-Balanced
19	13	12	3

these data-sets and made them publicly available. [2] The results are presented in Table 1. We observe that the *Clean-Split* algorithm works best, giving a correct split in 19 out of the 20 cases. The well-known *Spectral-Balanced* technique usually does not give correct splits in this application. The balance constraint causes it to put records about the same business on both sides of the partition (especially when all the "clean" splits are not well-balanced), which increases clustering error. As expected, the *Spectral-Gap* technique improves on this limitation (because it does not have a balance constraint), but the result is often still incorrect. The *2-Median* algorithm performs fairly well, but it is still a limited approach in this application: the optimal centers may correspond to listings about the same business, and even if they represent distinct businesses, the resulting split is still sometimes incorrect.

In addition to using the clean-split criterion, we also evaluate the computed splits using the correlation-clustering (cc) error. We find that using this criterion *Clean-Split* and *2-Median* compute the best splits, while the other two algorithms perform significantly worse. The results for *Clean-Split* and *2-Median* are presented in Table 2 - we see that *Clean-Split* is more reliable because it fails to reduce the cc-error only once, compared to 4 such failures for *2-Median*. Table 2 also reinforces our theoretical understanding of the effectiveness of our split algorithm. We know that a clean split is sufficient to reduce the cc-error, but it is not necessary. Our experiments illustrate these observations: *Clean-Split* makes progress in reducing the cc-error in 19 out of 20 cases (when the resulting split is clean), while we have also confirmed that *2-Median* sometimes still reduces the cc-error even when the resulting split is not clean.

5.2 Clustering Newsgroup documents

In order to test our entire interactive clustering model, we perform experiments on the 20 Newsgroups data set.[3] The points in the data set are posts to twenty different online forums (called newsgroups). We sample these data to generate 5 data sets of manageable size (containing 276-301 points), which are labeled A through E in the figures. Each data set contains some documents from every newsgroup. The purpose of these 5 data sets is to check the consistency of our results, given that there may be considerable variation in how separable the ground-truth clusters are in each case. For each data set, the ground-truth is determined by the newsgroups that the documents belong to.

Each Newsgroup document is represented by a term frequency - inverse document frequency (tf-idf) vector (Salton and Buckley, 1988). We use cosine similarity to compare these vectors, which gives a similarity score in $[0, 1]$. We compute an initial clustering by using the following procedure to perturb the ground-truth: for each document we keep its ground-truth cluster assignment with probability 0.5, and otherwise reassign it to one of the other clusters, which is chosen uniformly at random. This procedure generates initial clusterings with overclustering error of about 100, underclustering error of about 100, and correlation-clustering error of about 5000.

2. The anonymized Google business listings data sets are available here.
3. http://people.csail.mit.edu/jrennie/20Newsgroups/

Table 2: Change in correlation-clustering error for splits on the Google business listings data sets. A negative value corresponds to a decrease in error.

Data Set	Clean-Split	2-Median
1	-14	-14
2	-5	-5
3	-11	-11
4	-117	-117
5	-42	+90
6	-4	-4
7	-12	-30
8	-27	-27
9	-6	-6
10	-6	-6
11	+6	-8
12	-10	+14
13	-6	-6
14	-12	-22
15	-6	-6
16	-10	+14
17	-11	-27
18	-10	-10
19	-11	-5
20	-10	-10

To simulate user feedback, we compute the set of all valid split and merge requests. Consistent with our model, a split of a cluster is considered valid if it contains points from 2 or more ground-truth clusters, and a merge is considered valid if at least an η- fraction of the points in each cluster are from the same ground-truth cluster. We then choose one of the valid splits/merges uniformly at random, and ask the algorithm to compute the corresponding split/merge. We continue to iterate until we find the ground-truth clustering or reach a limit on the number of edit requests (set to 20000).

The Newsgroup data is know to be very challenging for unsupervised (Telgarsky and Dasgupta, 2012; Heller and Ghahramani, 2005) and semi-supervised clustering algorithms (Basu et al., 2002, 2004). In particular, on harder Newsgroup clustering instances Basu et al. (2002) and Basu et al. (2004) report that they are unable to come close to finding the ground-truth clustering even with considerable supervision.

To study how our algorithms perform on easier clustering instances, we also slightly prune our data sets. Our heuristic iteratively finds and removes the outlier in each target cluster, where the outlier is the point with minimum sum-similarity to the other points in the target cluster. For each

data set, we perform experiments with the original (unpruned) data set, a pruned data set with 2 points removed per target cluster, and a pruned data set with 4 points removed per target cluster, which prunes 40 and 80 points, respectively. Our pruning heuristic generates clustering instances where the ground-truth clusters are more separable, similar to the Different-3 data set of Basu et al. (2002) and the News-diff3 data set of Basu et al. (2004).

5.2.1 EXPERIMENTS IN THE η-MERGE MODEL

We first experiment with algorithms in the η-merge model. Here we use the algorithm in Figure 2 to perform the splits, and the algorithm in Figure 3 to perform the merges. We show the results for data set A in Figure 9; the complete experimental results are in Appendix B. We find that for larger settings of η, the amount of supervision (number of split/merge requests needed to find the target clustering) is quite low and consistent with our theoretical analysis. The results are better for pruned data sets, where we get very good performance regardless of the setting of η. The results for the algorithms in Figure 2 and Figure 4 (for the correlation-clustering objective) are very favorable as well.

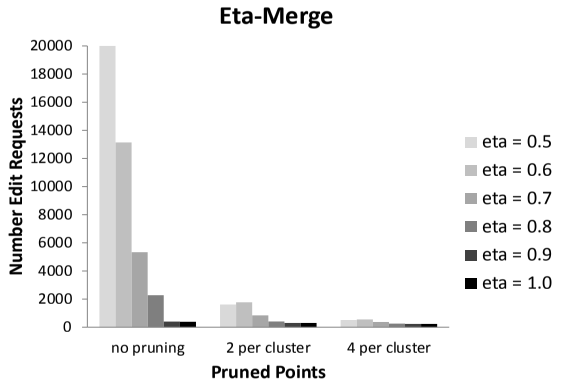
5.2.2 EXPERIMENTS IN THE UNRESTRICTED-MERGE MODEL

We also experiment with algorithms in the unrestricted merge model. Here we use the same algorithm to perform the splits, and use the algorithm in Figure 8 to perform the merges. We show the results on data set A in Figure 10; the complete experimental results are in Appendix B. We find that again for larger settings of η the amount of supervision is quite low and consistent with our theoretic analysis (we only show results for $\eta \geq 0.5$), and performance improves further for pruned data sets. Our investigations show that for unpruned data sets and smaller settings of η, we are still able to quickly achieve very small clustering error (results not shown), but the algorithms are unable to converge to the ground-truth clustering due to inconsistencies in the average-linkage tree. We can address some of these inconsistencies by constructing the tree in a more robust way, which indeed gives improved performance (see Appendix C).

5.2.3 EXPERIMENTS WITH SMALL INITIAL ERROR

We also consider a setting where the initial clustering is already very accurate. In order to simulate this scenario, when we compute the initial clustering, for each document we keep its ground-truth cluster assignment with probability 0.95, and otherwise reassign it to one of the other clusters, which is chosen uniformly at random. This procedure usually gives us initial clusterings with over-clustering and underclustering error between 5 and 20, and correlation-clustering error between 500 and 1000. Note that this setting also closely matches the scenario where the initial clustering is computed using a good clustering algorithm, which is quite realistic in some applications.

As expected, in this setting our interactive algorithms perform much better, especially on pruned data sets. Figure 11 displays the results; we can see that in these scenarios it often takes less than one hundred edit requests to find the target clustering in both models.



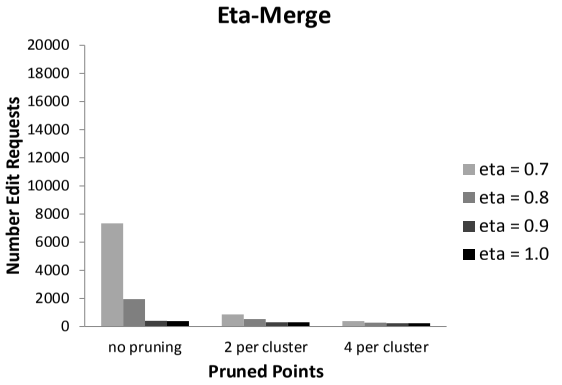


Figure 9: Experimental results in the η-merge model for data set A. The second chart corresponds to algorithms for correlation clustering error.

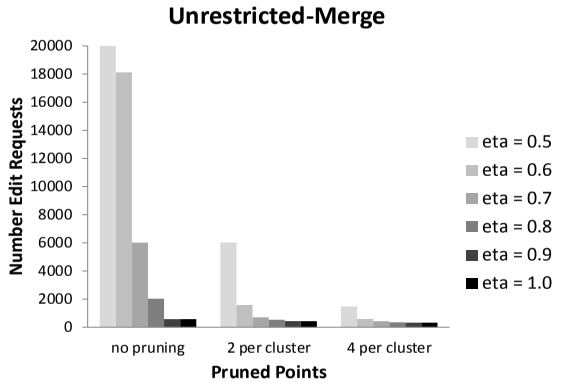


Figure 10: Experimental results in the unrestricted merge model for data set A.

5.2.4 COMPARISON WITH OTHER SEMI-SUPERVISED CLUSTERING MODELS

In this section we give a performance comparison on the 20 Newsgroups data sets with the semi-supervised clustering models of Basu et al. (2002) and Basu et al. (2004). The accuracy of the output clustering is stated in terms of *normalized mutual information* (NMI) (see Strehl et al., 2000).

Table 3 shows a comparison on a harder Newsgroup clustering instance; table 4 shows a comparison on an easier instance. The performance of our approach is with respect to the η-merge model, using the algorithm in Figure 2 to perform the splits, and the algorithm in Figure 3 to perform the merges; we focus on a single setting of $\eta = 0.7$. To compare the amount of supervision used by each model, consider that n instance class labels in the framework of Basu et al. (2002) are equivalent to $n(n-1)/2$ pairwise labels in the framework of Basu et al. (2004). Let us also oversimplify and assume that the cost of a pairwise label in the framework of Basu et al. (2004) is the same as the cost of one/split merge request in our framework. Also, we can adjust for differences in data set size by assuming that the proportion of supervision does not change. In particular, 1500 instance class labels on a data set of 3000 documents correspond to 150 instance class labels on a data set of 301 documents. These 150 instance class labels are then equivalent to 150 * 149 / 2 = 11175 pairwise labels. Similarly, 1500 instance class labels on a data set of 3000 documents correspond to 138 instance class labels on a data set of 276 documents, which are then equivalent to 138 * 137 / 2 = 9453 pairwise labels.

Then for a harder clustering instance we observe that we need less supervision than Basu et al. (2002) and roughly 7.5 times more supervision than Basu et al. (2004) to achieve a completely correct clustering (NMI = 1.0) as opposed to a clustering that is only half-correct (NMI = 0.44, 0.55). For an easier clustering instance, we observe that we need roughly 2.7 times less supervision than

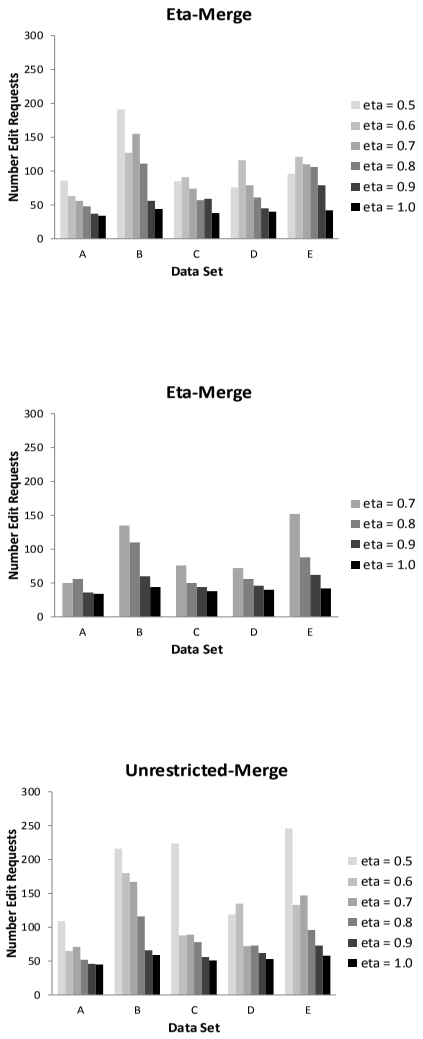


Figure 11: Experimental results for initial clusterings with small error. Results presented for pruned data sets (4 points per cluster). The second chart corresponds to algorithms for correlation clustering error.

Basu et al. (2002), and 3.9 times more supervision than Basu et al. (2004) to achieve a completely correct clustering (NMI = 1.0) as opposed to a mostly-correct clustering (NMI = 0.88).

These are favorable comparisons for our model - we are able to compute more accurate clusterings while using a comparable amount of supervision. We also note that for Basu et al. (2002) and Basu et al. (2004), additional supervision does not improve accuracy - the stated performance is the best accuracy they can achieve. Our framework is the only approach that has demonstrated a complete recovery of the ground truth on the Newsgroup data.

Table 3: Framework comparison on harder Newsgroup clustering instance.

Framework	Data Set Name	Data Set Size	Amount of Supervision	Output Accuracy
Basu et al. (2002)	Same-3	3000	1500 labeled data points	NMI = 0.44
Basu et al. (2004)	News-sim3	300	1000 pairwise constraints	NMI = 0.55
Interactive clustering	Data Set E	301	7573 split/merge requests	NMI = 1.0

Table 4: Framework comparison on easier Newsgroup clustering instance.

Framework	Data Set Name	Data Set Size	Amount of Supervision	Output Accuracy
Basu et al. (2002)	Different-3	3000	1500 labeled data points	NMI = 0.88
Basu et al. (2004)	News-diff3	300	1000 pairwise constraints	NMI = 0.88
Interactive clustering	Data Set D	276	3548 split/merge requests	NMI = 1.0

6. Discussion

In this work we motivated and studied a new framework and algorithms for interactive clustering. Our framework models practical constraints on the algorithms: we start with an initial clustering that we cannot modify arbitrarily, and are only allowed to make local edits consistent with user requests. In this setting, we develop several simple, yet effective algorithms under different assumptions about the nature of the edit requests and the structure of the data. We present theoretical analysis that shows that our algorithms converge to the target clustering after a limited number of edit requests. We also present experimental evidence that shows that our algorithms work well in practice.

Several directions come out of this work. It would be interesting to relax the condition on η in the η-merge model, and the assumption about the request sequences in the unrestricted-merge model. It is also important to study additional properties of an interactive clustering algorithm. In particular, it is often desirable that the algorithm never increase the error of the current clustering. Our algorithms in Figures 2, 4 and 8 have this property, but the algorithm in Figure 3 does not.

Other user feedback models have also been proposed (see Appendix A for how they relate to the split/merge feedback considered here). It is valuable to further study the practical applicability of each kind of feedback and understand under what conditions they are equivalent.

7. Acknowledgments

This work was supported in part by grants NSF CCF-0953192, NSF-CCF 1535967, NSF CCF-1422910, NSF IIS1618714, ONR grant N00014-09-1-0751, AFOSR grant FA9550-09-1-0538, a Google Research Award, and a Microsoft Research Fellowship.

References

Dimitris Achlioptas and Frank McSherry. On spectral learning of mixtures of distributions. In *COLT*, 2005.

Margareta Ackerman and Sajoy Dasgupta. Incremental clustering: The case for extra clusters. In *NIPS*, 2014.

Margareta Ackerman, Shai Ben-David, Simina Brânzei, and David Loker. Weighted clustering. In *AAAI*, 2012.

Dana Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1998.

Sanjeev Arora and Ravi Kannan. Learning mixtures of arbitrary gaussians. In *STOC*, 2001.

Hassan Ashtiani, Shrinu Kushagra, and Shai Ben-David. Clustering with same-cluster queries. *CoRR*, abs/1606.02404, 2016.

Pranjal Awasthi and Maria-Florina Balcan. Center based clustering: A foundational perspective. In Christian Hennig, Marina Meila, Fionn Murtagh, and Roberto Rocci, editors, *Handbook of Cluster Analysis*. CRC Press, 2015.

Pranjal Awasthi and Reza Bosagh Zadeh. Supervised clustering. In *NIPS*, 2010.

Maria-Florina Balcan and Avrim Blum. Clustering with interactive feedback. In *ALT*, 2008.

Maria-Florina Balcan and Pramod Gupta. Robust hierarchical clustering. In *COLT*, 2010.

Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. A discriminative framework for clustering via similarity functions. In *STOC*, 2008.

Maria-Florina Balcan, Yingyu Liang, and Pramod Gupta. Robust hierarchical clustering. *Journal of Machine Learning Research*, 15:3831–3871, 2014.

Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56 (1-3):89–113, 2004.

Sugato Basu, Arindam Banerjee, and Raymond Mooney. Semi-supervised clustering by seeding. In *ICML*, 2002.

Sugato Basu, Arindam Banerjee, and Raymond J. Mooney. Active semi-supervision for pairwise constrained clustering. In *SDM*, 2004.

Mikhail Belkin and Kaushik Sinha. Polynomial learning of distribution families. In *FOCS*, 2010.

Constantinos Boulis and Mari Ostendorf. Combining multiple clustering systems. In *PKDD*, 2004.

S. Charles Brubaker and Santosh Vempala. Isotropic PCA and affine-invariant clustering. *CoRR*, abs/0804.3575, 2008.

David Bryant and Vincent Berry. A structured family of clustering and tree construction methods. *Advances in Applied Mathematics*, 27(4):705–732, 2001.

D. Chaudhuri, B. B. Chaudhuri, and C. A. Murthy. A new split-and-merge clustering technique. *Pattern Recognition Letters*, 13(6):399–409, 1992.

Kamalika Chaudhuri and Sanjoy Dasgupta. Rates of convergence for the cluster tree. In *NIPS*, 2010.

Bo Dai, Baogang Hu, and Gang Niu. Bayesian maximum margin clustering. In *ICDM*, 2010.

Sanjoy Dasgupta. Learning mixtures of gaussians. In *FOCS*, 1999.

Sanjoy Dasgupta and Daniel Hsu. Hierarchical sampling for active learning. In *ICML*, 2008.

Chris Ding and Xiaofeng He. Cluster merging and splitting in hierarchical clustering algorithms. In *ICDM*, 2002.

Brian Erikkson, Gautam Dasarathy, Aarti Singh, and Robert Nowak. Active clustering: robust and efficient hierarchical clustering using adaptively selected similarities. In *AISTATS*, 2011.

Katherine A. Heller and Zoubin Ghahramani. Bayesian hierarchical clustering. In *ICML*, 2005.

Adam Tauman Kalai, Ankur Moitra, and Gregory Valiant. Efficiently learning mixtures of two Gaussians. In *STOC*, 2010.

Ravindran Kannan, Hadi Salmasian, and Santosh Vempala. The spectral method for general mixture models. In *COLT*, 2005.

Akshay Krishnamurthy, Sivaraman Balakrishnan, Min Xu, and Aarti Singh. Efficient active algorithms for hierarchical clustering. In *ICML*, 2012.

Juho Lee, Suha Kwak, Bohyung Han, and Seungjin Choi. Online video segmentation by bayesian split-merge clustering. In *ECCV*, 2012.

Marina Meilă. Comparing clusterings - an information based distance. *Journal of Multivariate Analysis*, 98(5):873–895, 2007.

Ankur Moitra and Gregory Valiant. Settling the polynomial learnability of mixtures of gaussians. In *FOCS*, 2010.

Feiping Nie, Dong Xu, and Xuelong Li. Initialization independent clustering with actively self-training method. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 42(1):17–27, 2012.

Alessandro Rinaldo and Larry Wasserman. Generalized density clustering. *The Annals of Statistics*, 38(5):2678–2722, 2010.

Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing and management*, 24(5):513–523, 1988.

Alexander Strehl, Joydeep Ghosh, and Raymond Mooney. Impact of similarity measures on web-page clustering. In *AAAI*, 2000.

Matus Telgarsky and Sanjoy Dasgupta. Agglomerative bregman clustering. In *ICML*, 2012.

Konstantin Voevodski, Maria-Florina Balcan, Heiko Röglin, Shang-Hua Teng, and Yu Xia. Active clustering of biological sequences. *Journal of Machine Learning Research*, 13:203–225, 2012.

Qiaoliang Xiang, Qi Mao, Kian Ming A. Chai, Hai Leong Chieu, Ivor Wai-Hung Tsang, and Zhendong Zhao. A split-merge framework for comparing clusterings. In *ICML*, 2012.

Shi Zhong. Generative model-based document clustering: a comparative study. *Knowledge and Information Systems*, 8:374–384, 2005.

Appendix A. Other feedback models

Besides the cluster split/merge feedback considered here, two other kinds of feedback have also been studied in the literature: *cluster-assignment* feedback (Basu et al., 2002; Nie et al., 2012) and *must-link/cannot-link* feedback (Basu et al., 2004; Ashtiani et al., 2016). The former reveals the ground-truth cluster assignment of a single data point; the latter reveals whether or not a pair of points are in the same ground-truth cluster.

Claim 28 *Cluster-assignment feedback and must-link/cannot link feedback can be converted to cluster split/merge feedback in the unrestricted-merge model.*

Proof We first observe that cluster-assignment feedback can be converted to must-link/cannot-link feedback as follows. For every two known cluster assignments for points x and y s.t. x and y belong to the same ground-truth cluster, output a must-link feedback for (x, y). For every two known cluster assignments for points x and y s.t. x and y belong to different ground-truth clusters, output a cannot-link feedback for (x, y).

We next observe that must-link/cannot-link feedback can be converted to cluster split/merge feedback in the unrestricted merge model as follows. For every must-link feedback for points x and y, if x and y belong to different clusters in the proposed clustering, request to merge these two clusters. For every cannot-link feedback for points x and y, if x and y belong to the same cluster in the proposed clustering, request to split this cluster. Observe that by definition, the output split/merge requests satisfy the assumptions of the unrestricted-merge model. ■

However, the conversion in Claim 28 is sometimes vacuous: cluster split/merge feedback is only output when the cluster-assignment or must-link/cannot-link feedback disagrees with the current clustering. Still, it is reasonable to assume that while the proposed clustering is not equivalent to the ground truth, a constant fraction of such feedback will disagree with the proposed clustering.

Appendix B. Complete experimental results

The following figures show the complete experimental results for all the algorithms. Figure 12 shows the results in the η-merge model. Figure 13 shows the results in the η-merge model for the algorithms in Figure 2 and Figure 4 (for the correlation-clustering objective). Figure 14 shows the results in the unrestricted-merge model.

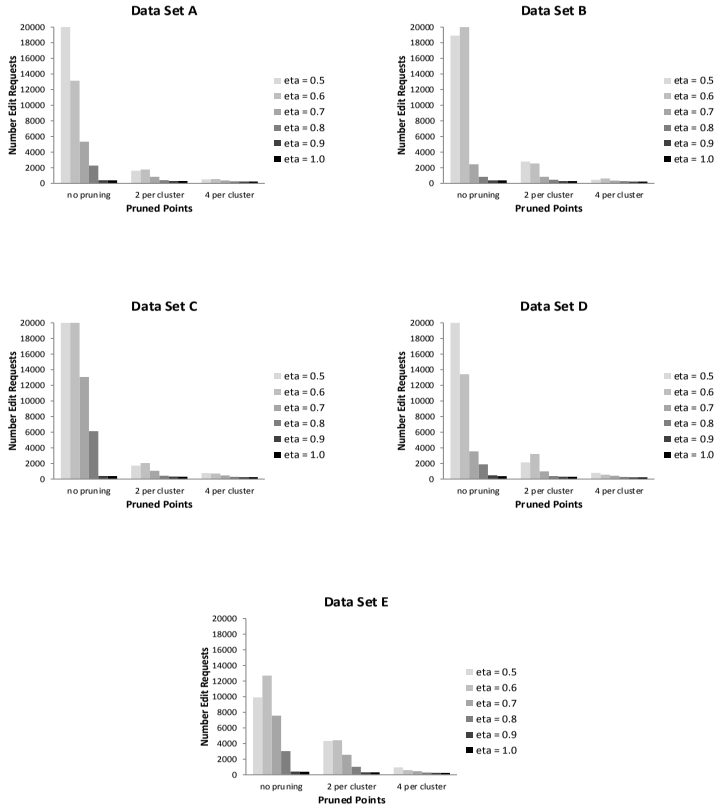


Figure 12: Experimental results in the η-merge model.

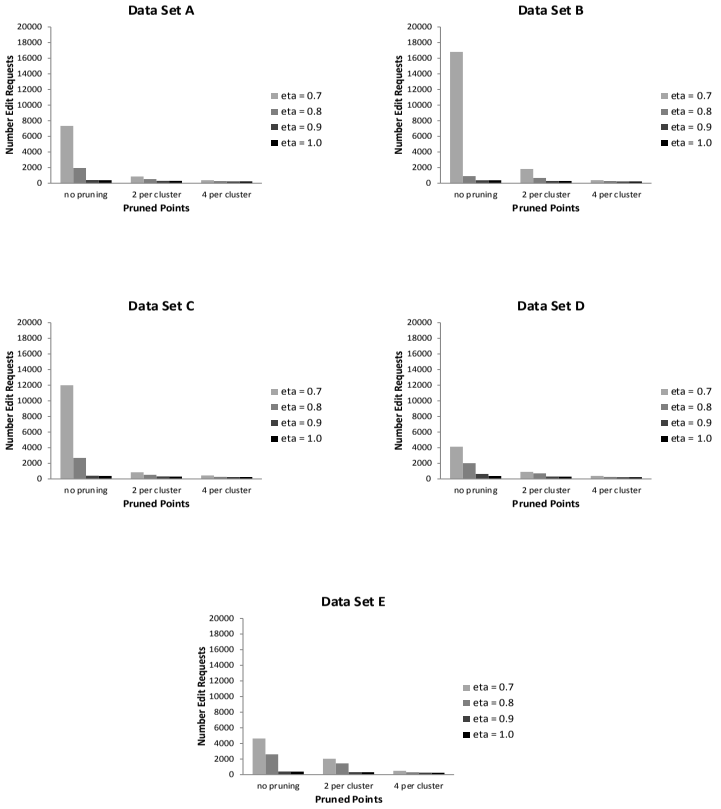


Figure 13: Experimental results in the η-merge model for algorithms for correlation-clustering objective.

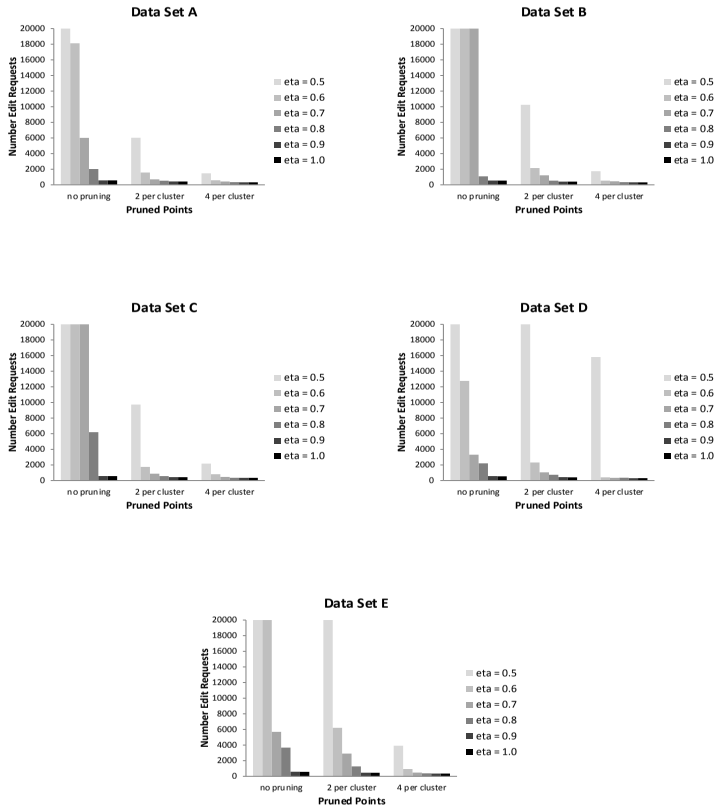


Figure 14: Experimental results in the unrestricted-merge model.

Appendix C. Experiments with robust average-linkage tree

When we investigate instances where our algorithms are unable to find the target clustering, we observe that there are outlier points that are attached near the root of the average-linkage tree, which are incorrectly split off and re-merged by the algorithm without making any progress towards finding the target clustering.

We can address these outliers by constructing the average-linkage tree in a more robust way: first find groups ("blobs") of similar points of some minimum size, compute an average-linkage tree for each group, and then combine these trees using average-linkage. The tree constructed in such fashion may then be used by our algorithms.

We tried this approach, using Algorithm 2 from Balcan and Gupta (2010) to compute the blobs. We find that using the robust average-linkage tree gives better performance for the unpruned data sets, but gives no gains for the pruned data sets, as expected. Figure 15 displays the comparison in the unrestricted merge model for the five unpruned data sets. For the pruned data sets, we expect the robust tree to be very similar to the standard tree, which explains why there is little difference in performance (results not shown).

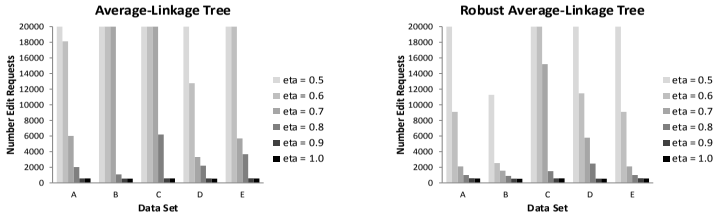


Figure 15: Experimental results in the unrestricted-merge model using a standard versus robust average-linkage tree. Results presented for unpruned data sets.

Journal of Machine Learning Research 18 (2017) 1-5 Submitted 9/15; Revised 10/16; Published 2/17

SnapVX: A Network-Based Convex Optimization Solver

David Hallac* HALLAC@STANFORD.EDU
Christopher Wong† CRWONG@CS.STANFORD.EDU
Steven Diamond† DIAMOND@CS.STANFORD.EDU
Abhijit Sharang† ABHISG@CS.STANFORD.EDU
Rok Sosič† ROK@CS.STANFORD.EDU
Stephen Boyd* BOYD@STANFORD.EDU
Jure Leskovec† JURE@CS.STANFORD.EDU
*Department of Electrical Engineering**
Department of Computer Science†
Stanford University, Stanford, CA, 94305

Editor: Cheng Soon Ong

Abstract

SnapVX is a high-performance solver for convex optimization problems defined on networks. For problems of this form, SnapVX provides a fast and scalable solution with guaranteed global convergence. It combines the capabilities of two open source software packages: Snap.py and CVXPY. Snap.py is a large scale graph processing library, and CVXPY provides a general modeling framework for small-scale subproblems. SnapVX offers a customizable yet easy-to-use Python interface with "out-of-the-box" functionality. Based on the Alternating Direction Method of Multipliers (ADMM), it is able to efficiently store, analyze, parallelize, and solve large optimization problems from a variety of different applications. Documentation, examples, and more can be found on the SnapVX website at `http://snap.stanford.edu/snapvx`.

Keywords: convex optimization, network analytics, graphs, data mining, ADMM

1. Introduction

Convex optimization is a widely used approach of modeling and solving problems in many different fields, as it offers well-established methods for finding globally optimal solutions. Numerous general-purpose optimization software packages exist (Sturm, 1999; Byrd et al., 2006; Mosek, 2010; Diamond and Boyd, 2016), but they typically rely on algorithms that are difficult to scale, so many modern machine learning problems cannot be solved by these common, yet general, approaches. Instead, solving large scale examples often requires developing problem-specific solution methods, which can be very fast but require significant optimization expertise to build. Furthermore, they are limited in scope, as they must be fine-tuned to only one particular type of problem and thus are hard to generalize.

In this paper, we build on the observation that many large convex optimization examples follow a common form in that they can often be split up into a series of subproblems using a network (graph) structure. Nodes are subproblems, representing anything from timestamps in a time-series data set to users in a social network. The edges then define the coupling, or relationships between the different nodes, and the combination of nodes and edges yields the original convex optimization problem. This representation can refer to problems defined on actual networks, such as social or transportation systems, or questions classically modeled in other ways, such as control theory or time-series analysis.

Here, we present SnapVX, a solver that is both scalable and general on optimization problems defined over networks. It combines the graph capabilities of Snap.py (Leskovec and Sosič, 2016) with the general modeling framework from CVXPY (Diamond and Boyd, 2016). We show how SnapVX works, present syntax and supported features, scale it to large problems, and describe how it can be used to solve convex optimization problems from a variety of different fields. The full version of our solver, which is released under the BSD Open-Source License, can be found at the project website, `http://snap.stanford.edu/snapvx`. In addition to the source code, the download contains installation instructions, unit tests, documentation, and several examples to help users get started.

2. SnapVX General Form

Consider an optimization problem on an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$, of the form:

$$\underset{x}{\text{minimize}} \quad \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{(j,k) \in \mathcal{E}} g_{jk}(x_j, x_k). \tag{1}$$

Variable x_i is associated with node i, for $i = 1, \ldots, |\mathcal{V}|$ (the size of x_i can vary at each node). In problem (1), f_i is the cost function at node i, and g_{jk} is the cost associated with edge (j, k). Constraints are represented by extended (infinite) values of f_i and g_{jk}. Note that nodes and edges can also have local, private optimization variables (which can also vary in size). These remain confined to a single node or edge, though, whereas the x_i's are shared between different parts of the network. We consider only convex objectives and constraints for f_i and g_{jk}, so the entire problem is a convex optimization problem.

To solve problem (1), SnapVX uses a splitting algorithm based on the Alternating Direction Method of Multipliers (ADMM) (Boyd et al., 2011). With ADMM, each individual component of the graph solves its own subproblem, iteratively passing small messages over the network and eventually converging to an optimal solution. See the SnapVX developer documentation for more details on the underlying derivation. By splitting up the problem into a series of subproblems, SnapVX is able to solve much larger convex optimization examples than standard solvers, which solve the whole problem at once. ADMM also keeps the optimality benefits that general solvers enjoy: not only are we guaranteed to obtain an optimal solution, but we also are given a certificate of optimality in the form of primal and dual residuals; see (Parikh and Boyd, 2014).

SnapVX stores the network using a Snap.py graph structure. This allows fast traversal and easy manipulation, as well as efficient storage. On top of the standard graph, each node/edge is given convex objectives and constraints using CVXPY syntax. To find a global solution, SnapVX automatically splits up the problem and solves each subproblem using CVXPY, iteratively handling the ADMM message passing behind the scenes. Though wrapped in a Python layer, CVXPY uses ECOS and CVXOPT (two high-performance numerical optimization packages) as its primary underlying solvers (Andersen et al., 2014; Domahidi et al., 2013), so the Python overhead is not significant and allows for easier interpretability and improved user interface. To parallelize the solver, a worker pool coordinates

©2017 David Hallac, Christopher Wong, Steven Diamond, Abhijit Sharang, Rok Sosič, Stephen Boyd, and Jure Leskovec.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v18/15-492.html`.

updates for the separate subproblems using Python's multiprocessing library. SnapVX is built to run on a single machine, parallelizing across multiple cores and allowing "out-of-the-box" functionality on machines ranging from standard laptops to large-memory servers.

3. Syntax and Supported Features

We now present usage of SnapVX on a simple example. Complete documentation and more examples are available on the SnapVX website. Consider two nodes with an edge between them. We solve for a problem where each node has an unknown variable $x_i \in \mathbb{R}^1$. The first node's objective is to minimize x_1^2 subject to $x_1 \leq 0$, the second's is to minimize $|x_2 + 3|$, and the edge objective penalizes the square norm difference between the two variables, $\|x_1 - x_2\|_2^2$. The following code specifies the optimization problem and solves it:

```
from snapvx import *
gvx = TGraphVX() #Create a new graph
x1 = Variable(1, name='x1') #Create a variable for node 1
gvx.AddNode(1, Objective=square(x1), Constraints=[x1 <= 0]) #Add new node
x2 = Variable(1, name='x2') #Repeat for node 2
gvx.AddNode(2, abs(x2 + 3), [])
gvx.AddEdge(1, 2, Objective=square(norm(x1 - x2)), Constraints=[]) #Add edge between nodes
gvx.Solve() #Solve the problem
gvx.PrintSolution() #Print the solution on a node-by-node basis
```

As SnapVX is meant to be a general-purpose solver, it has many customizable options to help easily and efficiently solve a wide range of convex optimization problems. These include ADMM iteration limits, verbose mode (to list intermediate steps at each iteration), and defining customized convergence thresholds. Two key features are highlighted below:

- **Bulk Loading -** Often, the objectives and constraints at each node or edge will share a common form. For example, all the nodes could be trying to minimize $\|x - a_i\|_2$ for different values of a_i. Rather than requiring users to manually input each of these values, SnapVX allows for "bulk loading" of data. The functions *AddNodeObjectives* and *AddEdgeObjectives* allow the user to specify the general form of the objectives and an external file with separate data, and SnapVX fills in the details. This functionality makes practical and significantly speeds up the loading of very large data sets.
- **ADMM ρ-update -** The convergence time of ADMM depends on the value of the penalty parameter ρ (Nishihara et al., 2015), as it affects the tradeoff between primal and dual convergence, both of which need to be obtained for the overall problem to be solved. SnapVX users are not only able to select the value of ρ (it defaults to $\rho = 1$), but can also define a function to update ρ after each iteration based on the primal and dual residual values (He et al., 2000; Fougner and Boyd, 2015).

4. Scalability

One of the biggest benefits of SnapVX is that it allows us to solve large problems very efficiently. It does so by automatically parallelizing the ADMM updates across multiple cores of a single machine using Python's multiprocessing class. Note that SnapVX is meant to be run on a single machine, rather than in a distributed computing environment. Convergence time depends on the problem complexity, but we empirically observe that it scales

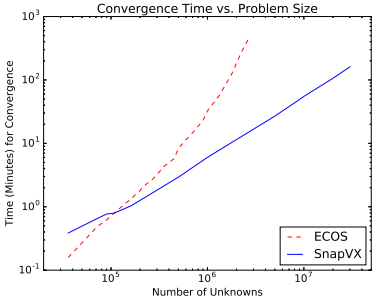


Figure 1: Timing comparison of SnapVX and a general-purpose solver.

# of Unknowns	SnapVX Solution Time (Minutes)
100,000	0.79
500,000	2.95
1 million	6.06
5 million	27.30
10 million	55.42
20 million	106.33
30 million	161.95

Table 1: SnapVX convergence time for several different problem sizes.

approximately linearly with problem size. We compare SnapVX and a general solver for a problem on a 3-regular graph. Each node solves for an unknown variable in $\mathbb{R}^{9000}$, where the node objectives are sum-of-Huber penalties (Huber, 1964) and edges have network lasso penalties (Hallac et al., 2015). By varying the number of nodes, we can span a wide range of problem sizes. The time required for SnapVX to converge, on a 40-core CPU where the entire problem can fit into memory, is shown in Table 1. Figure 1 displays a comparison to a general-purpose solver (ECOS, via CVXPY) in log-log scale.

5. Applications

SnapVX was created to allow users without significant optimization expertise to solve large network-based convex optimization problems by taking advantage of the powerful and scalable ADMM algorithm. Common examples from many different fields can be formulated in a SnapVX-friendly manner. Our software has been used to solve real-world machine learning problems ranging from housing price prediction (Hallac et al., 2015) to ride-sharing analysis (Ghosh et al., 2016) to high-dimensional regression (Yamada et al., 2016), often significantly outperforming other commonly used approaches. Examples in the SnapVX download package and on the project website — including financial modeling, network inference, image processing, PageRank, and time-series analysis — guide users towards applying the software to a variety of applications. Both user and developer documentation help new users get started and, if they are interested, contribute to the code base. A robust set of unit tests ensures that the software has been properly downloaded and that the code is functioning as expected. Overall, our open-source software provides an easy-to-use ADMM solver that can scale to large problems and apply to a wide variety of examples. With an active user base and rising interest from a range of scientific and engineering fields, we hope that SnapVX can become a useful tool for convex optimization problems, both in research and industry.

Acknowledgments

This work was supported by NSF IIS-1149837, NIH BD2K, DARPA XDATA, DARPA SIMPLEX, Stanford Data Science Initiative, Boeing, Bosch, Lightspeed, SAP, and Volkswagen.

References

M. Andersen, J. Dahl, and L. Vandenberghe. CVXOPT: A Python package for convex optimization, version 1.1.7. `http://cvxopt.org/`, 2014.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3:1–122, 2011.

R. Byrd, J. Nocedal, and R. Waltz. KNITRO: An integrated package for nonlinear optimization. In *Large-scale nonlinear optimization*, pages 35–59. Springer, 2006.

S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

A. Domahidi, E. Chu, and S. Boyd. ECOS: An SOCP solver for embedded systems. In *European Control Conference*, pages 3071–3076, 2013.

C. Fougner and S. Boyd. Parameter selection and pre-conditioning for a graph form solver. *arXiv preprint arXiv:1503.08366*, 2015.

S. Ghosh, K. Page, and D. De Roure. An application of network lasso optimization for ride sharing prediction. *arXiv preprint arXiv:1606.03276*, 2016.

D. Hallac, J. Leskovec, and S. Boyd. Network lasso: Clustering and optimization in large graphs. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 387–396, 2015.

B. He, H. Yang, and S. Wang. Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. *Journal of Optimization Theory and Applications*, 106(2):337–356, 2000.

P. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, 1964.

J. Leskovec and R. Sosič. SNAP: A general purpose network analysis and graph mining library. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(1):1, 2016.

ApS Mosek. The MOSEK optimization software. *Online at http://www. mosek. com*, 2010.

R. Nishihara, L. Lessard, B. Recht, A. Packard, and M. Jordan. A general analysis of the convergence of ADMM. In *International Conference on Machine Learning*, pages 343–352, 2015.

N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1: 123–231, 2014.

J. Sturm. Using SeDuMi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization methods and software*, 11(1-4):625–653, 1999.

M. Yamada, K. Takeuchi, T. Iwata, J. Shawe-Taylor, and S. Kaski. Localized lasso for high-dimensional regression. *arXiv preprint arXiv:1603.06743*, 2016.

Journal of Machine Learning Research 18 (2017) 1-30 Submitted 1/16; Revised 2/16; Published 1/17

Communication-efficient Sparse Regression

Jason D. Lee JASONLEE@MARSHALL.USC.EDU
Marshall School of Business
University of Southern California
Los Angeles, CA 90089

Qiang Liu QLIU@CS.DARTMOUTH.EDU
Department of Computer Science
Dartmouth University
Hanover, NH 02714

Yuekai Sun YUEKAI@UMICH.EDU
Department of Statistics
University of Michigan
Ann Arbor, MI 48109

Jonathan E. Taylor JONATHAN.TAYLOR@STANFORD.EDU
Department of Statistics
Stanford University
Stanford, CA 94305

Editor: Zhihua Zhang

Abstract

We devise a communication-efficient approach to distributed sparse regression in the high-dimensional setting. The key idea is to average "debiased" or "desparsified" lasso estimators. We show the approach converges at the same rate as the lasso as long as the dataset is not split across too many machines, and consistently estimates the support under weaker conditions than the lasso. On the computational side, we propose a new parallel and computationally-efficient algorithm to compute the approximate inverse covariance required in the debiasing approach, when the dataset is split across samples. We further extend the approach to generalized linear models.

Keywords: Distributed Sparse Regression, Averaging, Debiasing, lasso, high-dimensional statistics

1. Introduction

Explosive growth in the size of modern datasets has fueled interest in distributed statistical learning. For examples, we refer to Boyd et al. (2011); Dekel et al. (2012); Duchi et al. (2012); Zhang et al. (2013) and the references therein. The problem arises, for example, when working with datasets that are too large to fit on a single machine and must be distributed across multiple machines. The main bottleneck in the distributed setting is usually communication between machines/processors, so the overarching goal of algorithm design is to minimize communication costs.

©2017 Lee, Liu, Sun, and Taylor.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-002.html.

In distributed statistical learning, the simplest and most popular approach is *averaging*: each machine forms a local estimator $\hat{\theta}_k$ with the portion of the data stored locally, and a "master" averages the local estimators to produce an aggregate estimator: $\bar{\theta} = \frac{1}{m}\sum_{k=1}^{m} \hat{\theta}_k$. Averaging was first studied by Mcdonald et al. (2009) for multinomial regression. They derive non-asymptotic error bounds on the estimation error that show averaging reduces the variance of the local estimators, but has no effect on the bias (from the centralized solution). In follow-up work, Zinkevich et al. (2010) studied a variant of averaging where each machine computes a local estimator with stochastic gradient descent (SGD) on a random subset of the dataset. They show, among other things, that their estimator converges to the centralized estimator.

More recently, Zhang et al. (2013) studied averaged empirical risk minimization (ERM). They show that the mean squared error (MSE) of the averaged ERM decays like $O\big(N^{-\frac{1}{2}} + \frac{m}{N}\big)$, where m is the number of machines and N is the total number of samples. Thus, so long as $m \lesssim \sqrt{N}$, the averaged ERM matches the $N^{-\frac{1}{2}}$ convergence rate of the centralized ERM. Even more recently, Rosenblatt and Nadler (2014) studied the optimality of averaged ERM in two asymptotic settings: $N \to \infty$, m, p fixed and $p, n \to \infty$, $\frac{p}{n} \to \mu_l \in (0, 1)$, where $n = \frac{N}{m}$ is the number of samples per machine. They show that in the $n \to \infty$, p fixed setting, the averaged ERM is first-order equivalent to the centralized ERM. However, when $p, n \to \infty$, the averaged ERM is suboptimal (versus the centralized ERM).

We develop a divide and conquer approach to statistical learning. In the high-dimensional setting, regularization is essential. The key idea is to average *debiased* or *desparsified* regularized M-estimators. Under suitable conditions, it is possible to show that the local debiased estimators are asymptotically normal. thus the averaged estimator delivers the same statistical performance as the computationally infeasible centralized M-estimator.

Formally, we show that the error of the averaged estimator decomposes into a $\widetilde{O}_P\big(\frac{1}{\sqrt{N}}\big)$ asymptotically normal term and a remainder term. As long as $m \lesssim \frac{\sqrt{N}}{s \log p}$, where s is the sparsity of the unknown regression coefficients, the reminder term is asymptotically negligible. Thus the averaged estimator converges at the same rate as a centralized estimator. Further, the averaged estimator is model selection consistent under a weak minimum signal strength condition. In the following section, we review the theoretical properties of the lasso and debiased lasso and describe our contributions more formally.

2. A divide-and-conquer approach to sparse regression

To keep things simple, we focus on sparse linear regression. Consider the sparse linear model

$$y = X\beta^* + \epsilon,$$

where the rows of $X \in \mathbf{R}^{n \times p}$ are predictors, and the components of $y \in \mathbf{R}^n$ are the responses. To keep things simple, we assume

(A1) the predictors $x \in \mathbf{R}^p$ are independent σ_x-subgaussian random vectors with whose covariance Σ has smallest eigenvalue $\sigma_p(\Sigma) > \lambda_{\min}$ and largest eigenvalue $\sigma_1(\Sigma) < \lambda_{\max}$;

(A2) the regression coefficients $\beta^* \in \mathbf{R}^p$ are s-sparse, i.e. all but s components of β^* are zero;

(A3) the components of the noise ϵ are independent, mean zero σ_y-subgaussian random variables.

Given the predictors and responses, the lasso estimates β^* by

$$\hat{\beta} := \arg\min_{\beta \in \mathbf{R}^p} \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1.$$

There is a well-developed theory of the lasso that says, under suitable assumptions on X, the lasso estimator $\hat{\beta}$ is nearly minimax optimal(e.g. see Hastie et al. (2015), Chapter 11 for an overview). More precisely, under some conditions on $\frac{1}{n} X^T X$, the MSE of the lasso estimator is roughly $\frac{s \log p}{n}$, which is the minimax rate.

2.1 Background on the lasso and debiasing

However, the lasso estimator is also biased[1]. Since averaging only reduces variance, not bias, we gain (almost) nothing by averaging the biased lasso estimators. That is, it is possible to show if we naively averaged local lasso estimators, the MSE of the averaged estimator is of the same order as that of the local estimators. The key to overcoming the bias of the averaged lasso estimator is to "debias" the lasso estimators before averaging.

The *debiased lasso estimator* by Javanmard and Montanari (2013a) is

$$\hat{\beta}^d := \hat{\beta} + \frac{1}{n} \hat{\Theta} X^T (y - X\hat{\beta}), \tag{1}$$

where $\hat{\beta}$ is the lasso estimator and $\hat{\Theta} \in \mathbf{R}^{p \times p}$ is an approximate inverse to $\hat{\Sigma} = \frac{1}{n} X^T X$. Intuitively, the debiased lasso estimator trades bias for variance. The trade-off is obvious when $\hat{\Sigma}$ is non-singular: setting $\hat{\Theta} = \hat{\Sigma}^{-1}$ gives the ordinary least squares (OLS) estimator $(X^T X)^{-1} X^T y$.

Another way to interpret the debiased lasso estimator is a corrected estimator that compensates for the bias incurred by shrinkage. By the optimality conditions of the lasso, the correction term $\frac{1}{n} X^T (y - X\hat{\beta})$ is a subgradient of $\lambda \|\cdot\|_1$ at $\hat{\beta}$. By adding a term proportional to the subgradient of the regularizer, the debiased lasso estimator compensates for the bias incurred by regularization. The debiased lasso estimator has previously been used to perform inference on the regression coefficients in high-dimensional regression models. We refer to the papers by Javanmard and Montanari (2013a); van de Geer et al. (2013); Zhang and Zhang (2014); Belloni et al. (2011) for details.

The choice of $\hat{\Theta}$ in the correction term is crucial to the performance of the debiased estimator. Javanmard and Montanari (2013a) suggest forming $\hat{\Theta}$ row by row: the j-th row of $\hat{\Theta}$ is the optimum of

$$\begin{aligned} \underset{\theta \in \mathbf{R}^p}{\text{minimize}} \quad & \theta^T \hat{\Sigma} \theta \\ \text{subject to} \quad & \|\hat{\Sigma}\theta - e_j\|_\infty \leq \delta. \end{aligned} \tag{2}$$

1. We refer to Section 2.2 in Javanmard and Montanari (2013a) for a more formal discussion of the bias of the lasso estimator.

The parameter δ should large enough to keep the problem feasible, but as small as possible to keep the bias (of the debiased lasso estimator) small. As we shall see, when the rows of X are subgaussian, setting $\delta \sim \left(\frac{\log p}{n}\right)^{\frac{1}{2}}$ is usually large enough to keep (2) feasible.

Definition 1 (Generalized coherence) *Given $X \in \mathbf{R}^{n \times p}$, let $\hat{\Sigma} = \frac{1}{n} X^T X$. The* generalized coherence *between $\hat{\Sigma}$ and $\Theta \in \mathbf{R}^{p \times p}$ is*

$$\mathrm{GC}(\hat{\Sigma}, \Theta) = \max_{j \in [p]} \|\hat{\Sigma}\Theta_j^T - e_j\|_\infty.$$

The preceding definition is a generalization of the usual notion of coherence as it appears in the compressed sensing literature. Assume the columns of X are normalized so that $\|x_j\|_2 = 1$, and $\Theta = I$. The diagonal entries of $\hat{\Sigma}\Theta - I$ vanish. Thus $\mathrm{GC}(\hat{\Sigma}, \Theta)$ is the largest off diagonal entry of $\hat{\Sigma}$, which is the largest inner product between columns of X: $\frac{1}{n} \max_{i \neq j} \left| e_i^T X^T X e_j \right|$. We recognize the preceding quantity as the coherence of X.

Lemma 2 (Javanmard and Montanari (2013a)) *Under (A1), when $16 \kappa \sigma_x^4 n > \log p$, the event*

$$\mathcal{E}_{\mathrm{GC}}(\hat{\Sigma}) := \left\{ \mathrm{GC}(\hat{\Sigma}, \Sigma^{-1}) \leq \frac{8}{\sqrt{c_1}} \sqrt{\kappa} \sigma_x^2 \left(\frac{\log p}{n} \right)^{\frac{1}{2}} \right\}$$

occurs with probability at least $1 - 2p^{-2}$ for some $c_1 > 0$, where $\kappa := \frac{\lambda_{\max}(\Sigma)}{\lambda_{\min}(\Sigma)}$ is the condition number of Σ.

As we shall see, the bias of the debiased lasso estimate is of higher order than its variance under suitable conditions on $\hat{\Sigma}$. In particular, we require $\hat{\Sigma}$ to satisfy the *restricted eigenvalue (RE) condition.*

Definition 3 (RE condition) *For any $\mathcal{S} \subset [p]$, let*

$$\mathcal{C}(\mathcal{S}) := \{\Delta \in \mathbf{R}^p \mid \|\Delta_{\mathcal{S}^c}\|_1 \leq 3 \|\Delta_{\mathcal{S}}\|_1\}.$$

We say $\hat{\Sigma}$ satisfies the RE condition on the cone $\mathcal{C}(\mathcal{S})$ when

$$\Delta^T \hat{\Sigma} \Delta \geq \mu_l \|\Delta_{\mathcal{S}}\|_2^2$$

for some $\mu_l > 0$ and any $\Delta \in \mathcal{C}(\mathcal{S})$.

The RE condition requires $\hat{\Sigma}$ to be positive definite on $\mathcal{C}(\mathcal{S})$. When the rows of $X \in \mathbf{R}^{n \times p}$ are *i.i.d.* Gaussian random vectors, Raskutti et al. (2010) show there are constants $\mu_1, \mu_2 > 0$ such that

$$\frac{1}{n} \|X\Delta\|_2^2 \geq \mu_1 \|\Delta\|_2^2 - \mu_2 \frac{\log p}{n} \|\Delta\|_1^2 \text{ for any } \Delta \in \mathbf{R}^p$$

with probability at least $1 - c_2 \exp(-c_2 n)$. Their result implies the RE condition holds on $\mathcal{C}(\mathcal{S})$ (for any $\mathcal{S} \subset [p]$) as long as $n \gtrsim |S| \log p$, even when there are dependencies among the predictors. Their result was extended to subgaussian designs by Rudelson and Zhou (2013), also allowing for dependencies among the covariates. We state their result verbatim.

Lemma 4 (Rudelson and Zhou (2013)) *Under (A1), if* $n > 4000\tilde{s}\sigma_x^2 \log(\frac{Cp}{\tilde{s}})$ *and* $p > \tilde{s}$, *where* $\tilde{s} := s + 25920\kappa s = C's$, *the event*

$$\mathcal{E}_{\mathrm{RE}}(X) = \left\{\Delta^T\hat{\Sigma}\Delta \geq \frac{1}{2}\lambda_{\min}(\Sigma)\|\Delta_S\|_2^2 \text{ for any } \Delta \in \mathcal{C}(S)\right\}$$

occurs with probability at least $1 - 2e^{-\frac{n}{4000\sigma_x^4}}$, *where* C *and* C' *are universal constants.*

Proof The lemma is a consequence of Rudelson and Zhou (2013), Theorem 6. In their notation, we set $\delta = \frac{1}{\sqrt{2}}$, $k_0 = 3$ and bound $\max_{j\in[p]} \|Ae_j\|_2^2$ and $K(s_0, k_0, \Sigma^{\frac{1}{2}})$ by $\lambda_{\max}(\Sigma)$ and $\lambda_{\min}(\Sigma)^{-\frac{1}{2}}$. ■

When the RE condition holds, the lasso and debiased lasso estimators are consistent for a suitable choice of the regularization parameter λ. The parameter λ should be large enough to dominate the "empirical process" part of the problem: $\frac{1}{n}\left\|X^T y\right\|_\infty$, but as small as possible to reduce the bias incurred by regularization. As we shall see, setting $\lambda \sim \sigma_y(\frac{\log p}{n})^{\frac{1}{2}}$ is a good choice.

Lemma 5 *Under (A3),*

$$\frac{1}{n}\|X^T\epsilon\|_\infty \leq \max_{j\in[p]}(\hat{\Sigma}_{j,j})^{\frac{1}{2}}\sigma_y\Big(\frac{3\log p}{c_2 n}\Big)^{\frac{1}{2}}$$

with probability at least $1 - ep^{-2}$ *for any (non-random)* $X \in \mathbf{R}^{n\times p}$.

When $\hat{\Sigma}$ satisfies the RE condition and λ is large enough, Negahban et al. (2012) show that the lasso is consistent.

Lemma 6 (Negahban et al. (2012)) *If in addition to (A2) and (A3),*

1. $\hat{\Sigma}$ *satisfies the RE condition on* $\mathcal{C}(\mathrm{supp}(\beta^*))$ *with constant* μ_l
2. $\frac{1}{n}\|X^T\epsilon\|_\infty \leq \lambda$,

$$\|\hat{\beta} - \beta^*\|_1 \leq \frac{3}{\mu_l}s\lambda \text{ and } \|\hat{\beta} - \beta^*\|_2 \leq \frac{3}{\mu_l}\sqrt{s}\lambda.$$

When the lasso estimator is consistent, the debiased lasso estimator is also consistent. Further, it is possible to show that the bias of the debiased estimator is of higher order than its variance. Similar results by Javanmard and Montanari (2013a); van de Geer et al. (2013); Zhang and Zhang (2014); Belloni et al. (2011) are the key step in showing the asymptotic normality of the (components of) the debiased lasso estimator. The result we state is essentially Javanmard and Montanari (2013a), Theorem 2.3.

Lemma 7 *If in addition to (A2) and (A3),*

1. $\hat{\Sigma}$ *satisfies the RE condition on* $\mathcal{C}(\mathrm{supp}(\beta^*))$ *with constant* μ_l
2. $\frac{1}{n}\|X^T\epsilon\|_\infty \leq \lambda$,
3. $(\hat{\Sigma}, \hat{\Theta})$ *has generalized incoherence* δ,

the debiased lasso estimator has the form

$$\hat{\beta}^d = \beta^* + \frac{1}{n}\hat{\Theta}X^T\epsilon + \hat{\Delta},$$

where $\|\hat{\Delta}\|_\infty \leq \frac{3\delta}{\mu_l}s\lambda$.

Lemma 7, together with Lemmas 5 and 2, shows that the bias of the debiased lasso estimator is of higher order than its variance. In particular, setting λ and δ to be the order of the upper bounds on $\inf_{\Theta\in\mathbf{R}^{p\times p}} \mathrm{GC}(\hat{\Sigma}, \Theta)$ and $\frac{1}{n}\|X^T\epsilon\|_\infty$ given by Lemmas 5 and 2 gives a bias term $\|\hat{\Delta}\|_\infty$ that is $O_P(\frac{s\log p}{n})$. By comparison, the variance term $\frac{1}{n}\|\hat{\Theta}X^T\epsilon\|_\infty$ is the maximum of p subgaussian random variables with mean zero and variances of $O(1)$, which is $O\big((\frac{\log p}{n})^{\frac{1}{2}}\big)$. Thus the bias term is of higher order than the variance term as long as $n \gtrsim s^2\log p$.

Corollary 8 *If in addition to the conditions of Lemma 6,*

1. $(\hat{\Sigma}, \hat{\Theta})$ *has generalized incoherence* $\delta'(\frac{\log p}{n})^{\frac{1}{2}}$,
2. $\lambda = \max_{j\in[p]}(\hat{\Sigma}_{j,j})^{\frac{1}{2}}\sigma_y\big(\frac{3\log p}{c_2 n}\big)^{\frac{1}{2}}$,

$$\|\hat{\Delta}\|_\infty \leq \frac{3\sqrt{3}}{\sqrt{c_2}}\frac{\delta'\max_{j\in[p]}(\hat{\Sigma}_{j,j})^{\frac{1}{2}}}{\mu_l}\sigma_y\frac{s\log p}{n}.$$

The rest of the paper is organized as follows. In the subsequent section, we describe a divide-and-conquer approach to sparse regression and derive its theoretical properties. We show that

1. the averaged estimator converges at the same rate as the communication-intensive centralized lasso estimator. In particular, a thresholded version of the averaged estimator attains the same estimation rate as the centralized lasso estimator $\|\bar{\beta}^{ht} - \beta^*\|_2 \lesssim (\frac{s\log p}{N})^{\frac{1}{2}}$, and only requires one round of communication.
2. a thresholded version of the averaged estimator is model selection consistent as long as the minimum signal strength is at least $(\frac{\log p}{N})^{\frac{1}{2}}$. We remark that the model selection consistency result does not require X to obey an irrepresentability condition, which the centralized lasso does require.

Although the divide-and-conquer approach is communication efficient, it is costly in terms of floating point operations. The parallel runtime of debiasing is roughly equivalent to the cost of evaluating p lasso estimators, due to computation of $\hat{\Theta}$. In the rest of this section, we describe a more sophisticated approach to debiasing: each machine debiases $\frac{p}{m}$, instead of all p, regression coefficients. Thus the parallel runtime of the more sophisticated approach is roughly m times smaller than that of a naive approach.

In Section 4, we further refine the approach to reduce the sample complexity from $n \gtrsim ms^2\log p$ to $n \gtrsim ms\log p$. In Section 6, we show via simulation experiments that

the averaged debiased estimator outperforms averaging local lasso estimates, and performs as well as the centralized lasso. Section 5 generalizes our approach from least-squares to generalized linear models such as logistic regression.

Finally in Section 7, we show the optimality of our estimator in terms of the amount of communication, and rounds of communication using recent work on communication lower bounds. We also provide a comparison of the average debiased estimator and the centralized lasso estimator. The parallel runtime of the averaging debiased estimator is only larger than the centralized lasso by a constant multiplicative factor.

2.2 Averaging debiased lassos

Recall the problem setup: we are given N samples of the form (x_i, y_i) distributed across m machines:

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_m \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}.$$

The k-th machine has local predictors $X_k \in \mathbf{R}^{n_k \times p}$ and responses $y_k \in \mathbf{R}^{n_k}$. To keep things simple, we assume the data is evenly distributed, i.e. $n_1 = \cdots = n_k = n = \frac{N}{m}$. The *averaged debiased lasso* estimator (for lack of a better name) is

$$\bar{\beta} = \frac{1}{m}\sum_{k=1}^{m} \hat{\beta}_k^d = \frac{1}{m}\sum_{k=1}^{m} \big(\hat{\beta}_k + \hat{\Theta}_k X_k^T (y_k - X_k \hat{\beta}_k)\big), \tag{3}$$

We begin by studying the error of the $\bar{\beta}$ in the ℓ_∞ norm.

Lemma 9 *Suppose the local sparse regression problem on each machine satisfies the conditions of Corollary 8, that is when $m \le p$,*

1. $\{\hat{\Sigma}_k\}_{k\in[m]}$ *satisfy the RE condition on* $\mathcal{C}(\mathrm{supp}(\beta^*))$ *with constant* μ_l,
2. $\{(\hat{\Sigma}_k, \hat{\Theta}_k)\}_{k\in[m]}$ *have generalized incoherence* $c_{\mathrm{GC}}\big(\frac{\log p}{n}\big)^{\frac{1}{2}}$,
3. $\lambda_1 = \cdots = \lambda_m = c_\Sigma \sigma_y \big(\frac{3\log p}{c_2 n}\big)^{\frac{1}{2}}$.

Then

$$\|\bar{\beta} - \beta^*\|_\infty \le c\sigma_y \Big(\Big(\frac{c_\Omega \log p}{N}\Big)^{\frac{1}{2}} + \frac{c_{\mathrm{GC}} c_\Sigma}{\mu_l}\sigma_y \frac{s\log p}{n}\Big)$$

with probability at least $1 - ep^{-1}$, *where* $c > 0$ *is a universal constant,* $c_\Omega := \max_{j\in[p],k\in[m]}((\hat{\Theta}_k \hat{\Sigma}_k \hat{\Theta}_k^T)_{j,j})$ *and* $c_\Sigma := \max_{j\in[p],k\in[m]}((\hat{\Sigma}_k)_{j,j})^{\frac{1}{2}}$.

Lemma 10 hints at the performance of the averaged debiased lasso. In particular, we note the first term is $O\big(\big(\frac{\log p}{N}\big)^{\frac{1}{2}}\big)$, which matches the convergence rate of the centralized estimator. When n is large enough, $\frac{s\log p}{n}$ is negligible compared to $\big(\frac{\log p}{N}\big)^{\frac{1}{2}}$, and the error is $O\big(\big(\frac{\log p}{N}\big)^{\frac{1}{2}}\big)$.

Next, we show the conditions of Lemma 10 occur with high probability when the rows of X are independent subgaussian random vectors.

Lemma 10 *Under (A1), (A2), and (A3), when $m < p$, $p > \tilde{s}$,*

1. $n > \max\big\{4000\tilde{s}\sigma_x^2 \log(\frac{Cp}{\tilde{s}}),\ 8000\sigma_x^4 \log p,\ \frac{3}{c_1}\max\{\sigma_x^2, \sigma_x\}\log p\big\}$,
2. $\lambda_1 = \cdots = \lambda_m = \max_{j\in[p],k\in[m]}((\hat{\Sigma}_k)_{j,j})^{\frac{1}{2}}\sigma_y\big(\frac{3\log p}{c_2 n}\big)^{\frac{1}{2}}$,
3. $\delta_1 = \cdots = \delta_m = \frac{8}{\sqrt{c_1}}\sqrt{\kappa}\sigma_x^2\big(\frac{\log p}{n}\big)^{\frac{1}{2}}$ *and form* $\{\hat{\Theta}_k\}_{k\in[m]}$ *by* (2),

$$\|\bar{\beta} - \beta^*\|_\infty \le c\Big(\sigma_y\Big(\frac{\max_{j\in[p]}\Sigma^{-1}_{j,j}\log p}{N}\Big)^{\frac{1}{2}} + \frac{\sqrt{\kappa}\max_{j\in[p]}(\Sigma_{j,j})^{\frac{1}{2}}}{\lambda_{\min}(\Sigma)}\sigma_x^2\sigma_y\frac{s\log p}{n}\Big)$$

with probability at least $1 - (8+e)p^{-1}$ *for some universal constant* $c > 0$.

The averaged debiased lasso has one serious drawback versus the lasso: $\bar{\beta}$ is usually dense. The density of $\bar{\beta}$ detracts from the intrepretability of the coefficients and makes the estimation error large in the ℓ_2 and ℓ_1 norms. To remedy both problems, we threshold the averaged debiased lasso:

$$\mathrm{HT}_t(\bar{\beta}) \leftarrow \bar{\beta}_j \cdot \mathbf{1}_{\{|\bar{\beta}_j| \ge t\}},$$
$$\mathrm{ST}_t(\bar{\beta}) \leftarrow \mathrm{sign}(\bar{\beta}_j)\cdot\max\{|\bar{\beta}_j| - t,\, 0\}.$$

As we shall see, both hard and soft-thresholding give sparse aggregates that are close to β^* in ℓ_2 norm.

Lemma 11 *As long as $t > \|\bar{\beta} - \beta^*\|_\infty$, $\bar{\beta}^{ht} := \mathrm{HT}_t(\bar{\beta})$ satisfies*

1. $\|\bar{\beta}^{ht} - \beta^*\|_\infty \le 2t$,
2. $\|\bar{\beta}^{ht} - \beta^*\|_2 \le 2\sqrt{2s}t$,
3. $\|\bar{\beta}^{ht} - \beta^*\|_1 \le 2\sqrt{2}st$.

The analogous result also holds for $\bar{\beta}^{st} := \mathrm{ST}_t(\bar{\beta})$.

Proof By the triangle inequality,

$$\begin{aligned}\|\bar{\beta}^{ht} - \beta^*\|_\infty &\le \|\bar{\beta}^{ht} - \bar{\beta}\|_\infty + \|\bar{\beta} - \beta^*\|_\infty \\ &\le t + \big\|\bar{\beta} - \beta^*\big\|_\infty \\ &\le 2t.\end{aligned}$$

Since $t > \big\|\bar{\beta} - \beta^*\big\|_\infty$, $\bar{\beta}_j^{ht} = 0$ whenever $\beta_j^* = 0$. Thus $\bar{\beta}^{ht}$ is s-sparse and $\bar{\beta}^{ht} - \beta^*$ is $2s$-sparse. By the equivalence between the ℓ_∞ and ℓ_2, ℓ_1 norms,

$$\|\bar{\beta}^{ht} - \beta^*\|_2 \le 2\sqrt{2s}t,$$
$$\|\bar{\beta}^{ht} - \beta^*\|_1 \le 2\sqrt{2}st.$$

The argument for $\bar{\beta}^{st}$ is similar. ■

By combining Lemma 11 with Lemma 10, we show that $\bar{\beta}^{ht}$ converges at the same rates as the centralized lasso.

Theorem 12 *Under the conditions of Lemma 10, hard-thresholding $\bar{\beta}$ at $\sigma_y\Big(\frac{4\max_{j\in[p]}\Sigma^{-1}_{j,j}\log p}{c_2 N}\Big)^{\frac{1}{2}} + \frac{48\sqrt{6}}{\sqrt{c_1 c_2}}\frac{\sqrt{\kappa}\max_{j\in[p]}(\Sigma_{j,j})^{\frac{1}{2}}}{\lambda_{\min}(\Sigma)}\sigma_x^2\sigma_y\frac{s\log p}{n}$ gives*

1. $\|\bar{\beta}^{ht}-\beta^*\|_\infty \lesssim_P \sigma_y\Big(\frac{\max_{j\in[p]}\Sigma^{-1}_{j,j}\log p}{N}\Big)^{\frac{1}{2}} + \frac{\sqrt{\kappa}\max_{j\in[p]}(\Sigma_{j,j})^{\frac{1}{2}}}{\lambda_{\min}(\Sigma)}\sigma_x^2\sigma_y\frac{s\log p}{n}$,
2. $\|\bar{\beta}^{ht}-\beta^*\|_2 \lesssim_P \sigma_y\Big(\frac{\max_{j\in[p]}\Sigma^{-1}_{j,j}s\log p}{N}\Big)^{\frac{1}{2}} + \frac{\sqrt{\kappa}\max_{j\in[p]}(\Sigma_{j,j})^{\frac{1}{2}}}{\lambda_{\min}(\Sigma)}\sigma_x^2\sigma_y\frac{s^{\frac{3}{2}}\log p}{n}$,
3. $\|\bar{\beta}^{ht}-\beta^*\|_1 \lesssim_P \sigma_y\Big(\frac{\max_{j\in[p]}\Sigma^{-1}_{j,j}s^2\log p}{N}\Big)^{\frac{1}{2}} + \frac{\sqrt{\kappa}\max_{j\in[p]}(\Sigma_{j,j})^{\frac{1}{2}}}{\lambda_{\min}(\Sigma)}\sigma_x^2\sigma_y\frac{s^2\log p}{n}$.

Remark 13 *By Theorem 12, when $m \lesssim \frac{n}{s^2\log p}$, the variance term is dominant and the convergence rates given by the theorem simplify:*

1. $\|\bar{\beta}^{ht}-\beta^*\|_\infty \lesssim_P \big(\frac{\log p}{N}\big)^{\frac{1}{2}}$,
2. $\|\bar{\beta}^{ht}-\beta^*\|_2 \lesssim_P \big(\frac{s\log p}{N}\big)^{\frac{1}{2}}$,
3. $\|\bar{\beta}^{ht}-\beta^*\|_1 \lesssim_P \big(\frac{s^2\log p}{N}\big)^{\frac{1}{2}}$.

The convergence rates for the centralized lasso estimator $\hat{\beta}$ are identical (modulo constants):

1. $\|\hat{\beta}-\beta^*\|_\infty \lesssim_P \big(\frac{\log p}{N}\big)^{\frac{1}{2}}$,
2. $\|\hat{\beta}-\beta^*\|_2 \lesssim_P \big(\frac{s\log p}{N}\big)^{\frac{1}{2}}$,
3. $\|\hat{\beta}-\beta^*\|_1 \lesssim_P \big(\frac{s^2\log p}{N}\big)^{\frac{1}{2}}$.

The estimator $\bar{\beta}^{ht}$ matches the convergence rates of the centralized lasso in ℓ_1, ℓ_2, and ℓ_∞ norms. Furthermore, $\bar{\beta}^{ht}$ can be evaluated in a communication-efficient manner by a one-shot averaging approach.

Corollary 14 *Under the conditions of Lemma 10, further assume*

1. $m \lesssim \frac{n}{s^2\log p}$,
2. *β-min:* $|\beta^*_j| \gtrsim \big(\frac{\log p}{N}\big)^{\frac{1}{2}}$ *for any* $j \in \mathrm{supp}(\beta^*)$.

Then $\mathrm{supp}(\bar{\beta}^{ht}) = \mathrm{supp}(\beta^*)$.

Proof As long as we threshold at $t > \|\bar{\beta}^{ht}-\beta^*\|_\infty$, $\mathrm{supp}(\bar{\beta}^{ht}) \subset \mathrm{supp}(\beta^*)$. That is, all the zero components of β^* are correctly estimated. Further, as long as the non-zero components of β^* have magnitude at least $2t$, they are not set to zero by thresholding at t. By Theorem 12, there is such a $t \sim \big(\frac{\log p}{N}\big)^{\frac{1}{2}}$. ■

3. A distributed approach to debiasing

The averaged estimator we studied has the form

$$\bar{\beta} = \frac{1}{m}\sum_{k=1}^m \hat{\beta}_k + \hat{\Theta}_k X_k^T(y - X_k\hat{\beta}_k).$$

The estimator requires each machine to form $\hat{\Theta}_k$ by the solution of (2). Since the dual of (2) is an ℓ_1-regularized quadratic program:

$$\underset{\gamma\in\mathbf{R}^p}{\text{minimize}}\ \frac{1}{2}\gamma^T\hat{\Sigma}_k\gamma - \hat{\Sigma}_k\gamma + \delta\|\gamma\|_1, \tag{4}$$

forming $\hat{\Theta}_k$ is (roughly speaking) p times as expensive as solving the local lasso problem, making it the most expensive step (in terms of floating point operations) of evaluating the averaged estimator. To trim the cost of the debiasing step, we consider an estimator that forms only a single $\hat{\Theta}$:

$$\tilde{\beta} = \frac{1}{m}\sum_{k=1}^m \hat{\beta}_k + \frac{1}{N}\hat{\Theta}\sum_{k=1}^m X_k^T(y - X_k\hat{\beta}_k). \tag{5}$$

To evaluate (5),

1. each machine sends $\hat{\beta}_k$ and $\frac{1}{n}X_k^T(y - X_k\hat{\beta}_k)$ to a central server,
2. the central server forms $\frac{1}{m}\sum_{k=1}^m \hat{\beta}_k$ and $\frac{1}{N}\sum_{k=1}^m X_k^T(y - X_k\hat{\beta}_k)$ and sends the averages to all the machines,
3. each machine, given the averages, forms $\frac{p}{m}$ rows of $\hat{\Theta}$ and debiases $\frac{p}{m}$ coefficients:

$$\tilde{\beta}_j = \frac{1}{m}\sum_{k=1}^m \hat{\beta}_j + \hat{\Theta}_{j,\cdot}\Big(\frac{1}{N}\sum_{k=1}^m X_k^T(y - X_k\hat{\beta}_k)\Big),$$

where $\hat{\Theta}_{j,\cdot} \in \mathbf{R}^p$ is a row vector.

As we shall see, each machine can perform debiasing with only the data stored locally. Thus, forming the estimator (5) requires two rounds of communication.

The question that remains is how to form $\hat{\Theta}_{j,\cdot}$. We consider an estimator proposed by van de Geer et al. (2013): nodewise regression on the predictors. For some $j \in [p]$ that machine k is debiasing, the machine solves

$$\hat{\gamma}_j := \underset{\gamma\in\mathbf{R}^{p-1}}{\arg\min}\frac{1}{2n}\|X_{k,j} - X_{k,-j}\gamma\|_2^2 + \lambda_j\|\gamma\|_1,\ j \in [p],$$

where $X_{k,-j} \in \mathbf{R}^{n\times(p-1)}$ is X_k less its j-th column $X_{k,j}$. Implicitly, we are forming

$$\hat{C} := \begin{bmatrix} 1 & -\hat{\gamma}_{1,2} & \dots & -\hat{\gamma}_{1,p} \\ -\hat{\gamma}_{2,1} & 1 & \dots & -\hat{\gamma}_{2,p} \\ \vdots & \vdots & \ddots & \vdots \\ -\hat{\gamma}_{p,1} & -\hat{\gamma}_{p,2} & \dots & -\hat{\gamma}_{p,p} \end{bmatrix},$$

where the components of $\hat{\gamma}_j$ are indexed by $k \in \{1, \ldots, j-1, j+1, \ldots, p\}$. We scale the rows of $\hat{C}$ by $\mathbf{diag}\Big(\big[\hat{\tau}_1, \ldots, \hat{\tau}_p\big]\Big)$, where

$$\hat{\tau}_j = \big(\frac{1}{n}\|X_j - X_{-j}\hat{\gamma}_j\|_2^2 + \lambda_j\|\hat{\gamma}_j\|_1\big)^{\frac{1}{2}},$$

to form $\hat{\Theta} = \hat{T}^{-2}\hat{C}$. Each row of $\hat{\Theta}$ is given by

$$\hat{\Theta}_{j,\cdot} = -\frac{1}{\hat{\tau}_j^2}\begin{bmatrix}\hat{\gamma}_{j,1} & \ldots & \hat{\gamma}_{j,j-1} & 1 & \hat{\gamma}_{j,j+1} & \ldots & \hat{\gamma}_{j,p}\end{bmatrix}. \tag{6}$$

Since $\hat{\gamma}_j$ and $\hat{\tau}_j$ only depend on X_k, they can be formed without any communication.

Before we justify the choice of $\hat{\Theta}$ theoretically, we mention that it is an approximate "inverse" of $\hat{\Sigma}$ (in a component-wise sense). By the optimality conditions of nodewise regression,

$$\begin{aligned}\hat{\tau}_j^2 &= \frac{1}{n}\|X_j - X_{-j}\hat{\gamma}_j\|_2^2 + \lambda_j\|\hat{\gamma}_j\|_1 \\ &= \frac{1}{n}\|X_j - X_{-j}\hat{\gamma}_j\|_2^2 + \frac{1}{n}(X_j - X_{-j}\hat{\gamma}_j)^T X_{-j}^T\hat{\gamma}_j \\ &= \frac{1}{n}X_j(X_j - X_{-j}\hat{\gamma}).\end{aligned}$$

Recalling the defintition of $\hat{\Theta}$, we have

$$\frac{1}{n}\hat{\Theta}_{j,\cdot}X^T X_j = \frac{1}{\hat{\tau}_j^2}\frac{1}{n}(X_j - \hat{\gamma}_j^T X_{-j})^T X_j = \lambda_j \text{ and}$$
$$\frac{1}{n}\|\hat{\Theta}_{j,\cdot}X^T X_{-j}\|_\infty = \frac{1}{\hat{\tau}_j^2}\Big\|\frac{1}{n}(X_j - \hat{\gamma}_j^T X_{-j})^T X_{-j}\Big\|_\infty \leq \frac{\lambda_j}{\hat{\tau}_j^2}$$

for any $j \in [p]$. Thus

$$\max_{j\in[p]}\|\hat{\Theta}_{j,\cdot}\hat{\Sigma} - e_j\|_\infty \leq \frac{\lambda_j}{\hat{\tau}_j^2}. \tag{7}$$

van de Geer et al. (2013) show that when the rows of X are *i.i.d.* subgaussian random vectors and the precision matrix Σ^{-1} is sparse, $\hat{\Theta}_{j,\cdot}$ converges to Σ_j^{-1} at the usual convergence rate of the lasso. For completeness, we restate their result.

We consider a sequence of regression problems indexed by the sample size N, dimension p, sparsity s_0 that satisfies (A1), (A2), and (A3). As N grows to infinity, both $p = p(N)$ and $s = s(N)$ may also grow as a function of N. To keep notation manageable, we drop the index N. We further assume

(A4) the covariance of the predictors (rows of X) has smallest eigenvalue $\lambda_{\min}(\Sigma) \sim \Omega(1)$ and largest diagonal entry $\max_{j\in[p]}\Sigma_{j,j} \sim O(1)$,

(A5) the rows of Σ^{-1} are sparse: $\max_{j\in[p]}\frac{s_j^2\log p}{n} \sim o(1)$, where s_j is the sparsity of Σ_j^{-1}.

Lemma 15 (van de Geer et al. (2013), Theorem 2.4) *Under (A1)–(A5),* (6) *with suitable parameters* $\lambda_j \sim \big(\frac{\log p}{n}\big)^{\frac{1}{2}}$ *satisfies*

$$\|\hat{\Theta}_{j,\cdot} - \Sigma_j^{-1}\|_1 \lesssim_P \left(\frac{s_j^2\log p}{n}\right)^{\frac{1}{2}} \text{ for any } j \in [p].$$

We show that the averaged estimator (5) matches the convergence rate of the centralized lasso.

Theorem 16 *Under (A1)–(A5),* (5)*, where* $\hat{\Theta}$ *is given by* (6)*, with suitable parameters* $\lambda_j, \lambda_k \sim \big(\frac{\log p}{n}\big)^{\frac{1}{2}}$*,* $j \in [p]$*,* $k \in [m]$ *satisfies*

$$\|\bar{\beta} - \beta^*\|_\infty \lesssim_P \Big(\frac{\log p}{N}\Big)^{\frac{1}{2}} + \frac{s_{\max}\log p}{n},$$

where $s_{\max} := \max\{s_0, s_1, \ldots, s_p\}$.

Proof See the appendix. ■

By combining the Lemma 11 with Theorem 16, we can show that $\tilde{\beta}^{ht} := \mathrm{HT}(\tilde{\beta}, t)$ for an appropriate threshold t converges to β^* at the same rates as the centralized lasso.

Theorem 17 *Under the conditions of Theorem 16, hard-thresholding* $\tilde{\beta}$ *at* $t \sim \big(\frac{\log p}{N}\big)^{\frac{1}{2}} + \frac{s_{\max}\log p}{n}$ *gives*

1. $\|\tilde{\beta}^{ht} - \beta^*\|_\infty \lesssim_P \big(\frac{\log p}{N}\big)^{\frac{1}{2}} + \frac{s_{\max}\log p}{n}$,
2. $\|\tilde{\beta}^{ht} - \beta^*\|_2 \lesssim_P \big(\frac{s_0\log p}{N}\big)^{\frac{1}{2}} + \frac{\sqrt{s_0}s_{\max}\log p}{n}$,
3. $\|\tilde{\beta}^{ht} - \beta^*\|_1 \lesssim_P \big(\frac{s_0^2\log p}{N}\big)^{\frac{1}{2}} + \frac{s_0 s_{\max}\log p}{n}$.

Theorem 17 shows that for $m \lesssim \frac{n}{s_{\max}^2\log p}$, the variance term is dominant, so the convergence rates simplify:

1. $\|\tilde{\beta}^{ht} - \beta^*\|_\infty \lesssim_P \big(\frac{\log p}{N}\big)^{\frac{1}{2}}$,
2. $\|\tilde{\beta}^{ht} - \beta^*\|_2 \lesssim_P \big(\frac{s_{\max}\log p}{N}\big)^{\frac{1}{2}}$,
3. $\|\tilde{\beta}^{ht} - \beta^*\|_1 \lesssim_P \big(\frac{s_{\max}^2\log p}{N}\big)^{\frac{1}{2}}$.

Thus, estimator $\tilde{\beta}^{ht}$ shares the advantages of $\bar{\beta}^{ht}$ over the centralized lasso (cf. Remark 13). It also achieves computational gains over $\bar{\beta}^{ht}$ by amortizing the cost of debiasing across m machines.

4. A sharper estimation result

It is possible to obtain a sharper estimation result by forgoing the ℓ_∞ norm convergence rate. By sharper, we mean the sample complexity of the averaged estimator from $m \lesssim \frac{n}{s_0^2 \log p}$ to $m \lesssim \frac{n}{s_0 \log p}$.

The sharper estimation result depends on a result by Javanmard and Montanari (2013b), which we combine with Lemma 15 and restate for completeness. Before stating the results, we define the (∞, l) norm of a point $x \in \mathbf{R}^p$ as

$$\|x\|_{(\infty,l)} := \max_{\mathcal{A} \subset [p], |\mathcal{A}| \geq l} \frac{\|x_{\mathcal{A}}\|_2}{\sqrt{l}}.$$

When $l = 1$, the (∞, l) norm of x is its ℓ_∞ norm. When $l = p$, the (∞, l) norm is the ℓ_2 norm (rescaled by $\frac{1}{\sqrt{p}}$). Thus the (∞, l) norm interpolates between the ℓ_2 and ℓ_∞ norms. Javanmard and Montanari (2013b), Theorem 2.3 shows that the bias of the debiased lasso is of order $\frac{\sqrt{s_0} \log p}{n}$.

Lemma 18 *Under the conditions of Theorem 16,*

$$\|\hat{\Delta}_k\|_{(\infty, c' s_0)} \lesssim_P \frac{c\sqrt{s_0} \log p}{n} \text{ for any } k \in [m] \text{ for any } c' > 0,$$

where c is a constant that depends only on c' and Σ.

By Lemma 18, the estimator (5) is consistent in the (∞, s_0) norm. The argument is similar to the proof of Theorem 16.

Theorem 19 *Under the conditions of Theorem 16,*

$$\|\bar{\beta} - \beta^*\|_{(\infty, c' s_0)} \sim O_P\Big(\Big(\frac{\log p}{N}\Big)^{\frac{1}{2}} + \frac{\sqrt{s_0} \log p}{n}\Big).$$

Theorem 20 *Under the conditions of Theorem 16, hard-thresholding $\tilde{\beta}$ at $t = |\tilde{\beta}|_{(\hat{s}_0)}$ for some $\hat{s}_0 \sim s_0$, i.e. setting all but the largest $\hat{s}_0$ debiased coefficients to zero, gives*

1. $\|\tilde{\beta}^{ht} - \beta^*\|_2 \lesssim_P \big(\frac{s_0 \log p}{N}\big)^{\frac{1}{2}} + \frac{s_0 \log p}{n}$,
2. $\|\bar{\beta}^{ht} - \beta^*\|_1 \lesssim_P \big(\frac{s_0^2 \log p}{N}\big)^{\frac{1}{2}} + \frac{s_0^{3/2} \log p}{n}$.

By Theorem 20, when $m \lesssim \frac{N}{s_0 \log p}$, the variance term is dominant and the convergence rates given by the theorem simplify to the convergence rates of the (centralized) lasso estimator:

1. $\|\bar{\beta}^{ht} - \beta^*\|_2 \lesssim_P \big(\frac{s_0 \log p}{N}\big)^{\frac{1}{2}}$,
2. $\|\bar{\beta}^{ht} - \beta^*\|_1 \lesssim_P \big(\frac{s_0^2 \log p}{N}\big)^{\frac{1}{2}}$.

Thus, by forgoing estimation error in the ℓ_∞ norm, it is possible to reduce the sample complexity of the averaged estimator to $m \lesssim \frac{s_0 \log p}{N}$. When $m = 1$, we recover the sample complexity of the centralized lasso estimator.

5. Averaging debiased ℓ_1 regularized M-estimators

The distributed approach to debiasing extends readily to ℓ_1 regularized M-estimators. As before, we are given N pairs (x_i, y_i) stored on m machines. Let $\rho(y_i, a)$ be a loss function function, which is convex in a, and $\dot{\rho}$, $\ddot{\rho}$ be its derivatives with respect to a. That is

$$\dot{\rho}(y, a) = \frac{d}{da}\rho(y, a), \quad \ddot{\rho}(y, a) = \frac{d^2}{da^2}\rho(y, a).$$

We define $\ell_k(\beta) = \frac{1}{n}\sum_{i=1}^n \rho(y_i, x_i^T \beta)$, where the sum is only over the pairs on machine k. The averaged estimator is

$$\bar{\beta} := \frac{1}{m}\sum_{k=1}^m \hat{\beta}_k + \hat{\Theta}\Big(\frac{1}{m}\sum_{k=1}^m \nabla \ell_k(\hat{\beta}_k)\Big), \tag{8}$$

where $\hat{\beta}_k$ is the local ℓ_1 regularized M-estimator: $\hat{\beta}_k := \arg\min_{\beta \in \mathbf{R}^p} \ell_k(\beta) + \lambda_k \|\beta\|_1$. As before, we form $\hat{\Theta}$ by nodewise regression on the weighted design matrix $X_{\hat{\beta}_k} := W_{\hat{\beta}_k} X_k$, where $W_{\hat{\beta}_k}$ is diagonal and its diagonal entries are

$$\big(W_{\hat{\beta}_k}\big)_{i,i} := \ddot{\rho}(y_i, x_i^T \hat{\beta}_k)^{\frac{1}{2}}.$$

That is, for some $j \in [p]$ that machine k is debiasing, the machine solves

$$\hat{\gamma}_j := \arg\min_{\gamma \in \mathbf{R}^{p-1}} \frac{1}{2n}\|X_{\hat{\beta}_k, j} - X_{\hat{\beta}_k, -j}\gamma\|_2^2 + \lambda_j \|\gamma\|_1, \; j \in [p],$$

and forms

$$\hat{\Theta}_{j,\cdot} = -\frac{1}{\hat{\tau}_j^2}\begin{bmatrix}\hat{\gamma}_{j,1} & \dots & \hat{\gamma}_{j,j-1} & 1 & \hat{\gamma}_{j,j+1} & \dots & \hat{\gamma}_{j,p}\end{bmatrix},$$

where

$$\hat{\tau}_j = \big(\frac{1}{n}\|X_{\hat{\beta}_k, j} - X_{\hat{\beta}_k, -j}\hat{\gamma}_j\|_2^2 + \lambda_j \|\hat{\gamma}_j\|_1\big)^{\frac{1}{2}}.$$

We assume

(B1) the pairs $\{(x_i, y_i)\}_{i \in [N]}$ are *i.i.d.*; the predictors are bounded:

$$\max_{i \in [N]} \|x_i\|_\infty \lesssim 1;$$

the projection of $X_{\beta^*, j}$ on $\mathcal{R}(X_{\beta^*, -j})$ in the $\mathbf{E}\left[\nabla^2 \ell_k(\beta^*)\right]$ inner product is bounded: $\|X_{\beta^*, -j}\gamma_{\beta^*, j}\|_\infty \lesssim 1$ for any $j \in [p]$, where

$$\gamma_{\beta^*, j} := \arg\min_{\gamma \in \mathbf{R}^{p-1}} \mathbf{E}\left[\|X_{\beta^*, j} - X_{\beta^*, -j}\gamma\|_2^2\right].$$

(B2) the rows of $\mathbf{E}\left[\nabla^2 \ell_k(\beta^*)\right]^{-1}$ are sparse: $\max_{j \in [p]} \frac{s_j^2 \log p}{n} \sim o(1)$, where s_j is the sparsity of $\left(\mathbf{E}\left[\nabla^2 \ell_k(\beta^*)\right]^{-1}\right)_{j,\cdot}$.

(B3) the smallest eigenvalue of $\mathbf{E}\left[\nabla^2 \ell_k(\beta^*)\right]$ is bounded away from zero and its entries are bounded.

(B4) for any β such that $\|\beta - \beta^*\|_1 \le \delta$ for some $\delta > 0$, the diagonal entries of W_β stays away from zero, and

$$|\ddot{\rho}(y, x^T\beta) - \ddot{\rho}(y, x^T\beta^*)| \le |x^T(\beta - \beta^*)|.$$

(B5) we have $\frac{1}{n}\|X_k(\hat{\beta}_k - \beta^*)\|_2^2 \lesssim_P s_0\lambda_k^2$ and $\|\hat{\beta}_k - \beta^*\|_1 \lesssim_P s_0\lambda_k$.

(B6) the derivatives $\dot{\rho}(y, a)$, $\ddot{\rho}(y, a)$ is locally Lipschitz:

$$\max_{i\in[N]} \sup_{|a,a'-x_i^T\beta^*|\le\delta} \sup_y \frac{|\ddot{\rho}(y,a)-\ddot{\rho}(y,a')|}{|a-a'|} \le K \text{ for some } \delta > 0.$$

Further,

$$\max_{i\in[N]} \sup_y |\dot{\rho}(y, x_i^T\beta)| \sim O(1),$$
$$\max_{i\in[N]} \sup_{|a-x_i^T\beta^*|\le\delta} \sup_y |\ddot{\rho}(y, a)| \sim O(1).$$

(B7) the diagonal entries of

$$\mathbf{E}\big[\nabla^2\ell_k(\beta^*)\big]^{-1}\mathbf{E}\big[\nabla\ell_k(\beta^*)\nabla\ell_k(\beta^*)^T\big]\mathbf{E}\big[\nabla^2\ell_k(\beta^*)\big]^{-1}$$

are bounded.

The preceding assumptions deserve elaboration. Assumptions (B1), (B4), (B6), and (B7) are standard in the literature on high-dimensional regression. They ensure the various intermediate quantities, such as $\rho(y, x^T\beta)$ and its derivaties, remain bounded. Assumption (B2) is perhaps the most restrictive. The assumption serves to ensure that the debiasing step is effective in reducing the bias of the regularized estimator. It may be relaxed (at the cost of additional technicalities) to the rows of $\mathbf{E}\big[\nabla^2\ell_k(\beta^*)\big]^{-1}$ admit a s_j-sparse approximation. We refer to Bühlmann and Van De Geer (2011) for the details. Assumption (B3) is a quantitative version of the usual rank condition in regression. It ensures the regression coefficients are identifiable in the limit. Assumption (B5) is not necessary; it is implied by the other assumptions. We refer to Bühlmann and Van De Geer (2011), Chapter 6 for the details. Here we state it as an assumption to simplify the exposition.

We are ready to state our main results concerning the averaged estimator(8). It shows the averaged estimator achieves the convergence rate of the centralized ℓ_1-regularized M-estimator.

Theorem 21 *Under (B1)–(B7),* (8) *with suitable parameters* $\lambda_j, \lambda_k \sim \big(\frac{\log p}{n}\big)^{\frac{1}{2}}$, $j \in [p]$, $k \in [m]$ *satisfies*

$$\|\bar{\beta} - \beta^*\|_\infty \lesssim_P \Big(\frac{\log p}{N}\Big)^{\frac{1}{2}} + \frac{s_{\max}\log p}{n}, \tag{9}$$

where $s_{\max} := \max\{s_0, s_1, \ldots, s_p\}$.

Proof The averaged estimator is given by

$$\bar{\beta} - \beta^* = \frac{1}{m}\sum_{k=1}^m \hat{\beta}_k - \hat{\Theta}\nabla\ell_k(\hat{\beta}_k)(\hat{\beta}_k - \beta^*) - \beta^*.$$

By the smoothness of ρ,

$$\dot{\rho}(y_i, x_i^T\hat{\beta}_k) = \dot{\rho}(y_i, x_i^T\beta^*) + \ddot{\rho}(y_i, \tilde{a}_i)x_i^T(\hat{\beta}_k - \beta^*),$$

where $\tilde{a}_i$ is a point between $x_i^T\hat{\beta}_k$ and $x_i^T\beta^*$. Thus

$$\begin{aligned}\bar{\beta} - \beta^* &= \frac{1}{m}\sum_{k=1}^m \hat{\beta}_k - \hat{\Theta}(\nabla\ell_k(\beta^*) + Q_k(\hat{\beta}_k - \beta^*)) - \beta^* \\ &= -\hat{\Theta}\big(\tfrac{1}{m}\textstyle\sum_{k=1}^m \nabla\ell_k(\beta^*)\big) + \frac{1}{m}\sum_{k=1}^m \big(I - \hat{\Theta}Q_k\big)(\hat{\beta}_k - \beta^*).\end{aligned}$$

where $Q_k = \frac{1}{n}\sum_{i=1}^n \ddot{\rho}(y_i, \tilde{a}_i)x_i x_i^T$, where the sum is over the data points on machine k. Taking norms, we obtain

$$\|\bar{\beta} - \beta^*\|_\infty \le \big\|\hat{\Theta}\big(\tfrac{1}{m}\textstyle\sum_{k=1}^m \nabla\ell_k(\beta^*)\big)\big\|_\infty + \frac{1}{m}\sum_{k=1}^m \big\|\big(I - \hat{\Theta}Q_k\big)(\hat{\beta}_k - \beta^*)\big\|_\infty.$$

It is possible to show that $\big\|\hat{\Theta}\big(\frac{1}{m}\sum_{k=1}^m \nabla\ell_k(\beta^*)\big)\big\|_\infty \lesssim_P \big(\frac{\log p}{N}\big)^{\frac{1}{2}}$, which corresponds to the first term in (9). We refer to Bühlmann and Van De Geer (2011), Chapter 6 for the details.

We turn our attention to the second term. By the triangle inequality,

$$\begin{aligned}&\|(I - \hat{\Theta}Q_k)(\hat{\beta}_k - \beta^*)\|_\infty \\ &\quad\le \big\|\big(I - \hat{\Theta}\nabla^2\ell_k(\hat{\beta}_k)\big)(\hat{\beta}_k - \beta^*)\big\|_\infty + \big\|\hat{\Theta}(\nabla^2\ell_k(\hat{\beta}_k) - Q_k)(\hat{\beta}_k - \beta^*)\big\|_\infty \\ &\quad\le \max_{j\in[p]}\big\|e_j^T - \hat{\Theta}_{j,\cdot}\nabla^2\ell_k(\hat{\beta}_k)\big\|_\infty\|\hat{\beta}_k - \beta^*\|_1 \\ &\qquad + \frac{1}{n}\sum_{i=1}^n \|\hat{\Theta}x_i\|_\infty|\ddot{\rho}(y_i, x_i^T\hat{\beta}_k) - \ddot{\rho}(y_i, \tilde{a}_i)x_i^T(\hat{\beta}_k - \beta^*)|.\end{aligned}$$

We proceed term by term. By (7),

$$\max_{j\in[p]}\big\|e_j^T - \hat{\Theta}_{j,\cdot}\nabla^2\ell_k(\hat{\beta}_k)\big\|_\infty \le \frac{\lambda_j}{\hat{\tau}_j^2} \lesssim \frac{1}{\hat{\tau}_j^2}\Big(\frac{\log p}{n}\Big)^{\frac{1}{2}}.$$

By van de Geer et al. (2013), Theorem 3.2,

$$|\hat{\tau}_j^2 - \tau_j^2| \lesssim_P \Big(\frac{\max\{s_0, s_j\}\log p}{n}\Big)^{\frac{1}{2}}$$

Thus $\max_{j\in[p]}\big\|e_j^T - \hat{\Theta}_{j,\cdot}\nabla^2\ell_k(\hat{\beta}_k)\big\|_\infty \lesssim_P \big(\frac{\log p}{n}\big)^{\frac{1}{2}}$ and, by (B5),

$$\max_{j\in[p]}\big\|e_j^T - \hat{\Theta}_{j,\cdot}\nabla^2\ell_k(\hat{\beta}_k)\big\|_\infty\|\hat{\beta}_k - \beta^*\|_1 \lesssim_P \frac{s_{\max}\log p}{n}.$$

We turn our attention to the second term. We have $\|\hat{\Theta}x_i\|_\infty \lesssim_P 1$ because

$$\begin{aligned}\|\hat{\Theta}x_i\|_\infty &\le \max_{j\in[p]} \|\hat{\Theta}_{j,\cdot} X_k^T\|_\infty \lesssim \max_{j\in[p]} \|\hat{\Theta}_{j,\cdot} X_{k,\beta^*}^T\|_\infty \\ &\le \max_{j\in[p]} \frac{1}{\hat{\tau}_j^2} \|(X_{k,\beta^*})_j - (X_{k,\beta^*})_{-j}\hat{\gamma}_j\|_\infty.\end{aligned}$$

Again, by van de Geer et al. (2013), Theorem 3.2,

$$\begin{aligned}&\lesssim_P \max_{j\in[p]} \frac{1}{\tau_j^2} \|(X_{k,\beta^*})_j - (X_{k,\beta^*})_{-j}\hat{\gamma}_j\|_\infty \\ &\lesssim_P \max_{j\in[p]} \frac{1}{\tau_j^2} \|(X_{k,\beta^*})_j - (X_{k,\beta^*})_{-j}\gamma_j\|_\infty \\ &\quad + \frac{1}{\tau_j^2} \|(X_{k,\beta^*})_j\|_\infty \|(\hat{\gamma}_j - \gamma_j)\|_1.\end{aligned}$$

which, by (B1) and van de Geer et al. (2013), Theorem 3.2,

$$\lesssim_P 1 + \frac{s_j \log p}{n}.$$

Thus

$$\begin{aligned}&\frac{1}{n}\sum_{i=1}^n \|\hat{\Theta}x_i\|_\infty \left|\ddot{\rho}(y_i, x_i^T\hat{\beta}_k) - \ddot{\rho}(y_i, \tilde{a}_i) x_i^T(\hat{\beta}_k - \beta^*)\right| \\ &\lesssim_P \frac{1}{n}\sum_{i=1}^n \left|\ddot{\rho}(y_i, x_i^T\hat{\beta}_k) - \ddot{\rho}(y_i, \tilde{a}_i) x_i^T(\hat{\beta}_k - \beta^*)\right|,\end{aligned}$$

which, by (B5) and (B6), is at most

$$\lesssim \frac{1}{n}\|X_k(\hat{\beta}_k - \beta^*)\|_2^2 \lesssim_P \frac{s_0 \log p}{n}.$$

We put the pieces together to deduce $\frac{1}{m}\sum_{k=1}^m \left\|(I - \hat{\Theta}Q_k)(\hat{\beta}_k - \beta^*)\right\|_\infty \lesssim_P \frac{s_{\max}\log p}{n}$. ■

By combining the Lemma 11 with Theorem 16, we can show that $\tilde{\beta}^{ht} := \mathrm{HT}(\tilde{\beta}, t)$ for an appropriate threshold t converges to β^* at the same rates as the centralized ℓ_1-regularized M-estimator.

Theorem 22 *Under the conditions of Theorem 21, hard-thresholding $\tilde{\beta}$ at $t \sim \left(\frac{\log p}{N}\right)^{\frac{1}{2}} + \frac{\max_{j\in[p]} s_j \log p}{n}$ gives*

1. $\|\tilde{\beta}^{ht} - \beta^*\|_\infty \lesssim_P \left(\frac{\log p}{N}\right)^{\frac{1}{2}} + \frac{s_{\max}\log p}{n}$,
2. $\|\tilde{\beta}^{ht} - \beta^*\|_2 \lesssim_P \left(\frac{s_0 \log p}{N}\right)^{\frac{1}{2}} + \frac{\sqrt{s_0} s_{\max}\log p}{n}$,
3. $\|\tilde{\beta}^{ht} - \beta^*\|_1 \lesssim_P \left(\frac{s_0^2 \log p}{N}\right)^{\frac{1}{2}} + \frac{s_0 \max_{j\in[p]} s_j \log p}{n}$.

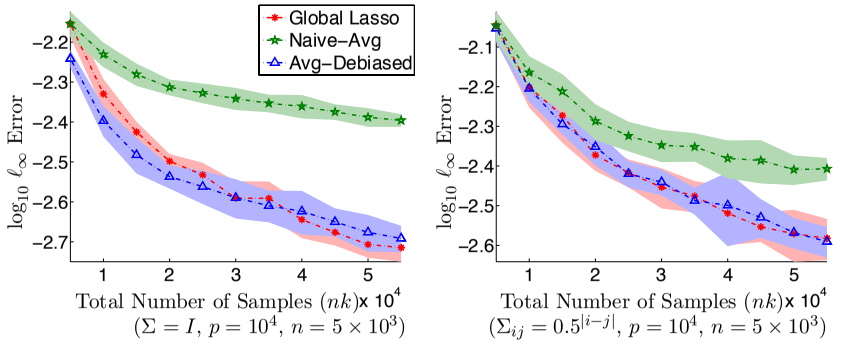


Figure 1: The estimation error (in ℓ_∞ norm) of the averaged debiased lasso estimator versus that of the centralized lasso when the predictors are Gaussian. In both settings, the estimation error of the averaged debiased estimator is comparable to that of the centralized lasso, while that of the naive averaged lasso is much worse.

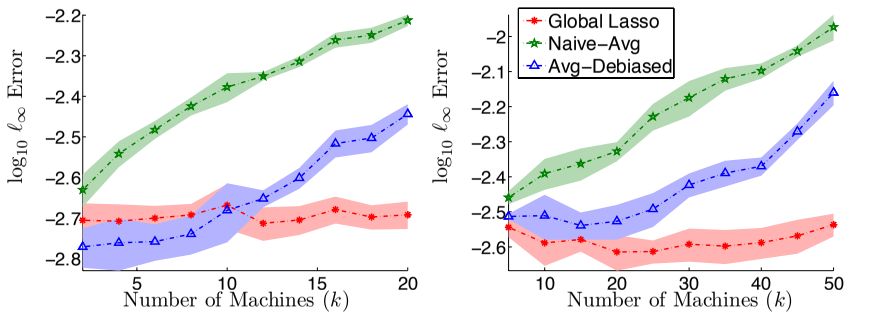
Assuming $s_0 \sim s_{\max}$, Theorem 22 shows when $m \lesssim \frac{n}{s_0^2 \log p}$, the variance term is dominant, so the convergence rates simplify to

1. $\|\tilde{\beta}^{ht} - \beta^*\|_\infty \lesssim_P \left(\frac{\log p}{N}\right)^{\frac{1}{2}}$,
2. $\|\tilde{\beta}^{ht} - \beta^*\|_2 \lesssim_P \left(\frac{s_0 \log p}{N}\right)^{\frac{1}{2}}$,
3. $\|\tilde{\beta}^{ht} - \beta^*\|_1 \lesssim_P \left(\frac{s_0^2 \log p}{N}\right)^{\frac{1}{2}}$.

6. Simulations

We validate our theoretical results with simulations. First, we study the estimation error of the averaged debiased lasso in ℓ_∞ norm. To focus on the effect of averaging, we grow the number of machines m linearly with the (total) sample size N. In other words, we fix the sample size per machine n and grow the total sample size N by adding machines. The tuning parameters were set to their oracle values stated in the Theorem 12. Figure 1 compares the estimation error (in ℓ_∞ norm) of the averaged debiased lasso estimator with that of the centralized lasso. We see the estimation error of the averaged debiased lasso estimator is comparable to that of the centralized lasso, while that of the naive averaged lasso is much worse.

We conduct a second set of simulations to study the effect of the number of machines on the estimation effort of the averaged estimator. To focus on the effect of the number of



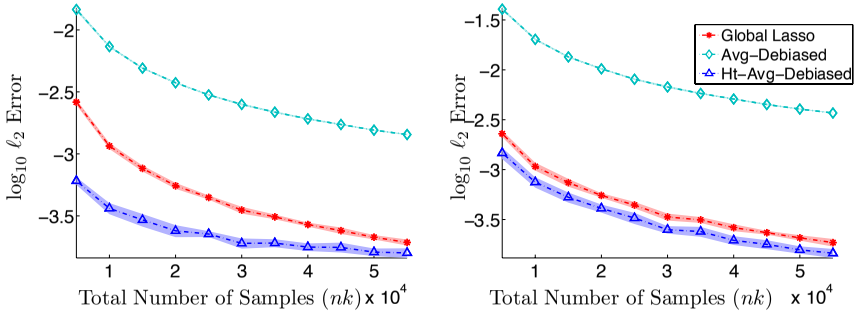
($\Sigma = I$, $p = 10^4$, $nk = 2 \times 10^5$) ($\Sigma_{ij} = 0.5^{|i-j|}$, $p = 10^4$, $nk = 2 \times 10^5$)

Figure 2: The estimation error (in ℓ_∞ norm) of the averaged estimator as the number of machines k vary. When the number of machines is small, the error is comparable to that of the centralized lasso. However, when the number of machines exceeds a certain threshold, the bias term (which grows linearly in k) is dominant, and the performance of the averaged estimator degrades.

machines k, we fix the (total) sample size N and vary the number of machines the samples are distributed across. The tuning parameters were again set to the oracle values stated in the Theorem 12. Figure 2 shows how the estimation error (in ℓ_∞ norm) of the averaged estimator grows as the number of machines grows. When the number of machines is small, the estimation error of the averaged estimator is comparable to that of the centralized lasso. However, when the number of machines exceeds a certain threshold, the estimation error grows with the number of machines. This transition occurs when $\frac{s \log p}{n} \gtrsim \left(\frac{\log p}{N}\right)^{\frac{1}{2}}$, or equivalently, when $k \gtrsim \left(\frac{N}{s^2 \log p}\right)^{\frac{1}{2}}$. The preceding observation is consistent with the prediction of Lemma 10: when the number of machines exceeds a certain threshold, the bias term of order $\frac{s \log p}{n}$ becomes dominant. Since $\frac{s \log p}{n} \propto k$, we expect the error to grow linearly with k, which agrees with the trends in Figure 2.

We conduct a third set of simulations to study the effect of thresholding on the estimation error in ℓ_2 norm. The tuning parameters were set to the oracle values stated in the Theorem 12. Figure 3 compares the estimation error incurred by the averaged estimator with and without thresholding versus that of the centralized lasso. Since the averaged estimator is usually dense, its estimation error (in ℓ_2 norm) is large compared to that of the centralized lasso. However, after thresholding, the averaged estimator performs comparably versus the centralized lasso. This demonstrates the importance of the thresholding step to achieve low ℓ_2 error.

In practice, it is possible to set the tuning parameter δ by the bisection method in the accompanying code to Javanmard and Montanari (2013a); via bisection, they search for the smallest δ such that the optimization program in (2) is feasible. The lasso tuning parameter λ is set by first estimating the noise variance using the residuals and then using the formula



($\Sigma = I$, $p = 10^4$, $n = 5 \times 10^3$) ($\Sigma_{ij} = 0.5^{|i-j|}$, $p = 10^4$, $n = 5 \times 10^3$)

Figure 3: The estimation error (in ℓ_2 norm) of the averaged estimator with and without thresholding versus that of the centralized lasso when the predictors are Gaussian. In both settings, thresholding reduces the estimation error by order(s) of magnitude. Although the estimation error of the averaged estimator is large compared to that of the centralized lasso, the thresholded averaged estimator performs comparably, or even better than, the centralized lasso.

$\lambda = \sigma\sqrt{2 \log p}$. The parameter λ can be chosen independently of σ, if we replace the lasso with the sqrt-lasso Belloni et al. (2011); all of the same theoretical guarantees still apply, since the sqrt-lasso has the same consistency guarantees. For generalized linear models, the oracle choice of λ depends on known quantities Negahban et al. (2012).

7. Summary and discussion

We devised a communication-efficient approach to distributed sparse regression in the high-dimensional setting. The key idea is first "debiasing" local lasso estimators, and then averaging the debiased estimators. We show that as long as the data is not split across too many machines, the averaged estimator achieves the convergence rate of the centralized lasso estimator. In the appendix, we show that by foregoing consistency in the ℓ_∞ norm, it is possible to further reduce the sample complexity of the averaged estimator to that of the centralized lasso estimator. Further, the distributed approach to debiasing extends readily to other ℓ_1 regularized M-estimators. In concurrent work, the approach of averaging debiased M-estimators was proposed by Battey et al. (2015) for high-dimensional inference.

7.1 Communication and Computational complexity

In recent years, there has a been a flurry of work on establishing communication lower bounds for mean estimation in the Gaussian distribution. In other words, they establish the minimum communication C needed to obtain ℓ_2^2 risk R , where $\|\hat{\beta} - \beta^*\|_2^2 \leq R$ (Duchi et al., 2014; Garg et al., 2014). These results are not directly applicable to sparse linear

regression, since they do not impose sparsity on the mean. In Braverman et al. (2015), the authors established that to obtain risk $R \leq \frac{s \log p}{N}$ at least $\Omega\big(\frac{m \min(n,p)}{\log p}\big)$ bits of communication is required. Our approach communicates $\tilde{O}(mp)$ bits to achieve risk of $\frac{s \log p}{N}$, so is communication-optimal when $p \lesssim n$. To our knowledge, lowest known communication complexity for solving the lasso is at least $mp \frac{\sigma_{\max}(\Sigma)}{\sigma_{\min}(\Sigma)} \log \frac{1}{\epsilon}$, for any desired accuracy $\epsilon \gtrsim \sqrt{\frac{s \log p}{n}}$ (Agarwal et al., 2012). This communication cost is larger than our algorithm by a multiplicative factor of $\frac{\sigma_{\max}(\Sigma)}{\sigma_{\min}(\Sigma)} \log \frac{1}{\epsilon}$, which is substantially larger when the problem is poorly conditioned.

In light of the fact that our approach is essentially optimal in terms of communication cost, we turn to the computational complexity of our method. The most intensive step of our approach is the computation of $\hat{\Theta}$. To compute one row of $\hat{\Theta}$ requires solving an optimization problem whose cost is equivalent to a lasso in dimension p with n samples. For the purpose of comparison, let us assume that the lasso solver performs T iterations. Thus the complexity of solving a lasso in dimension p with n samples is $O(npT)$. In the simple approach of Section 2 where each machine computes its own $\hat{\Theta}$, the parallel runtime is $O(np^2T)$. However using the approach of Section 3, each machine is only computing p/m rows of $\hat{\Theta}$. This brings down the parallel runtime to $O(\frac{np^2T}{m})$. In comparison, the cost of solving the lasso using a state-of-art optimization method such as Agarwal et al. (2012) has parallel runtime $O(npT)$, so our computational cost is larger by a factor of $O(\frac{p}{m})$. It is reasonable to think of $\frac{p}{m}$ as a constant, since as the number of variables p increases the dataset sizes increases and we will be forced to use a larger cluster size m due to memory constraints on a single machine. Although our computational complexity is larger in the distributed setting, the dominant factor is often bottlenecked by the communication and latency limitations, rather than local computation.

Appendix A. Proofs of Lemmas

Proof [Proof of Lemma 2] Let $z_i = \Sigma^{-\frac{1}{2}} x_i$. The generalized coherence between X and Σ^{-1} is given by

$$|||\Sigma^{-1}\hat{\Sigma} - I|||_\infty = |||\frac{1}{n}\sum_{i=1}^n (\Sigma^{-\frac{1}{2}} z_i)(\Sigma^{\frac{1}{2}} z_i)^T - I|||_\infty,$$

where $|||X|||_\infty$ is the maximum entry of a X. Each entry of $\frac{1}{n}\sum_{i=1}^n (\Sigma^{-\frac{1}{2}} z_i)(\Sigma^{\frac{1}{2}} z_i)^T - I$ is a sum of independent subexponential random variables. Their subexponential norms are bounded by

$$\|(\Sigma^{-\frac{1}{2}} z_i)_j (\Sigma^{\frac{1}{2}} z_i)_k - \mathbf{1}\{j=k\}\|_{\psi_1} \leq 2\|(\Sigma^{-\frac{1}{2}} z_i)_j (\Sigma^{\frac{1}{2}} z_i)_k\|_{\psi_1},$$

where $\|X\|_{\psi_1}$ and $\|Y\|_{\psi_2}$ are the sub-exponential and sub-Gaussian norms of the random variables X and Y. Recall for any two subgaussian random variables X, Y, we have

$$\|XY\|_{\psi_1} \leq 2\|X\|_{\psi_2}\|Y\|_{\psi_2}.$$

Thus

$$\|(\Sigma^{-\frac{1}{2}} z_i)_j (\Sigma^{\frac{1}{2}} z_i)_k - \delta_{j,k}\|_{\psi_1} \leq 4\|(\Sigma^{-\frac{1}{2}} z_i)_j\|_{\psi_2}\|(\Sigma^{\frac{1}{2}} z_i)_k\|_{\psi_2} \leq 4\sqrt{\kappa}\sigma_x^2,$$

where $\sigma_x = \|z_i\|_{\psi_2}$. By a Bernstein-type inequality,

$$\mathbf{Pr}\Big(\frac{1}{n}\sum_{i=1}^n (\Sigma^{-\frac{1}{2}} z_i)_j (\Sigma^{\frac{1}{2}} z_i)_k - \delta_{j,k} \geq t\Big) \leq 2e^{-c_1 \min\{\frac{nt^2}{\tilde{\sigma}_x^4}, \frac{nt}{\tilde{\sigma}_x^2}\}},$$

where $c_1 > 0$ is a universal constant and $\tilde{\sigma}_x^2 := 4\sqrt{\kappa}\sigma_x^2$. Since $\tilde{\sigma}_x^4 n > \log p$, we set $t = \frac{2\tilde{\sigma}_x^2}{\sqrt{c_1}}\big(\frac{\log p}{n}\big)^{\frac{1}{2}}$ to obtain

$$\mathbf{Pr}\Big(\frac{1}{n}\sum_{i=1}^n (\Sigma^{-\frac{1}{2}} z_i)_j (\Sigma^{\frac{1}{2}} z_i)_k - \delta_{j,k} \geq \frac{2\tilde{\sigma}_x^2}{\sqrt{c_1}}\Big(\frac{\log p}{n}\Big)^{\frac{1}{2}}\Big) \leq 2p^{-4}.$$

We obtain the stated result by taking a union bound over the p^2 entries of $\frac{1}{n}\sum_{i=1}^n (\Sigma^{-\frac{1}{2}} z_i)(\Sigma^{\frac{1}{2}} z_i)^T - I$. ■

Proof [Proof of Lemma 5] By Vershynin (2010), Proposition 5.10,

$$\mathbf{Pr}\Big(\frac{1}{n}|x_j^T\epsilon| > t\Big) \leq e\exp\Big(-\frac{c_2 n^2 t^2}{\sigma_y^2\|x_j^T\|_2^2}\Big) \leq e\exp\Big(-\frac{c_2 n^2 t^2}{\sigma_y^2 \max_{j\in[p]}\hat{\Sigma}_{j,j}}\Big).$$

We take a union bound over the p components of $\frac{1}{n}X^T\epsilon$ to obtain

$$\mathbf{Pr}\Big(\frac{1}{n}\|X^T\epsilon\|_\infty > t\Big) \leq e\exp\Big(-\frac{c_2 n^2 t^2}{\sigma_y^2 \max_{j\in[p]}\hat{\Sigma}_{j,j}} + \log p\Big).$$

We set $t = \max_{j\in[p]}\hat{\Sigma}_{j,j}^{\frac{1}{2}}\sigma_y\big(\frac{3\log p}{c_2 n}\big)^{\frac{1}{2}}$ to obtain the desired conclusion. ■

Proof [Proof of Lemma 7] We start by substituting the linear model into (1):

$$\hat{\beta}^d = \hat{\beta} + \frac{1}{n}\hat{\Theta}X^T(y - X\hat{\beta}) = \beta^* + \hat{\Theta}\hat{\Sigma}(\beta^* - \hat{\beta}) + \frac{1}{n}\hat{\Theta}X^T\epsilon.$$

By adding and subtracting $\hat{\Delta} = \beta^* - \hat{\beta}$, we obtain

$$\hat{\beta}^d = \beta^* + \frac{1}{n}\hat{\Theta}X^T(y - X\hat{\beta}) = \beta^* + (\hat{\Theta}\hat{\Sigma} - I)(\beta^* - \hat{\beta}) + \frac{1}{n}\hat{\Theta}X^T\epsilon.$$

We obtain the expression of $\hat{\beta}^d$ by setting $\hat{\Delta}$.

To show $\|\hat{\Delta}\|_\infty \leq \frac{3\delta}{\mu}s\lambda$, we apply Hölder's inequality to each component of $\hat{\Delta}$ to obtain

$$|(\hat{\Theta}\hat{\Sigma} - I)(\beta^* - \hat{\beta})| \leq \max_j \|\hat{\Sigma}m_j^T - e_j\|_\infty\|\hat{\beta} - \beta^*\|_1 \leq \delta\|\hat{\beta} - \beta^*\|_1, \tag{10}$$

where δ is the generalized incoherence between X and $\hat{\Theta}$. By Lemma 6, $\|\hat{\beta} - \beta^*\|_1 \leq \frac{3}{\mu}s\lambda$. We combine the bound on $\|\hat{\beta} - \beta^*\|_1$ with (10) to obtain the stated bound on $\|\hat{\Delta}\|_\infty$. ■

Proof [Proof of Lemma 10] By Lemma 7,

$$\bar{\beta} - \beta^* = \frac{1}{N}\sum_{k=1}^{m} \hat{\Theta}_k X_k^T \epsilon_k + \frac{1}{m}\sum_{k=1}^{m} \hat{\Delta}_k.$$

We take norms to obtain

$$\|\bar{\beta} - \beta^*\|_\infty \leq \Big\|\frac{1}{N}\sum_{k=1}^{m} \hat{\Theta}_k X_k^T \epsilon_k\Big\|_\infty + \frac{1}{m}\sum_{k=1}^{m} \|\hat{\Delta}_k\|_\infty.$$

We focus on bounding the first term. Let $a_j^T := e_j^T \begin{bmatrix}\hat{\Theta}_1 X_1^T & \dots & \hat{\Theta}_m X_m^T\end{bmatrix}$. By Vershynin (2010), Proposition 5.10,

$$\mathbf{Pr}\Big(|\frac{1}{N} a_j^T \epsilon| > t\Big) \leq e\exp\Big(-\frac{c_2 N^2 t^2}{\|a_j\|_2^2 \sigma_y^2}\Big)$$

for some universal constant $c_2 > 0$. Further,

$$\|a_j\|_2^2 = \sum_{k=1}^{m} \|X_k \hat{\Theta}_k^T e_j\|_2^2 = n\sum_{k=1}^{m} \big(\hat{\Theta}_k \hat{\Sigma}_k \hat{\Theta}_k^T\big)_{j,j} \leq c_\Omega N,$$

where $c_\Omega := \max_{j\in[p],k\in[m]} \big(\hat{\Theta}_k \hat{\Sigma}_k \hat{\Theta}_k^T\big)_{j,j}$. By a union bound over $j \in [p]$,

$$\mathbf{Pr}\Big(\max_{j\in[p]} |\frac{1}{N} a_j^T \epsilon| > t\Big) \leq e\exp\Big(-\frac{c_2 N t^2}{c_\Omega \sigma_y^2} + \log p\Big).$$

We set $t = \sigma_y\big(\frac{2c_\Omega \log p}{c_2 N}\big)^{\frac{1}{2}}$ to deduce

$$\mathbf{Pr}\Big(\max_{j\in[p]} |\tfrac{1}{N} a_j^T \epsilon| \geq \sigma_y\big(\tfrac{2c_\Omega \log p}{c_2 N}\big)^{\frac{1}{2}}\Big) \leq ep^{-1}.$$

We turn our attention to bounding the second term. By Lemma 5 and a union bound over $j \in [p]$, when we set

$$\lambda_1 = \dots = \lambda_m = \lambda := \max_{j\in[p],k\in[m]} \big((\hat{\Sigma}_k)_{j,j}\big)^{\frac{1}{2}} \sigma_y\Big(\frac{3\log p}{c_2 n}\Big)^{\frac{1}{2}},$$

we have $\frac{1}{n}\|X_k^T \epsilon\|_\infty \leq \lambda$ for any $k \in [m]$ with probability at least $1 - \frac{em}{p^2} \geq 1 - ep^{-1}$. By Lemma 7, when

1. $\{\hat{\Sigma}_k\}_{k\in[m]}$ satisfy the RE condition on $\mathcal{C}(\mathrm{supp}(\beta^*))$ with constant μ_l,
2. $\{(\hat{\Sigma}_k, \hat{\Theta}_k)\}_{k\in[m]}$ have generalized incoherence $c_{\mathrm{GC}}\big(\frac{\log p}{n}\big)^{\frac{1}{2}}$,

the second term is at most $\frac{3\sqrt{3}}{\sqrt{c_2}} \frac{c_{\mathrm{GC}} c_\Sigma}{\mu_l} \sigma_y \frac{s\log p}{n}$. We put the pieces together to obtain

$$\|\bar{\beta} - \beta^*\|_\infty \leq \sigma_y\Big(\frac{2c_\Omega \log p}{c_2 N}\Big)^{\frac{1}{2}} + \frac{3\sqrt{3}}{\sqrt{c_2}} \frac{c_{\mathrm{GC}} c_\Sigma}{\mu_l} \sigma_y \frac{s\log p}{n},$$

■

Proof [Proof of Lemma 10] We start with the conclusion of Lemma 10:

$$\|\bar{\beta} - \beta^*\|_\infty \leq \sigma_y\Big(\frac{2c_\Omega \log p}{c_2 N}\Big)^{\frac{1}{2}} + \frac{3\sqrt{3}}{\sqrt{c_2}} \frac{c_{\mathrm{GC}} c_\Sigma}{\mu_l} \sigma_y \frac{s\log p}{n}.$$

First, we show that the two constants $c_\Omega = \max_{j\in[p],\, k\in[m]} (\hat{\Theta}_k \hat{\Sigma}_k \hat{\Theta}_k^T)_{j,j}$ and $c_\Sigma := \max_{j\in[p],k\in[m]} ((\hat{\Sigma}_k)_{j,j})^{\frac{1}{2}}$ are bounded with high probability.

Lemma 23 *Under (A1),*

$$\mathbf{Pr}\big(\max_{j\in[p]} \Sigma_j^{-1}\hat{\Sigma}\Sigma_j^{-1} > 2\max_{j\in[p]} \Sigma_{j,j}^{-1}\big) \leq 2pe^{-c_1 \min\{\frac{n}{\sigma_x^2}, \frac{n}{\sigma_x}\}}$$

for some universal constant $c_1 > 0$.

Proof [Proof of Lemma 23] We express

$$\Sigma_{j,\cdot}^{-1}\hat{\Sigma}\Sigma_{j,\cdot}^{-1} = \Sigma_{j,\cdot}^{-1}\hat{\Sigma}\Sigma_{j,\cdot}^{-1} - \Sigma_{j,j}^{-1} + \Sigma_{j,j}^{-1} = \frac{1}{n}\sum_{i=1}^{n} (x_i^T \Sigma_{\cdot,j}^{-1})^2 - \Sigma_{j,j}^{-1} + \Sigma_{j,j}^{-1}.$$

Since the subgaussian norm of $z_i = \Sigma^{-\frac{1}{2}} x_i$ is σ_x, $x_i^T \Sigma_{\cdot,j}^{-1}$ is also subgaussian with subgaussian norm bounded by

$$\|x_i^T \Sigma_{\cdot,j}^{-1}\|_{\psi_2} \leq \|z_i\|_{\psi_2} \|\Sigma_{\cdot,j}^{-\frac{1}{2}}\|_2 \leq \sigma_x (\Sigma_{j,j}^{-1})^{\frac{1}{2}}.$$

We recognize $\frac{1}{n}\sum_{i=1}^{n} (x_i^T \Sigma_{\cdot,j}^{-1})^2 - \Sigma_{j,j}^{-1}$ as a sum of *i.i.d.* subexponential random variables with subexponential norm bounded by

$$\|(x_i^T \Sigma_{\cdot,j}^{-1})^2 - \Sigma_{j,j}^{-1}\|_{\psi_1} \leq 2\|(x_i^T \Sigma_{\cdot,j}^{-1})^2\|_{\psi_1} \leq 4\|x_i^T \Sigma_{\cdot,j}^{-1}\|_{\psi_2}^2 \leq 4\sigma_x^2 \Sigma_{j,j}^{-1}.$$

By Vershynin (2010), Proposition 5.16, we have

$$\mathbf{Pr}\Big(\frac{1}{n}\sum_{i=1}^{n} (x_i^T \Sigma_{\cdot,j}^{-1})^2 - \Sigma_{j,j}^{-1} > t\Big) \leq 2e^{-c_1 \min\{\frac{nt^2}{16\sigma_x^2 (\Sigma_{j,j}^{-1})^2}, \frac{nt}{4\sigma_x \Sigma_{j,j}^{-1}}\}}$$

for some absolute constant $c_1 > 0$. For $t = \Sigma_{j,j}^{-1}$, the bound simplifies to

$$\mathbf{Pr}\Big(\frac{1}{n}\sum_{i=1}^{n} (x_i^T \Sigma_{\cdot,j}^{-1})^2 - \Sigma_{j,j}^{-1} > \Sigma_{j,j}^{-1}\Big) \leq 2e^{-c_1 \min\{\frac{n}{16\sigma_x^2}, \frac{n}{4\sigma_x}\}}.$$

We take a union bound over $j \in [p]$ to obtain the stated result. ■

Since we form $\{\hat{\Theta}_k\}_{k\in[m]}$ by (2),

$$(\hat{\Theta}_k \hat{\Sigma}_k \hat{\Theta}_k^T)_{j,j} \leq \max_{j\in[p]} (\Sigma^{-1}\hat{\Sigma}_k\Sigma^{-1}))_{j,j}.$$

Lemma 23 implies

$$\max_{j\in[p]} (\Sigma^{-1}\hat{\Sigma}_k\Sigma^{-1}))_{j,j} \leq 2\max_{j\in[p]} \Sigma_{j,j}^{-1} \text{ for each } k \in [m]$$

with probability at least $1 - 2pe^{-c_1 \min\{\frac{n}{\sigma_x^2}, \frac{n}{\sigma_x}\}}$.

Lemma 24 *Under (A1),*

$$\mathbf{Pr}(\max_{j\in[p]}(\hat{\Sigma}_{j,j})^{\frac{1}{2}} > \sqrt{2}\max_{j\in[p]}(\Sigma_{j,j})^{\frac{1}{2}}) \leq 2pe^{-c_1\min\{\frac{n}{16\sigma_x^2},\frac{n}{4\sigma_x}\}}$$

for some universal constant $c_1 > 0$.

We put the pieces together to obtain the stated result:

1. By Lemma 23 (and a union bound over $k \in [m]$),

$$\mathbf{Pr}(c_\Omega \geq 2\max_j \Sigma^{-1}_{j,j}) \leq 2mpe^{-c_1\min\{\frac{n}{\sigma_x^2},\frac{n}{\sigma_x}\}}.$$

Since $m \leq p$, when $n > \frac{3}{c_1}\max\{\sigma_x^2, \sigma_x\}\log p$,

$$\mathbf{Pr}\big(c_\Omega < 2\max_j \Sigma^{-1}_{j,j}\big) \geq 1 - 2p^{-1}.$$

2. By Lemma 24 (and a union bound over $k \in [m]$),

$$\mathbf{Pr}(c_\Sigma < \sqrt{2}\max_{j\in[p]}(\Sigma_{j,j})^{\frac{1}{2}}) \geq 1 - 2mpe^{-c_1\min\{\frac{n}{16\sigma_x^2},\frac{n}{4\sigma_x}\}}.$$

When $n > \frac{3}{c_1}\max\{\sigma_x^2, \sigma_x\}\log p$, the right side is again at most $2p^{-1}$.

3. By Lemma 4, as long as

$$n > \max\{4000\tilde{s}\sigma_x^2\log(\tfrac{Cp}{\tilde{s}}),\ 8000\sigma_x^4\log p\},$$

$\hat{\Sigma}_1, \ldots, \hat{\Sigma}_m$ all satisfy the RE condition with probability at least

$$1 - 2me^{-\frac{n}{4000\sigma_x^4}} \geq 1 - 2p^{-1}.$$

4. By Lemma 2,

$$\mathbf{Pr}\big(\cap_{k\in[m]}\mathcal{E}_{\mathrm{GC}}(\hat{\Sigma}_k)\big) \geq 1 - 2p^{-2}.$$

Since $m < p$, the probability is at least $1 - 2p^{-1}$.

We apply the bounds $c_\Omega \leq 2\max_{j\in[p]}\Sigma^{-1}_{j,j}$, $c_\Sigma \leq \sqrt{2}\max_{j\in[p]}(\Sigma_{j,j})^{\frac{1}{2}}$, $c_{\mathrm{GC}} = \frac{8}{\sqrt{c_1}}\sqrt{\kappa}\sigma_x^2$, and $\mu_l = \frac{1}{2}\lambda_{\min}(\Sigma)$ to obtain

$$\|\bar{\beta} - \beta^*\|_\infty \leq \sigma_y\left(\frac{4\max_{j\in[p]}\Sigma^{-1}_{j,j}\log p}{c_2 N}\right)^{\frac{1}{2}} + \frac{48\sqrt{6}}{\sqrt{c_1 c_2}}\frac{\sqrt{\kappa}\max_{j\in[p]}(\Sigma_{j,j})^{\frac{1}{2}}}{\lambda_{\min}(\Sigma)}\sigma_x^2\sigma_y\frac{s\log p}{n}.$$

■

Proof [Proof of Lemma 24] We follow a similar argument as the proof of Lemma 23:

$$\hat{\Sigma}_{k;j,j} = \hat{\Sigma}_{j,j} = \hat{\Sigma}_{j,j} - \Sigma_{j,j} + \Sigma_{j,j} = \frac{1}{n}\sum_{i=1}^n x_{i,j}^2 - \Sigma_{j,j} + \Sigma_{j,j}.$$

Since the $z_i = \Sigma^{-\frac{1}{2}}x_i$ is subgaussian with subgaussian norm σ_x, $x_{i,j}$ is also subgaussian with subgaussian norm bounded by

$$\|x_{i,j}\|_{\psi_2} \leq \|\Sigma^{\frac{1}{2}}_{j,\cdot}z_i\|_{\psi_2} \leq \sigma_x(\Sigma_{j,j})^{\frac{1}{2}}.$$

We recognize $\hat{\Sigma}_{j,j} - \Sigma_{j,j} = \frac{1}{n}\sum_{i=1}^n x_{i,j}^2 - \Sigma_{j,j}$ as a sum of *i.i.d.* subexponential random variables with subexponential norm bounded by

$$\|\hat{\Sigma}_{j,j} - \Sigma_{j,j}\|_{\psi_1} \leq 2\|x_{i,j}^2\|_{\psi_1} \leq 4\|x_{i,j}\|_{\psi_2}^2 \leq 4\sigma_x^2\Sigma_{j,j}.$$

By Vershynin (2010), Proposition 5.16, we have

$$\mathbf{Pr}(|\hat{\Sigma}_{j,j} - \Sigma_{j,j}| > t) \leq 2e^{-c_1\min\{\frac{nt^2}{16\sigma_x^2\Sigma_{j,j}^2},\frac{nt}{\sigma_x\Sigma_{j,j}}\}}$$

for some absolute constant $c_1 > 0$. For $t = \Sigma_{j,j}$, the bound simplifies to

$$\mathbf{Pr}(|\hat{\Sigma}_{j,j} - \Sigma_{j,j}| > \Sigma_{j,j}) \leq 2e^{-c_1\min\{\frac{n}{16\sigma_x^2},\frac{n}{4\sigma_x}\}}.$$

We take a union bound over $j \in [p]$ to obtain the stated result. ■

Proof [Proof of Theorem 16] We start by substituting the linear model into (5):

$$\begin{aligned}\tilde{\beta} &= \frac{1}{m}\sum_{k=1}^m \hat{\beta}_k - \hat{\Theta}\hat{\Sigma}_k(\hat{\beta}_k - \beta^*) + \frac{1}{n}\hat{\Theta}X_k^T\epsilon_k \\ &= \frac{1}{m}\sum_{k=1}^m \hat{\beta}_k - \hat{\Theta}\hat{\Sigma}_k(\hat{\beta}_k - \beta^*) + \frac{1}{N}\hat{\Theta}X^T\epsilon.\end{aligned}$$

Subtracting β^* and taking norms, we obtain

$$\|\tilde{\beta} - \beta^*\|_\infty \leq \frac{1}{m}\sum_{k=1}^m \|(I - \hat{\Theta}\hat{\Sigma}_k)(\hat{\beta}_k - \beta^*)\|_\infty + \big\|\frac{1}{N}\hat{\Theta}X^T\epsilon\big\|_\infty. \tag{11}$$

By Vershynin (2010), Proposition 5.16, and Lemma (23), it is possible to show that

$$\big\|\frac{1}{N}\hat{\Theta}X^T\epsilon\big\|_\infty \lesssim_P \left(\frac{\log p}{N}\right)^{\frac{1}{2}}.$$

We turn our attention to the first term in (11). It's straightforward to see each term in the sum is bounded by

$$\begin{aligned}&\|(I - \hat{\Theta}\hat{\Sigma}_k)(\hat{\beta}_k - \beta^*)\|_\infty \\ &\quad\leq \|(I - \Sigma^{-1}\hat{\Sigma}_k)(\hat{\beta}_k - \beta^*)\|_\infty + \|(\Sigma^{-1} - \hat{\Theta})\hat{\Sigma}_k(\hat{\beta}_k - \beta^*)\|_\infty \\ &\quad\leq \max_{j\in[p]}\|e_j^T - \Sigma_j^{-1}\hat{\Sigma}_k\|_\infty\|\hat{\beta}_k - \beta^*\|_1 + \|\Sigma_j^{-1} - \hat{\Theta}_{j,\cdot}\|_1\|\hat{\Sigma}_k(\hat{\beta}_k - \beta^*)\|_\infty.\end{aligned}$$

We put the pieces together to deduce each term is $O\big(\frac{s_{\max}\log p}{n}\big)$:

1. By Lemmas 4, 6, 24, $\|\hat{\beta}_k - \beta^*\|_1 \lesssim_P \sqrt{s_0}\lambda_k$.
2. By Lemma 15, $\|\Sigma_j^{-1} - \hat{\Theta}_{j,\cdot}\|_1 \lesssim_P s_j\big(\frac{\log p}{n}\big)^{\frac{1}{2}}$.
3. By the triangle inequality,

$$\|\hat{\Sigma}_k(\hat{\beta}_k - \beta^*)\|_\infty \leq \Big\|\frac{1}{n}X_k^T(y_k - X_k\hat{\beta}_k)\Big\|_\infty + \Big\|\frac{1}{n}X_k^T\epsilon_k\Big\|_\infty.$$

   By the optimality conditions of the (local) lasso estimators, the first term is λ_k, and it is possible to show, by Lemma 23 and Vershynin (2010), Proposition 5.16, that the second term is $O_P\big(\big(\frac{\log p}{n}\big)^{\frac{1}{2}}\big)$.

Since $\lambda_k \asymp \big(\frac{\log p}{n}\big)^{\frac{1}{2}}$, by a union bound over $k \in [m]$, we obtain

$$\|\bar{\beta} - \beta^*\|_\infty \sim O_P\Big(\Big(\frac{\log p}{N}\Big)^{\frac{1}{2}} + \frac{s_{\max}\log p}{n}\Big),$$

where $s_{\max} := \max\{s_0, s_1, \ldots, s_p\}$. ■

Proof [Proof of Lemma 18] The result is essentially Javanmard and Montanari (2013b), Theorem 2.3 with $\hat{\Omega} = \hat{\Theta}$ given by (6). Lemma 15 shows that

$$\max_{j\in[p]} \|\hat{\Theta}_{j,\cdot} - \Sigma_j^{-1}\|_1 \lesssim_P s_j\big(\tfrac{\log p}{n}\big)^{\frac{1}{2}},$$

Since $\frac{\max_{j\in[p]} s_j^2 \log p}{n} \sim o(1)$, $\hat{\Theta}$ satisfies the conditions of Javanmard and Montanari (2013b), Theorem 2.3:

$$\|\hat{\Delta}_k\|_{(\infty,c's_0)} \lesssim_P \frac{c\sqrt{s_0}\log p}{n} \text{ for any } k \in [m],$$

The bound is uniform in $k \in [m]$ by a union bound for suitable parameters $\lambda_k \sim \big(\frac{\log p}{n}\big)^{\frac{1}{2}}$. ■

Proof [Proof of Theorem 19] We start by substituting the linear model into (5):

$$\tilde{\beta} = \frac{1}{m}\sum_{k=1}^m \hat{\Delta}_k + \frac{1}{N}\hat{\Theta}X^T\epsilon.$$

Subtracting β^* and taking norms, we obtain

$$\|\tilde{\beta} - \beta^*\|_{(\infty,c's_0)} \leq \frac{1}{m}\sum_{k=1}^m \|\hat{\Delta}_k\|_{(\infty,c's_0)} + \big\|\frac{1}{N}\hat{\Theta}X^T\epsilon\big\|_{(\infty,c's_0)}. \tag{12}$$

By Lemma 18, the first (bias) term is of order $\frac{c\sqrt{s_0}\log p}{n}$. We focus on showing the second (variance) term is of order $\big(\frac{\log p}{N}\big)^{\frac{1}{2}}$. Since the (∞, l) norm is non-increasing in l,

$$\big\|\frac{1}{N}\hat{\Theta}X^T\epsilon\big\|_{(\infty,c's_0)} \leq \big\|\frac{1}{N}\hat{\Theta}X^T\epsilon\big\|_\infty.$$

By Vershynin (2010), Proposition 5.16 and Lemma 23, it is possible to show that

$$\big\|\frac{1}{N}\hat{\Theta}X^T\epsilon\big\|_\infty \sim O_P\Big(\Big(\frac{\log p}{N}\Big)^{\frac{1}{2}}\Big).$$

Thus the second term in (12) is of order $\big(\frac{\log p}{N}\big)^{\frac{1}{2}}$. We put all the pieces together to obtain the stated conclusion. ■

Proof [Proof of Theorem 20] Since $\tilde{\beta}^{ht} - \beta^*$ is $2s_0$-sparse,

$$\|\tilde{\beta}^{ht} - \beta^*\|_2^2 \lesssim s_0\|\tilde{\beta}^{ht} - \beta^*\|_{(\infty,c's_0)}^2$$

or, equivalently,

$$\|\tilde{\beta}^{ht} - \beta^*\|_2 \lesssim \sqrt{s_0}\|\tilde{\beta}^{ht} - \beta^*\|_{(\infty,c's_0)}.$$

By the triangle inequality,

$$\begin{aligned}\|\tilde{\beta}^{ht} - \beta^*\|_{(\infty,c's_0)} &\leq \|\tilde{\beta}^{ht} - \tilde{\beta}\|_{(\infty,c's_0)} + \|\tilde{\beta} - \beta^*\|_{(\infty,c's_0)} \\ &\leq 2\|\tilde{\beta} - \beta^*\|_{(\infty,c's_0)},\end{aligned}$$

where the second inequality is by the fact that thresholding at $t = |\tilde{\beta}|_{(c's_0)}$ minimizes $\|\beta - \beta^*\|_{(\infty,c's_0)}$ over $c's_0$-sparse points β. Thus

$$\|\tilde{\beta}^{ht} - \beta^*\|_2 = O_P\Big(\Big(\frac{s_0\log p}{N}\Big)^{\frac{1}{2}} + \frac{s_0\log p}{n}\Big).$$

To complete the proof, we observe that the estimation error bound of $\tilde{\beta}^{ht}$ in the ℓ_1 norm follows by the fact that $\tilde{\beta}^{ht} - \beta^*$ is $2s_0$-sparse. ■

References

Alekh Agarwal, Sahand Negahban, Martin J Wainwright, et al. Fast global convergence of gradient methods for high-dimensional statistical recovery. *The Annals of Statistics*, 40 (5):2452–2482, 2012.

Heather Battey, Jianqing Fan, Han Liu, and Junwei Lu. Splitotic analysis for distributed estimation and hypothesis testing. *preprint (personal communication)*, 2015.

Alexandre Belloni, Victor Chernozhukov, and Christian Hansen. Inference for high-dimensional sparse econometric models. *arXiv preprint arXiv:1201.0220*, 2011.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

Mark Braverman, Ankit Garg, Tengyu Ma, Huy L Nguyen, and David P Woodruff. Communication lower bounds for statistical estimation problems via a distributed data processing inequality. *arXiv preprint arXiv:1506.07216*, 2015.

Peter Bühlmann and Sara Van De Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer, 2011.

Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. Optimal distributed online prediction using mini-batches. *The Journal of Machine Learning Research*, 13(1):165–202, 2012.

John C Duchi, Alekh Agarwal, and Martin J Wainwright. Dual averaging for distributed optimization: convergence analysis and network scaling. *Automatic Control, IEEE Transactions on*, 57(3):592–606, 2012.

John C Duchi, Michael I Jordan, Martin J Wainwright, and Yuchen Zhang. Optimality guarantees for distributed statistical estimation. *arXiv preprint arXiv:1405.0782*, 2014.

Ankit Garg, Tengyu Ma, and Huy L Nguyen. Lower bound for high-dimensional statistical learning problem via direct-sum theorem. *arXiv preprint arXiv:1405.1665*, 2014.

Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical learning with sparsity: the lasso and its generalizations*. CRC Press, 2015.

Adel Javanmard and Andrea Montanari. Confidence intervals and hypothesis testing for high-dimensional regression. *arXiv preprint arXiv:1306.3171*, 2013a.

Adel Javanmard and Andrea Montanari. Nearly optimal sample size in hypothesis testing for high-dimensional regression. *arXiv preprint arXiv:1311.0274*, 2013b.

Ryan Mcdonald, Mehryar Mohri, Nathan Silberman, Dan Walker, and Gideon S Mann. Efficient large-scale distributed training of conditional maximum entropy models. In *Advances in Neural Information Processing Systems*, pages 1231–1239, 2009.

Sahand N Negahban, Pradeep Ravikumar, Martin J Wainwright, and Bin Yu. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. *Statistical Science*, 27(4):538–557, 2012.

Garvesh Raskutti, Martin J Wainwright, and Bin Yu. Restricted eigenvalue properties for correlated gaussian designs. *J. Mach. Learn. Res.*, 11:2241–2259, 2010.

Jonathan Rosenblatt and Boaz Nadler. On the optimality of averaging in distributed statistical learning. *arXiv preprint arXiv:1407.2724*, 2014.

Mark Rudelson and Shuheng Zhou. Reconstruction from anisotropic random measurements. *Information Theory, IEEE Transactions on*, 59(6):3434–3447, 2013.

Sara van de Geer, Peter Bühlmann, Ya'acov Ritov, and Ruben Dezeure. On asymptotically optimal confidence regions and tests for high-dimensional models. *arXiv preprint arXiv:1303.0518*, 2013.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Cun-Hui Zhang and Stephanie S Zhang. Confidence intervals for low dimensional parameters in high dimensional linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 76(1):217–242, 2014.

Yuchen Zhang, John C Duchi, and Martin J Wainwright. Communication-efficient algorithms for statistical optimization. *Journal of Machine Learning Research*, 14:3321–3363, 2013.

Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J Smola. Parallelized stochastic gradient descent. In *Advances in Neural Information Processing Systems*, pages 2595–2603, 2010.

Journal of Machine Learning Research 18 (2017) 1-36 Submitted 2/16; Revised 11/16; Published 1/17

Improving Variational Methods via Pairwise Linear Response Identities

Jack Raymond jack.raymond@physics.org
Dipartimento di Fisica, La Sapienza University of Rome
Piazzale Aldo Moro 5
Rome, Italy

Federico Ricci-Tersenghi federico.ricci@uniroma1.it
Dipartimento di Fisica, INFN–Sezione di Roma1 and CNR–Nanotec,
La Sapienza University of Rome
Piazzale Aldo Moro 5
Rome, Italy

Editor: Manfred Opper

Abstract

Inference methods are often formulated as variational approximations: these approximations allow easy evaluation of statistics by marginalization or linear response, but these estimates can be inconsistent. We show that by introducing constraints on covariance, one can ensure consistency of linear response with the variational parameters, and in so doing inference of marginal probability distributions is improved. For the Bethe approximation and its generalizations, improvements are achieved with simple choices of the constraints. The approximations are presented as variational frameworks; iterative procedures related to message passing are provided for finding the minima.

Keywords: variational inference, graphical models, message passing algorithms, statistical physics, linear response

1. Introduction

Given a probability distribution $p(x)$, estimation of marginal probability distributions such as $p(x_i)$ and $p(x_i; x_j)$ is one of the most important inference tasks addressed in graphical models, alongside estimation of the maximum probability state and the log partition function (see Wainwright and Jordan, 2008; Mezard and Montanari, 2009; MacKay, 2004). The challenge is addressed in many research fields by a variety of methods. In Boltzmann machine learning and probabilistic independent component analysis the expectation-maximization algorithm requires such estimates (see Wainwright and Jordan, 2008; Miskin and MacKay, 2000). In heuristic optimization, a branch and bound search (or decimation procedure) over a high dimensional space can be made more efficient, by branching on x_i (or some small set of variables) in an informed manner using approximate probabilities (see Montanari et al., 2007). In channel coding we wish to determine the likely state of a bit sent over a noisy channel, which can be inferred with a measure of certainty from the marginal probability (see Richardson and Urbanke, 2008). In statistical physics, marginal probability distributions provide insight into phase transitions and thermodynamic phases (see Parisi, 1987).

Approximation of marginal probability distributions (called marginals henceforth) with high accuracy is NP-hard even in the case of Ising spins (binary variables) with pairwise interactions (see Dagum and Luby, 1993; Long and Servedio, 2010), but in practice, many schemes might be applied successfully. Approximate inference of marginal distributions is often performed by Markov chain Monte Carlo (MCMC) procedures (see Andrieu et al., 2003). These methods are a workhorse of inference, but have some disadvantages: the estimates are achieved with an accuracy that decays only slowly with time resources (exploiting the central limit theorem), the result is stochastic, and takes a non-parametric form. For these reasons variational approximations are often preferred, the price being a (difficult to quantify) bias in the approximations (see Wainwright and Jordan, 2008; Mezard and Montanari, 2009; MacKay, 2004).

In a basic variational approximation an intractable probability distribution $p(x)$ is approximated by a tractable one $q(x)$, the parameters of q are determined by minimizing the Kullback Leibler (KL) divergence $D_{KL}[p||q]$. The challenge of marginalization is thus replaced by two linked challenges: appropriate construction of q, and minimization of a KL-divergence. An example of a tractable distribution is a factorized one: $q(x) = \prod_i q(x_i)$, which leads to a mean-field variational approximation; KL-divergence could then typically be minimized by an iteration of fixed point equations (see Wainwright and Jordan, 2008; Mezard and Montanari, 2009). It is common for the estimates obtained by variational approximations to be over-confident, the uncertainty in some variables is reduced since the structure of the approximation q discounts some sources of variance.

A given variational framework may be minimized by several algorithms, and it is interesting that many famous heuristic algorithms developed independently of variational frameworks have been shown to be particular solutions to variational approximations. Most notably loopy belief propagation has been shown to be one method to solve the Bethe variational approximation, and expectation propagation was shown to be one method to solve the expectation consistent variational approximation. This connection to variational frameworks has allowed interesting insight into algorithm construction, proofs of solution existence and convergence (see Yedidia et al., 2005; Wainwright and Jordan, 2008; Yuille, 2002).

We have so far identified one mechanism by which to estimate marginals: we can directly marginalize our approximation $q(x)$, which is tractable by construction. These same estimates (when not extremal) can be found as first derivatives of the variational function (in the above description, the KL-divergence), as we later show. Second order derivatives of the KL-divergence are also possible and can give covariance estimates. These covariance estimates are called linear responses, since they measure the response of some expectation (the first derivative estimate) to an infinitesimal linear perturbation (the second derivative). Linear response and marginalization estimates are tractable for variational approximations; and for some statistics, we can use either method to obtain an estimate for the same quantity.

©2017 Jack Raymond and Federico Ricci-Tersenghi.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-070.html.

We will consider variational approximations applied to a model of N discrete variables x_i defined by probability [1]

$$p(x) = \frac{1}{Z} \prod_{a=1}^{M} \psi_a(x_a) \,, \tag{1}$$

where ψ_a are the potentials (also called factors) and are non-negative functions of the variables indexed by subset a, $x_a = \{x_i : i \in a\}$, Z is the partition function. Probabilities of this kind can be represented as a factor graph (see Wainwright and Jordan, 2008; Mezard and Montanari, 2009), as shown in Figure 1. Our aim is to demonstrate a mechanism whereby existing variational schemes can be leveraged for improved inference of marginals.

This paper considers a new self-consistent approximation to improve variational methods, with an emphasis on the Bethe approximation and its generalizations (called region-based, or cluster-variational, approximations). We propose the addition of constraints requiring the consistency of estimates obtained via direct marginalization and linear response. We minimize the variational function subject to an agreement of these estimates and show that the resulting unique estimate is an improvement on the two estimates that are obtained without the constraints.

1.1 Literature Review

The linear response has been leveraged to improve estimates of marginals in a variety of problems, the idea originating in statistical physics (see Parisi, 1987; Opper and Winther, 2003; Welling and Teh, 2004).

Physics approaches often aim to improve understanding of phase transitions for problem classes in the limit of a large number of variables (see Parisi, 1987), rather than in development of algorithmic approaches to solve particular finite instances. An early application of linear response was the self-consistent Ornstein-Zernike approximation (SCOZA), proposed by Høye and Stell (1977), which was later applied by Dickman and Stell (1996) to simple graphical models. The SCOZA has been applied to disordered models where marginals are not homogeneous, for example to the random field Ising model in Kierlik, Rosinberg, and Tarjus (1999), but in the service of estimating globally averaged and disorder averages quantities, and never in such a way as to understand particular single variable or pairwise marginals, which is the question we address.

Opper and Winther (2001) proposed the adaptive-TAP approach as an extension of the standard Thouless-Anderson-Palmer mean-field method; in the original formulation of this method, a self-consistency relation between the linear response and magnetizations of a mean-field approximation were reconciled to arrive at a more advanced mean-field theory. Opper and Winther (2005) and Winther and Opper (2005) later reinterpreted this method as a special case of expectation consistent approximate inference, that made a connection between moment-matching algorithms such as expectation propagation and variational frameworks, as well as expanding the range of applications. Expansions of the expectation consistent approximation to mitigate for errors on higher order cumulants have

1. For brevity in expressions we will use the notation x both for a random variable and its realization, relying on context for the distinction.

shown promise in Opper, Paquet, and Winther (2013) and Paquet, Winther, and Opper (2009), but at this point come with few theoretical guarantees.

In the context of machine learning, other related approaches for improving mean-field estimation have been successfully demonstrated in Kappen and Rodriguez (1998) and Giordano, Broderick, and Jordan (2015). Mean-field variational Bayes is an important application of variational approximations, but the absence of an accurate understanding of covariance in the model parameters had been a weakness. Recently it was shown by Giordano, Broderick, and Jordan (2015) that linear response could be used to more accurately estimate these quantities.

The Bethe variational approximation is also an important approximation in the context of sparse graphical models, for which loopy belief propagation (LBP) is the most famous algorithm. Linear response has also been used to improve this approximation, examples include Montanari and Rizzo (2005) and Mooij, Wemmenhove, Kappen, and Rizzo (2007). A large part of this development has been through loop-correction algorithms since the failure of the approximation is known to be related to loops in the graphical model representation. There also exist elegant loop correction methods not relying on the linear response: libDAI is a code repository that has collected some of the methods together (see Mooij, 2010), we developed our methods based on this library, in particular, the implementation of Heskes, Albers, and Kappen (2003). An expansion about the loop free approximation was developed by Chertkov and Chernyak (2006), but is cumbersome when many loops are present.

Several papers related to an extension of the Bethe approximation were published by Yasuda and Tanaka (2013), Raymond and Ricci-Tersenghi (2013b),Raymond and Ricci-Tersenghi (2013a) and Yasuda (2013). The idea was very similar to that of adaptive-TAP, to minimize the variational function subject to the constraint of statistical consistency. When applied to the mean-field approximation it was realized these methods were equivalent to adaptive-TAP, but in the context of Bethe and region-based free energies improved performance was identified. In Raymond and Ricci-Tersenghi (2013b) and Huang and Kabashima (2013) linear response frameworks were also leveraged in the reverse direction to solve the inverse-Ising problem (inferring parameters from statistics).

These variational frameworks were applied initially to Bethe and mean-field approximations on pairwise binary state models, in Yasuda (2013) an extension to general discrete states was provided, whereas Raymond and Ricci-Tersenghi (2013b) applied the technique to region-based variational frameworks and a broader range of constraint types. In this paper, we consider generic discrete alphabets, region-based free energies (inclusive of the Bethe approximation) and both single-variable and pairwise variable consistency constraints. Building on a belief propagation approach we derive tools for minimizing constrained variational free energies.

1.2 Outline

In Section 2, we define a set of variational approximations and motivate the inclusion of additional constraints. In Section 3, we describe methods to minimize the free energy subject to these constraints. In Section 4, we compare the performance of constrained approximations against exact results on some standard models, demonstrating a significant advantage in many cases. In Section 5, we discuss our findings in the context of all experimental results

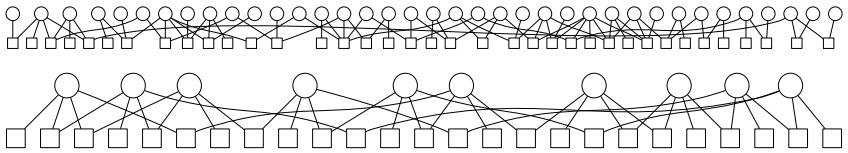


Figure 1: The constrained variational approximations we present can be applied to models with multi-variable interactions, as represented by factor graphs. Two examples are shown. Top: the alarm network is a well-known toy example of a Bayesian net, here represented as a factor graph. Squares denote factors $\psi_a(x_a)$, which act over subsets of variables x_a, each variable represented by a circle. Bottom: $N = 10$ variables interacting according to a random cubic graph is represented, each coupling (J) is represented by a factor with two connections, each field (h) by a singly connected factor.

and other insights gained, before concluding in Section 6. Appendices include exact expressions for the fully connected ferromagnet example, pseudocode and algorithmic details, proofs of convergence for some methods, discussion of solution existence and convergence, and how to select constraints for inclusion.

2. Constrained Variational Approximations

Variational free energy approximations are powerful tools for approximate inference (see Yedidia et al., 2005; Opper and Winther, 2005; Wainwright and Jordan, 2008; Mezard and Montanari, 2009). We introduce in this section the mean-field, Bethe, and region-based (also called Kikuchi) approximations. A set of simplified expressions appropriate to the Bethe approximation for an Ising model is given alongside the general expressions. We first introduce a generalization of the probability over the N variables x, introducing auxiliary parameters ν,

$$p_\nu(x) = \frac{1}{Z(\nu)} \prod_{a=1}^{M} \psi_a(x_a) \prod_{i=1}^{N} \left[\prod_{y} \exp\left(\nu_{i,y} \delta_{x_i,y}\right) \right] . \tag{2}$$

This coincides with (1) in the limit $\nu \to 0$. The product on y is over all possible states of x_i. For our method to apply the probability needs to be differentiable with respect to the parameters ν, though modifications allow for the more general case[2]. The choice of statistics $\{\delta_{x_i,y}\}$ simplifies our presentation, but more generally we might consider a set of single variable functions $\{\phi_y(x_i)\}$, this is discussed in Appendix B.2.

A Boltzmann machine will be presented as a running example. The model[3] has Ising spin variables $x \in \{-1, 1\}^N$, fields h and pairwise couplings J. A connected graphical model will be assumed for notational simplicity, so each variable has connectivity $k_i \geq 1$, and a unique connected component exists. If an edge set $E = \{(i, j)\}$ specifies the interacting

2. Special care should be taken in cases where $\psi_a(x_a) = 0$ for some x_a, in such cases it may not be meaningful to perturb by ν. Furthermore, we note that ν are a redundant set of parameters since variation of $\sum_y \nu_{i,y}$ leaves the probability unchanged.
3. For binary variables, we will use Ising spins $s = \pm 1$ in place of binary states $b \in \{0, 1\}$. The transformation $s = 1 - 2b$ allows conversion between these two conventions.

variables, then

$$p_\nu(x) = \frac{1}{Z(\nu)} \prod_{(i,j)\in E} \exp(J_{ij} x_i x_j) \prod_i \exp[(h_i + \nu_i) x_i] , \tag{3}$$

where we use a non-redundant set of auxiliary parameters $\nu_i = \nu_{i,1} - \nu_{i,-1}$.

In a variational approximation a trial probability distribution $q(x)$ is related to the log-partition function of the full model, derived from the Kullback-Leibler divergence

$$D_{KL}[q||p_\nu] = \sum_x q(x) \left[\log q(x) - \sum_{i,y} \nu_{i,y} \delta_{x_i,y} - \sum_a \log \psi_a(x_a) \right] + \log Z(\nu) . \tag{4}$$

The variational free energy (VFE) is defined

$$F_\nu(q) = D_{KL}[q||p_\nu] - \log Z(\nu) , \tag{5}$$

which is the tractable part of (4). The optimal variational parameters q^* are those minimizing (5).

In the simplest mean-field approximation a factorized variational form is considered $q(x) = \prod_{i=1}^N q_i(x_i)$. The parameters $\{q_i\}$ are precisely marginal distributions on single variables. Iterative methods are often successful in minimizing the VFE.

In another class of approximations the entropy term in the VFE, $-\sum_x q(x) \log q(x)$, is decomposed as a truncated sum of marginal entropies. In the Bethe approximation a redundant set of marginal probability distributions $\{q_a(x_a); q_i(x_i)\}$[4] are introduced in one-to-one correspondence with the model factors and variables, and the entropy is approximated as

$$-\sum_x q(x) \log q(x) \approx -\sum_a \sum_{x_a} q_a(x_a) \log q_a(x_a) - \sum_i (1 - k_i) \sum_{x_i} q_i(x_i) \log q_i(x_i) , \tag{6}$$

with k_i equal to the variable connectivity (the number of factors in which variable i participates). The reason this approximation may improve upon mean-field is that through $q_a(x_a)$ some correlations amongst variables may be explicitely represented, which are absent in the factorial form of mean-field. The variational parameters, q_a and q_i, are referred to as beliefs.

The Bethe approximation is a special case of the more general region-based approximation (see Yedidia et al., 2005), where the entropy approximation is implied by a choice of outer regions. Each outer region is defined by a set of variables x_α, and a generalized factor $\psi_\alpha(x_\alpha)$ that describes the interactions between those variables. The outer regions must be chosen to span all variables, and the factors (as well as auxiliary parameters ν) can be distributed amongst the generalized factors such that $\prod_\alpha \psi_\alpha(x) \propto p_\nu(x)$. Some examples of region selections discussed in this paper are shown in Figure 2.

The entropy approximation is implied by the choice of outer regions: it is the sum of the entropy on the outer regions α corrected by a weighted sum of entropies on region

4. $q_A(x_A)$ denotes a variational parameter, that can be interpreted as an approximation to the marginal probability $p(x_A)$.

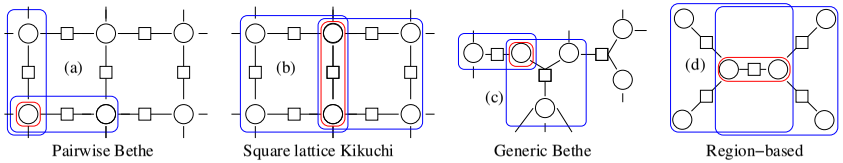


Figure 2: In a region-based approximation (Bethe is a special case), the assignment of outer regions determines the entropy approximation. Sensible choices collect nearby sets of strongly dependent variables. A region graph has outer regions (blue) and intersection regions (red), such that every factor is associated to exactly one outer region; the set of factors in a region α define an auxiliary interaction (8). Selection (a) is a Bethe approximation, the outer regions are pairs of variables, that intersect on single variables. (b) Alternatively, for a square lattice, we might consider outer regions of 4 variables (the central interaction could be assigned to either the right or left region), that intersect on pairs of variables, which in turn intersect on single variables. This approximation is powerful for square lattice models (see Raymond and Ricci-Tersenghi, 2013b; Domínguez et al., 2011; Lage-Castellanos et al., 2013). (c) The Bethe approximation on a factor graph with mixed (multi-variable) interaction types has outer regions containing exactly one factor and its variables, the intersections are single variables. (d) On a locally tree-like graph, we can also make an interesting approximation: outer regions are stars that intersect on edge regions. This approximation relates to the loop-correction algorithm of Montanari and Rizzo (2005).

intersections (β). The region based free energy is defined

$$F_\nu(q) = \sum_\alpha \sum_{x_\alpha} q_\alpha(x_\alpha) \log\left(\frac{q_\alpha(x_\alpha)}{\psi_\alpha(x_\alpha)}\right) + \sum_\beta c_\beta \sum_{x_\beta} q_\beta(x_\beta) \log q_\beta(x_\beta) \,, \tag{7}$$

where c_β take integer values according to a simple rule (see Yedidia et al., 2005; Heskes et al., 2003). Larger regions are capable of capturing more correlations between variables explicitely, but at a computational cost that scales (in the absence of further approximations) exponentially with region size. This trade-off determines the choice of regions.

In the Bethe approximation, the outer regions (α) are in one-to-one correspondence with the factors (a) of the model, and intersection regions are single variables. In our running example of the Boltzmann machine (3), a Bethe approximation has edges as regions. Generalized factors can be chosen as

$$\psi_{(i,j)}(x_i, x_j) = \exp\left(J_{ij} x_i x_j + \frac{h_i + \nu_i}{k_i} + \frac{h_j + \nu_j}{k_j}\right) \,. \tag{8}$$

We can define marginal variational parameters for each marginal probability in the free energy and then minimize (7) subject to local consistency constraints

$$\sum_{x_\alpha} q_\alpha(x_\alpha) = 1 \,; \qquad \forall \alpha \,, \tag{9}$$

$$\sum_{x_\alpha \setminus x_\beta} q_\alpha(x_\alpha) = q_\beta(x_\beta) \,; \qquad \forall \alpha, \forall \beta \subset \alpha \,. \tag{10}$$

Minimization of this free energy is a well-studied problem. Heuristic approaches normally lead to message passing algorithms, which are often convergent to good solutions even where guarantees of convergence are lacking. It is always possible to find minima of the region based free energy using a convex-concave procedure, which is guaranteed to converge to a local minima (see Yuille, 2002; Heskes et al., 2003).

Having found a minimum of the free energy at $q = q^*$, we can define the linear response in the beliefs to a perturbation in $\nu_{i,y}$, as

$$q^*_{\alpha,(i,y)}(x_\alpha) = \left.\frac{\partial q_\alpha(x_\alpha)}{\partial \nu_{i,y}}\right|_{q=q^*} \,.$$

Whereas the variational parameters q have an interpretation away from the fixed point, the linear response is only defined about a global (or heuristically, local) minima. Notation * will be used to denote an evaluation at such a fixed point.

The entropy approximations (6-7) can be interpreted as truncated series (see Pelizzola, 2005; Wainwright and Jordan, 2008), that can be made good either by considering sufficiently large regions (those defining a junction tree, see Wainwright and Jordan, 2008), or including loop corrections (see Chertkov and Chernyak, 2006). The made-good approaches are not tractable in many interesting models of modest scale. The region based methods most often lead to improvements over mean-field, but entropy expansion can also lead to counterintuitive features. For example, the entropy estimate can be negative under such an approximation.

2.1 Inconsistency of Covariance Approximations

The covariance of a pair of statistics ϕ_1 and ϕ_2, under probability distribution p is defined as

$$\begin{aligned} \mathbb{V}_p(\phi_1, \phi_2) &= \mathbb{E}_p(\phi_1\phi_2) - \mathbb{E}_p(\phi_1)\mathbb{E}_p(\phi_2) \,, \\ \text{with} \quad \mathbb{E}_p(\phi) &= \sum_x p(x)\phi(x) \,. \end{aligned}$$

For the case $\phi_1(x_\alpha) = \delta_{x_{i_1},y_1}$ and $\phi_2(x_\alpha) = \delta_{x_{i_2},y_2}$, we can replace the probability of interest p by q_α to obtain

$$\mathbb{V}_p(\delta_{x_{i_1},y_1}, \delta_{x_{i_2},y_2}) \approx C_{(i_1,y_1),(i_2,y_2)} \coloneqq \mathbb{V}_{q_\alpha}[\delta_{x_{i_1},y_1}, \delta_{x_{i_2},y_2}] \,.$$

C will be called the marginal approximation to the covariance, and is a function of the variational parameter q_α. The optimal value $C^* = C(q^*)$ is independent of the region α, owing to the consistency constraints (9)-(10).

Alternatively, we can begin with a second derivative identity. By introducing parameters ν conjugate to each statistics (2), we have

$$\mathbb{V}_p(\delta_{x_{i_1},y_1}, \delta_{x_{i_2},y_2}) = \frac{\partial^2 \log Z(\nu)}{\partial \nu_{i_1,y_1} \partial \nu_{i_2,y_2}} \,.$$

We obtain a tractable approximation replacing $\log Z(\nu)$ by $-F_\nu(q^*)$

$$\mathbb{V}_p(\delta_{x_{i_1},y_1}, \delta_{x_{i_2},y_2}) \approx \chi_{(i_1,y_1),(i_2,y_2)} \coloneqq \sum_{x_\alpha} q^*_{\alpha,(i_1,y_1)}(x_\alpha) \delta_{x_{i_2},y_2} \,, \tag{11}$$

for any α containing i_2, which is called the linear response estimate. χ will be called the linear response approximation to the covariance, it is defined only at the minima q^*, and is a symmetric matrix. We do not need to make explicit reference to the region α used in Eq. (11) since the linear response χ does not depend on that choice.

The name 'linear response' for the quantity χ comes from the fact it can be interpreted as

$$\chi_{(i_1,y_1),(i_2,y_2)} \approx \left.\frac{\partial \mathbb{E}_p[\delta_{x_{i_1},y_1}]}{\partial \nu_{i_2,y_2}}\right|_{\nu=0} = \left.\frac{\partial \mathbb{E}_p[\delta_{x_{i_2},y_2}]}{\partial \nu_{i_1,y_1}}\right|_{\nu=0} , \tag{12}$$

that is the linear variation of the mean value of a single variable statistic to a small perturbation in the the parameter ν conjugated to another single variable statistic.

We denote the difference of these estimates for two statistics on variables (i_1, i_2) contained in region α as

$$\Delta_{(i_1,y_1),(i_2,y_2),\alpha}(q_\alpha, q^*_{\alpha,(i_1,y_1)}) = C_{(i_1,y_1),(i_2,y_2)} - \chi_{(i_1,y_1),(i_2,y_2)} \, . \tag{13}$$

Except for some simple models $C^* - \chi$ is non-zero, exposing an inconsistency in the variational method. The best marginal approximation does not match the best linear response estimate. To decide which estimate to use, either connected correlations or linear responses, we might consider the distance of these two different estimates from the correct value:

$$\Delta^{(1)}_{(i_1,y_1),(i_2,y_2)} = C^*_{(i_1,y_1),(i_2,y_2)} - \mathbb{V}_p\left(\delta_{x_{i_1},y_1}, \delta_{x_{i_2},y_2}\right) , \tag{14}$$

$$\Delta^{(2)}_{(i_1,y_1),(i_2,y_2)} = \chi_{(i_1,y_1),(i_2,y_2)} - \mathbb{V}_p\left(\delta_{x_{i_1},y_1}, \delta_{x_{i_2},y_2}\right) . \tag{15}$$

Graphical models in which the Bethe approximation is most successful have relatively weak correlations and/or few short loops. In these cases it is known that the linear response estimate (15) improves significantly upon (14) for $i_1 \neq i_2$ (see Welling and Teh, 2004; Raymond and Ricci-Tersenghi, 2013a). However, as the approximation breaks down (due to poor approximations of loops in the graphical model), the response estimate can be much worse, even giving infinite values for bounded statistics. For pairs with $i_1 = i_2$ (called diagonal), bounds can be violated even in regimes where the approximation is good. A simple example is the model of Section 4.1 with zero field ($h = 0$): whilst it is true that $\mathbb{V}_p(x_i, x_i) \leq 1$ for any model, $\chi_{i,i} > 1$ for in the weakly coupled (high temperature) regime.

2.2 Covariance Constraints

We would like to use the linear response information, in a safe manner to select the best covariance estimate, but also to make the approximation self-consistent. Rather than simply minimizing the free energy to determine q^*, we do this in the subspace where $\{\Delta_{(i_1,y_1),(i_2,y_2),\alpha} = 0\}$ for a subset of the covariances.

An important question will be which covariances to constrain. Expansion methods indicate that adding all constraints is best when the approximation is very good (see Raymond and Ricci-Tersenghi, 2013a). More generally we wish to add important constraints in so far as it does not prevent solution existence and allows algorithmic stability as discussed in Appendices B.1-B.3.

Although general expressions are derived, experimental sections in this paper are restricted principally to the Bethe approximation, and with simple patterns of constraints

discussed in Appendix B.2. The Bethe approximation with the addition of all possible constraints (called *on and off diagonal*[5]) is considered, as well as the Bethe approximation with addition of only constraints for which $i_1 = i_2$ (*diagonal*). In Section 4.1 we also present results for the mean-field approximation with all possible constraints (since $i_1 = i_2$ in all cases, this is also called diagonal), as well as the Bethe approximation including only constraints for which $i_1 \neq i_2$ (*off-diagonal*). In other experiments, we do not present these latter two regimes since they performed consistently worse except in some narrow parameter ranges where all regimes were performing poorly.

2.3 Bethe Approximation to the Boltzmann Machine

In the case of our running example of the Boltzmann machine (3) we are considering a variation of $\nu_i = \nu_{i,1} - \nu_{i,-1}$, accordingly we can abbreviate notation everywhere (i, y) to i. In the diagonal constraint approximations, we will require consistency of $\mathbb{V}(x_i, x_i) \; \forall \; i$. In the on and off diagonal constrained approximations, we require in addition consistency of $\mathbb{V}(x_i, x_j)$ for all coupled pairs of variables.

The quantities made consistent are written more concisely as

$$\begin{aligned} \mathbb{V}_p(x_i, x_j) &\approx C_{i,j} \coloneqq \mathbb{V}_{q_{(ij)}}(x_i, x_j) \, , \\ \mathbb{V}_p(x_i, x_j) &\approx \chi_{i,j} \coloneqq \sum_{x_i x_j} q^*_{(ij),i}(x_i, x_j) x_j \, . \end{aligned}$$

When evaluated at minima of the free energy, both the approximation by marginalization C^*, and approximation by linear response χ, are symmetric.

3. Minimizing with Respect to the Constraints

We choose a Lagrangian formulation for minimizing the constrained free energy, introducing a Lagrange multiplier for each constraint connecting a linear response approximation to a marginalization approximation. Each statistic pair constraint will be associated with some unique outer region α, as this allows for a cavity heuristic that we later introduce. This association is not unique and may affect (to a limited extent) the convergence of the algorithms we will develop, but not the fixed points that might be achieved. This is discussed further in Appendix B.3. The set of constraints associated to region α are denoted $\boldsymbol{\omega}_\alpha = \{[(i_1, y_1), (i_2, y_2)]\}$ with an associated set of Lagrange multipliers $\boldsymbol{\lambda}_\alpha = \{\lambda_{(i_1,y_1),(i_2,y_2)}\}$. The constrained minimization is then achieved by minimizing the Lagrangian

$$F_\nu(q, \lambda, \chi) = F_\nu(q) + \sum_\alpha \left[\sum_{[(i_1,y_1),(i_2,y_2)] \in \boldsymbol{\omega}_\alpha} \lambda_{(i_1,y_1),(i_2,y_2)} \Delta_{\alpha,(i_1,y_1),(i_2,y_2)}(q_\alpha, q^*_{\alpha,(i_1,y_1)}) \right] , \tag{16}$$

with Lagrange multipliers set to meet the constraints. The global minima q^* for fixed λ can often be found. In cases where local minima can be avoided this is done either heuristically following a constrained loopy belief propagation (CLBP) approach, or in a more robust manner using a provably convergent method, as discussed in Appendix B. CLBP has the same (asymptotic) computational complexity as belief propagation, and is procedurally similar.

5. This terminology arises by considering which elements of the covariance matrix are constrained.

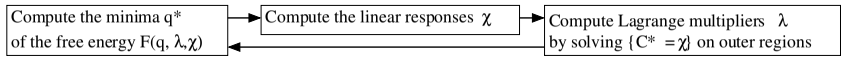


Figure 3: The basic scheme for minimizing our constrained variational free energy. The first stage is solved by a procedure closely related to belief propagation (that minimizes a convex-concave function), the second by a procedure related to susceptibility propagation (that solves a system of linear equations), and the final stage by a cavity approximation. In our experiments, we take $\lambda = 0$ as the initial condition. In the experiments presented we gradually increase or decrease T using the solution at $T \pm \delta T$ as an initial condition for the next experiment, this enables a solution to evolve continuously from a well-understood limit, but is not necessary for convergence in general.

It is shown in Appendix B.6 that given such a fixed point, whether a local or global minimum, it is subsequently relatively easy to calculate the linear response. The method we propose is procedurally similar to susceptibility propagation, originally introduced in Mezard and Mora (2008), which is a computational procedure that minimizes the variational parameters. If the number of constraints is linear in the size of the system the computational complexity is quadratic in system size. We call the linear response scheme constrained loopy susceptibility propagation (CLSP). Yasuda and Tanaka (2013) proposed a closely related approach, specific to the case of diagonal constraints in the Bethe approximation.

Suppose we also have a method for iteratively determining λ, then we can approach the problem of finding a constrained local minima by a 3-stage iterative procedure (the same as proposed in Yasuda and Tanaka (2013)), and shown schematically in Figure 3. We also experimented with other minimization schemes, but found this to be algorithmically the most stable, and also pleasing in that we move all uncertainty in convergence of the method onto the two questions: (1) does there exist a fixed point at all and (2) does the iterative scheme for λ converge. These two questions are unfortunately very difficult to answer in general. To determine λ a heuristic *cavity* method is proposed in Section 3.1.

We found that in weakly correlated regimes the CLBP and CLSP, and iterative updating of λ quickly converges. However, in some other regimes where the variational approximation is less accurate, we found that the effect of non-zero λ could be either a help or a hindrance to the convergence of the CLBP. We also found that λ would sometimes not converge, even with strong damping (implementation of damping is discussed in Appendix C). Associated with regimes of non-convergence, we normally find a divergence of some λ values as discussed in the experimental section. This indicates that failure of the method is most likely related to non-existence (or criticality) of constrained solutions, the issue of solution existence is discussed in Appendix B.1.

3.0.1 Bethe Approximation to the Boltzmann Machine

For the on-and-off diagonal constrained case, the Lagrangian for the Boltzmann machine can be written in the form[6].

$$F_\nu(\boldsymbol{q}, \boldsymbol{\lambda}, \chi) = \sum_{(i,j)\in E} \sum_{x_i,x_j} q_{ij}(x_i, x_j) \log q_{ij}(x_i, x_j) + \sum_i (1 - k_i) \sum_{x_i} q_i(x_i) \log q_i(x_i)$$
$$+ \frac{1}{2} \sum_i \lambda_{i,i} \left[(1 - M_i^2) - \chi_{i,i} \right] + \sum_{(i,j)\in E} \lambda_{(i,j)} \left[\sum_{x_i,x_j} q_{ij}(x_i, x_j) x_i x_j - M_i M_j - \chi_{i,j} \right] , \quad (17)$$

introducing abbreviations for single variable magnetization $M_i := \sum_{x_i} q_i(x_i) x_i$. We recover the diagonal constraint regime when $\lambda_{i,j} = 0 \quad \forall i, j$, and the unconstrained regime when all multipliers are zero. Constraints introduce quadratic functions of q, but terms are neither convex nor concave.

For given λ, a set of message passing equations can be written as a generalization of loopy belief propagation, a simplification of the general case in Appendix B.5 is presented here. At time t the edge-belief is approximated as the solution to the equation

$$q_{ij}^t(x_i, x_j) \propto \mu_{i\to(i,j)}^t(x_i) \mu_{j\to(i,j)}^t(x_j)$$
$$\exp\left[(J_{ij} - \lambda_{ij}) x_i x_j + (h_i + \lambda_{ij} M_{ij,j}^t + \lambda_i M_{ij,i}^t) x_i + (h_j + \lambda_{ij} M_{ij,i}^t + \lambda_j M_{ij,j}^t) x_j \right] ,$$

where we introduce two auxiliary magnetization parameters per edge[7]

$$M_{ij,i}^t = \sum_{x_i x_j} x_i q_{ij}^t(x_i, x_j) . \quad (18)$$

Then we can define messages[8] μ which are determined iteratively as

$$\mu_{(i,j)\to i}^t(x_i) \propto \sum_{x_j} \mu_{j\to(i,j)}^t(x_j)$$
$$\exp\left[(J_{ij} - \lambda_{ij}) x_i x_j + (h_j + \lambda_j M_{ij,j}^t + \lambda_{ij} M_{ij,i}^t) x_j + \lambda_{ij} M_{ij,j}^t x_i \right] ,$$
$$\mu_{i\to(i,j)}^{t+1}(x_i) = \prod_{k\in\partial_i\setminus j} \mu_{(i,k)\to i}^t(x_i) , \quad (19)$$

where ∂_i are the variables interacting with i. Following message updates q^t and M^t must be made consistent, in the examples of this paper (and in general for small λ) this can be achieved simply by iterating Equations (18) and (19). For the experiments messages and beliefs are initialized as constants.

There are several methods by which linear response can be established. One standard approach is susceptibility propagation, which simply involves linearizing the above equations accounting for a small perturbation in some component of ν, this is the approach taken for the examples of this paper; for the general case, expressions are provided in Appendix B.6.

6. To maintain consistency with published research on Ising spin models a factor $1/2$ precedes the diagonal constraint term.
7. Note that at intermediate stages of the message passing, magnetizations for variable i on different beliefs (say $M_{ij,i}$, $M_{ik,i}$) may not agree, but will agree after convergence.
8. These differ by a simple transformation from the messages of the Appendix B.5, so that the limiting case ($\lambda = 0$) agrees with standard presentations for Ising models in the literature.

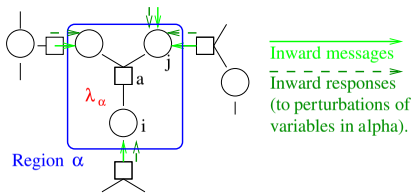


Figure 4: Given C^* and χ, determination of λ breaks into a set of independent problems on outer regions α. This can be understood as a cavity approximation. Assuming the incoming messages, and responses, to be fixed and approximately independent of the Lagrange multipliers to be set ($\boldsymbol{\lambda}_\alpha$), we can independently, on each region, solve the system of equations $\boldsymbol{\Delta}_\alpha = 0$ for $\boldsymbol{\lambda}_\alpha$. A similar assumption often justifies message passing and mean-field iterative heuristics.

3.1 Determination of λ

To determine λ we propose the following scheme. We wish to solve at each α a set of non-linear equations $\{\boldsymbol{\Delta}_\alpha(q_\alpha, q^*_{\alpha,(\cdot)}) = 0\}$, where q_α and $q^*_{\alpha,(\cdot)}$ are functions of all $\boldsymbol{\lambda}$ (through the system of message passing equations). One possibility is to linearize these equations about the current estimate (that is apply Newton's method), but this leads to an impractical $O(N^3)$ procedure, dominating other algorithmic time-scales for moderately sized systems.

Instead, we resort to a locally consistent and parallelizable approximation: For some $\boldsymbol{\lambda}$ we find a minimum defined by messages $\boldsymbol{\mu}$ (Appendix B.5), and the linear response for these messages (Appendix B.6). If $\boldsymbol{\lambda}$ is approximately correct then the messages passing into the region α should be weakly dependent on any changes to $\boldsymbol{\lambda}_\alpha$ in that region. Since we can define $\boldsymbol{\Delta}_\alpha$ in terms of the local parameters $\boldsymbol{\lambda}_\alpha$, and the incoming messages (which we argue are unchanged by an update of $\boldsymbol{\lambda}_\alpha$) the problem for determining $\boldsymbol{\lambda}$ is reduced to solving for $\boldsymbol{\lambda}_\alpha$ independently on every region.

In this way, we arrive at a cavity-approximation style argument common in the motivation of message passing algorithms, see Figure 4. However, it is noteworthy that unlike LBP, incoming messages and responses depend on $\boldsymbol{\lambda}_\alpha$ locally. There is a direct feedback that exists even in the absence of loops.

Unfortunately, $\boldsymbol{\Delta}_\alpha = \mathbf{0}$ remains a (small) system of non-linear equations in $\boldsymbol{\lambda}_\alpha$, that does not allow a closed form solution in general. One exception is when only diagonal constraints are applied (see Yasuda, 2013; Yasuda and Tanaka, 2013), and the cavity argument is applied to single-variable regions. More generally we use Newton's method to solve these equations

$$\left[\frac{\partial \boldsymbol{\Delta}_\alpha(q^*, q^*_{(\cdot)})}{\partial \boldsymbol{\lambda}_\alpha}\right]^{-1} \boldsymbol{\Delta}_\alpha\Big|_{\lambda=\lambda^t} \quad \delta\boldsymbol{\lambda}_\alpha^{t+1} = -\,\boldsymbol{\Delta}_\alpha(q^*, q^*_{(\cdot)})\Big|_{\lambda=\lambda^t} \;, \tag{20}$$

where the dependence of q^* and $q^*_{(\cdot)}$ on λ follows from (??) and (28) with messages fixed.

An alternative expression based on a closed form for the covariance matrix approximation χ is discussed in Appendix C, alongside some other technical details on the determination of λ.

4. Results

In this section, we study the performance of our method on well-understood toy model frameworks. The scale and/or symmetries of these models mean they are exactly solvable, allowing precise statistical estimates to evaluate the method quality.

In Section 4.1 we study a fully connected ferromagnetic model (that is a model with a positive coupling between any pair of variables) with symmetry broken (that is with a non-zero mean value for each variable). This is a simple model for which we can present analytic results and understanding. There is either no mode (for weak coupling), or one dominating mode (for strong couplings). We expect the method to be weakest for intermediate coupling strength since the Bethe approximation becomes exact (with corrections $O(1/N)$) in the limit of strong (a single mode) or weak (no modes) coupling. In Section 4.2 we study a model with frustration (that is couplings have different signs and it is not possible to find a configuration satisfying them all at the same time) and a random distribution of optima and sub-optima. The problem is multi-modal in the limit of strong couplings and the Bethe approximation breaks down. In Section 4.3 we consider a simple model with an expanded discrete alphabet, where randomness is introduced through a random graphical structure. Like the ferromagnetic example, a single mode dominates for strong couplings. We anticipate the Bethe approximation to become exact in the limit of large problems, but for smaller problems the presence of short loops leads to inaccuracy. In the final example of Section 4.4 we consider a well-studied toy model involving both multi-variable interactions and multi-states, both the Bethe approximation and linear response perform poorly on this model. These examples cover a range of scenarios in which our method might be applied. Special cases of the variational approximation we present have previously been applied to lattice models commonly studied in statistical physics, and sparse prior models in Bayesian image modeling (see Raymond and Ricci-Tersenghi, 2013b,a; Yasuda and Tanaka, 2013; Yasuda, 2013).

We present behaviour of the Lagrange multipliers λ, the self-consistency error (13), errors on the pair statistics when using either marginals (14) or linear responses (15), and errors on the marginal estimates

$$\Delta^{(0)}_{(i,y)}(q^*) = q^*(x_i = y) - p(x_i = y) \;.$$

The maximum absolute deviation (MAD) on the marginals is defined as the largest error over all variables, or pairs of variables, depending on error type.

It is interesting to understand how the quality of approximation changes as a function of the goodness of the approximation, to do this we introduce a temperature parameter T in each model, that can sharpen or flatten the distribution.

$$p_T(x) \propto p(x)^{1/T} \;.$$

We can minimize with relative ease both the constrained and unconstrained free energies in the large T regime, and expect approximations to be correct at leading order in $1/T$ (see Raymond and Ricci-Tersenghi, 2013a). In some cases of small T, in which the probability is well described by a single mode, concentrated about some unique value $x_{GS} = \operatorname{argmax} p_T(x)$, the approximations we present are also exact up to $O(T)$ corrections.

We find that in many of the models, minimization of the constrained free energy is slow or impossible for T over some intermediate range, or below some threshold. To extend the range of T for which solutions could be found, an annealing procedure is employed: beginning at large (or small) T and proceeding through a sequence of models slowly changing T, and using the solution to the previous model as the initial condition for the subsequent minimization. Under this procedure, we find that $\lambda(T)$ and the variational parameters q evolve smoothly, but that there appears still to be a limit in the accessible temperature range. In the applications presented we did not find fixed points that appeared discontinuously. Solutions are reached by annealing from low temperature, from high temperature, or are absent.

Our motivation for introducing T is threefold: to study the breakdown of the approximation (absence of solutions), to mitigate for non-convergence, and to increase the speed of convergence. The annealing procedure introduces additional computational costs. We have not made timing comparisons against loopy belief propagation or other competitors, for either the simple or annealed procedure. Instead, we have prioritized an exploration of the nature of solutions that can be discovered, and we have sought very accurate estimates to the parameters describing those solutions. Compromises in the accuracy of constraint satisfaction, annealing rate (or absence of annealing), and damping can all have a significant impact on the speed of the method.

4.1 The Fully Connected Ferromagnet

We begin with a simple but informative case, that of a fully connected ferromagnetic Ising model in an external field.

$$p_T(x) = \frac{1}{Z} \exp\left[\frac{1}{2T}\left(h + \sum_{i=1}^{N} x_i\right)^2\right] . \tag{21}$$

Variables are Ising spins $x_i = \pm 1$. The marginals for this model can be solved up to $O(1/N)$ by the mean-field approximation, and up to $O(1/N^2)$ by the Bethe approximation. Accuracy is a function of temperature; when this is large or small compared to $1/N$, the accuracy is correspondingly high. Under our method the equations to be solved and associated errors can be expressed concisely; this is done in Appendix A.

Three kinds of constraint are considered: diagonal $\{\Delta_{ii} = 0,\ \forall\, i\}$; off-diagonal $\{\Delta_{ij} = 0,\ \forall i \neq j\}$; and on-and-off diagonal applying both sets. Results for both the mean-field and Bethe approximations are presented. Only diagonal constraints can be applied in the mean-field case since the variational parameters are consistent only with zero off-diagonal covariances.

The left panel in Figure 5 shows the behavior of the magnetizations, $\mathbb{E}(x_i)$ for the case $N = 10$ and $h = 1$. The most accurate estimates are obtained with on and off-diagonal constraints applied at the Bethe approximation at high and low temperature. The unconstrained approximations overestimate the magnetization (bias in variables). The addition of constraints corrects this bias: in one constrained regime (off-diagonal constraints only) the suppression is clearly too strong at intermediate T, while in all other cases effective.

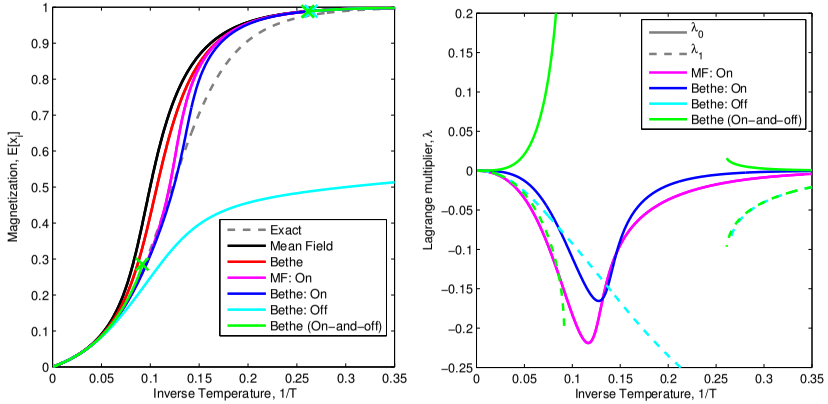


Figure 5: Results for the fully connected model (21) with $N = 10$ and $h = 1$. Results are presented for the mean-field and Bethe approximation schemes, with and without constraints. (left) The magnetization is overestimated in the Bethe and Mean-field approximations, the effect of the constraints is to suppress the magnetization. Certain solutions exist only in the high-temperature or low-temperature regimes, those solutions are demarcated by 'x'. One solution to the Bethe approximation with off-diagonal constraints only suppresses the magnetization too strongly, there is another solution that is present only at low temperature, where the estimate is more reasonable. Where it exists, the Bethe approximation with on and off-diagonal constraints is the most accurate. (right) There are at most two independent values for λ in this model: diagonal λ_0 and off-diagonal λ_1. The Lagrange multipliers deviate most from zero at $T \sim 1/N$ (this is related to a ferromagnetic phase transition). λ diverge rapidly in some solutions demarcating regions where no solutions can be found.

In some constrained approximations there is a smooth evolution of the optimal beliefs with T, which reflects the behavior of the exact marginals. In others, the optimal beliefs for different T are not continuously related.

When off-diagonal constraints are applied in the Bethe approximation, either alongside or without diagonal constraints, we see a discontinuous emergence of the strongly magnetized solution. In the case of only off-diagonal constraints, there is a coexistence of two fixed points for small T. This means that as we vary the parameter T the q^* moves discontinuously from a relatively smooth approximation to one characterized by a single mode. As N increases the domain with coexistence shrinks, and all approximation approach the correct result for large N.

With on and off-diagonal constraints we find a range of T for which no solutions can be found by a continuous evolution of the low or high-temperature solutions. It seems highly likely that no solution exists, and empirically we were not able to find fixed points (for any model) that were not continuously related to either a high or low-temperature solution.

The behaviour of λ (see right panel in Figure 5) and indeed a careful examination of the fixed points indicate why solutions disappear in this simple case. The values of the Lagrange multipliers diverge, and this is related to marginals approaching their boundary values where variational inference breaks down due to the inflexibility of the parameters. By decreasing h or N we decrease the accuracy of the Bethe approximation at intermediate T, this can lead to discontinuity also for the diagonal constrained solution, and reduces the range of temperatures over which the low-temperature solution (the one with a large magnetization) can be found for constrained problems.

4.2 The Wainwright-Jordan Set-up

A common toy model on spin variables $x_i = \pm 1$ is the Wainwright-Jordan set-up where $N = L \times L$ Ising spins are arranged on a square grid (see Opper et al., 2009).

$$p_T(x) = \prod_{(i,j)\in E} \exp(J_{ij} x_i x_j / T) \prod_i \exp(h_i x_i / T) \; .$$

Fields h_i are independent and identically distributed (iid) samples from $[-0.25, 0.25]$ and couplings J_{ij} are sampled i.i.d on $[-1, 1]$. The Bethe approximation fails on these models for smaller T owing to multi-modality of the distribution, but for larger T (where correlations are weaker) the approximation is a significant improvement upon the mean-field approximation.

Diagonal $\{\Delta_{i,i} = 0, \; \forall \; i\}$, or diagonal and off-diagonal $\{\Delta_{i,j} = 0, \forall (i,j) \in E\}$ constraint regimes are studied for the Bethe approximation.

We considered 20 instances for $L = 4$ and $L = 7$, and plot the MAD results in Figures 6 and 7. Shown are the quartiles, for cases where methods did not converge we assigned value +Inf to the MAD so that the quartile values are truncated in cases where the fraction of non-convergent cases exceeded the quartile. For the smaller ($L = 4$) system, the Bethe approximation succeeds for typical cases to full scale ($T = 1$) either without constraints or with only diagonal constraints; the on-and-off diagonal scheme fails in a larger fraction of cases. For the larger ($L = 7$) system, LBP and CLBP fail at full scale: the model which prevails to lowest temperature is CLBP with diagonal constraints; the model failing soonest

is CLBP with on and off-diagonal constraints. The exact range of T for which methods were convergent was sensitive to the annealing procedure, the amount of damping used, and the convergence criteria; strong damping and slow annealing broaden the range for all methods.

Where the algorithm is convergent, there is significant improvement adding diagonal constraints, and more so with on and off-diagonal constraints.We find the improvement in the maximum deviation represents well the changes seen across the entire distribution of marginals in most models, almost all marginals are improved. The quartiles in figures 6 and 7 reflect model to model variations, and demonstrate that the MAD advantage is robust. For $N = 16$, we can compare the median behavior against that reported in Opper, Paquet, and Winther (2009); where the strongest method (tree-EP) also improves the MAD result for marginals by approximately one order of magnitude. Expansion methods offer some further, but modest gains (see Paquet et al., 2009; Opper et al., 2013).

On-diagonal constraints allow an increased range (in T) for convergence. Qualitatively, we offer the following explanation. In the constrained regimes (in the high-temperature regime, the only one for which we demonstrate solutions) the majority of Lagrange multipliers are negative. The effect is to suppress biases and decrease susceptibility (the sensitivity of biases to small changes in the parameters). We assume that reduced susceptibility also correlates with reduced sensitivity to small fluctuations in the messages, which should aid algorithmic stability. With on-and-off diagonal constraints the diagonal and off-diagonal Lagrange multipliers are strongly dependent where they constrain the same variable. The off-diagonal multipliers are typically negative, and suppressing biases; by contrast, the diagonal multipliers are positive and reinforce biases. The typical net effect is to reduce biases and susceptibility, as with the on-diagonal case. However, the strongly correlated nature of the parameters may be the source of convergence problems.

The failure of CLBP is most often due to non-convergence of λ, rather than a failure in the CLBP or CLSP iterative algorithms (at fixed λ). Figure 8 indicates why the iterative update is failing: some Lagrange multipliers are diverging in a strongly correlated manner and it seems likely there is a critical value of T close to the failure point beyond which no solutions exist, as found in Section 4.1.

The LBP implementation follows the same procedure as the constrained cases of Appendix B, with the difference that the innermost do-while loop is always convergent in one iteration, and $\boldsymbol{\lambda} = 0$. Using a double loop procedure we might force convergence to a minimum of the Bethe approximation, but the convergence properties of LBP are still an interesting point of comparison.

We might seek to extend the range of convergence for CLBP by clever modifications. However, it seems that breakdown of convergence is closely related to breakdown of the underlying (Bethe) approximation. Algorithmic innovations would not extend significantly the range of problems for which the constrained approximation is useful in practice; just as the availability of double loop methods has not revolutionized the use of the Bethe approximation: where LBP fails the Bethe approximation is almost always a poor approximation.

For this type of problem a significant improvement to the Bethe approximation is made in moving to a plaquette-based Kikuchi approximations for the case of regular lattices (see Domínguez et al., 2011; Lage-Castellanos et al., 2013). Limited experiments on constrained

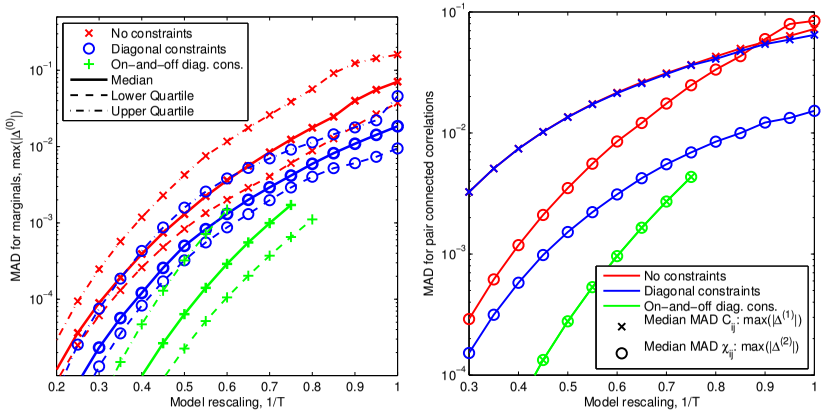


Figure 6: $L = 4$ Wainwright-Jordan set-up: The error on the marginals, and connected correlations (which together provide a sufficient description of pair probabilities) are improved everywhere by adding constraints, as long as the method converges. As discussed, the MAD for C_{ij} is worse than for χ_{ij}, although they are becoming comparable approaching $T = 1$. On the right the median over 20 models is shown, on the left the median and quartiles. The advantage is consistent across all models.

Kikuchi approximations also indicate a modest decrease in error on marginals with the application of constraints.

4.3 Potts Model in an External Field

Next we consider random 3-regular graphs $G = \{V, E\}$ of $N = |V|$ variables, where each variable is allowed 3 states: $x_i \in \{0, 1, 2\}$. The problems are defined by the probability

$$p(x) \propto \prod_{(ij)\in E} \exp\left(J_{ij}\delta_{x_i,x_j}/T\right) \prod_{i\in V} \exp\left(4\delta_{x_i,0}/T\right) \ ,$$

where couplings are i.i.d. random variables $J_{ij} \in \{-1, 1\}$. An example of a corresponding factor graph is shown in Figure 1 (lower panel). Like the fully connected Ising model, typical instances of this model are solved at leading in order in N by the Bethe approximation, with finite size effects strongest at intermediate temperatures. At high temperature, the probability is relatively flat and disperse, whereas at low temperatures there is a single dominating mode concentrated about the value $\mathbf{0} = \text{argmax}_x\, p(x)$.

We consider a diagonal constrained Bethe approximation: a set of 4 non-redundant statistic pairs are constrained per variable.

Figure 9 demonstrates that in graphs of size $N = 40$, the MAD on $p(x_i)$ is significantly improved with the addition of diagonal constraints. Furthermore, we see that as the Bethe approximation becomes accurate (high and low temperature, or large N) it becomes easier to enforce the constraints, as indicated by smaller values for the Lagrange multipliers.

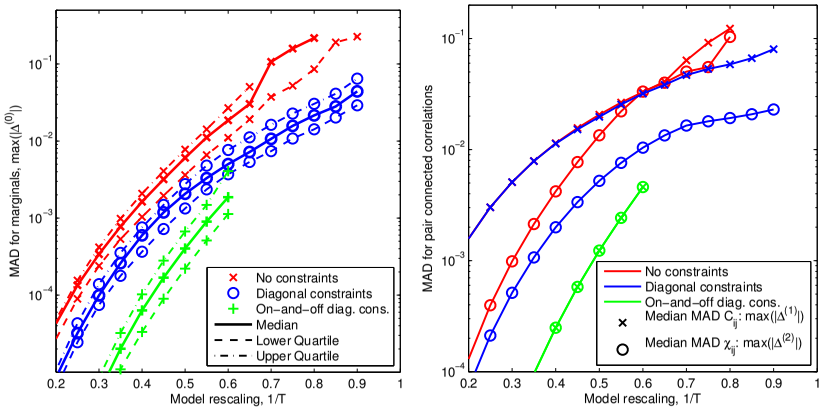


Figure 7: $L = 7$ Wainwright-Jordan set up: Trends are comparable to the smaller system in Figure 6. Where solutions exist, significant gains are made in all models with the addition of constraints. However, all approximations are now failing to reach full scale ($T = 1$). The method which is stable to lowest temperature is the model with diagonal constraints only, while the one with on-and-off constraints is the most accurate.

4.4 The Alarm Network

The alarm net is a pedagogical example of a graphical model that has been studied in the context of loop correction algorithms and is available in libDAI repository (see Mooij et al., 2007; Mooij, 2010). It has 37 variables, each variable takes either 2, 3 or 4 states (a mixture of Ising and Potts spins). Variables have biases, and participate in 2, 3 and 4 point interactions as shown in Figure 1 (upper graph). The model involves a mixture of factor types and variables. The Bethe approximation performs relatively poorly, due to short loops. We can again add a temperature and consider solutions up to the full scale $T = 1$ that defines the model.

Two constraint regimes were applied: in the first pure-diagonal constraints were applied: $\{(i, y), (i, y) : \forall i, y\}$, and in the second all block-diagonal constraints were applied $\{(i, y_1), (i, y_2) : \forall i, y_1, y_2\}$. The ability of both methods to improve local statistics were comparable and modest; for both the local diagonal and off-diagonal statistics as shown in Figure 10. Other schemes such as tree-EP and LCBP show improvements of one or two orders of magnitude on this problem (see Mooij et al., 2007; Opper et al., 2009). However, the Bethe approximation and linear response estimates are known to be poor on this model, so it is not a surprise to see modest gains for our scheme.

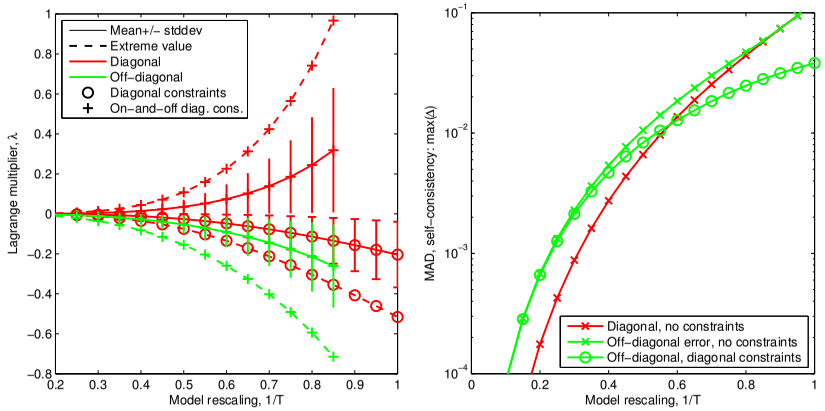


Figure 8: A typical problem in the Wainwright Jordan set up for $N = 16$. (left) Negative λ has the effect of effectively reducing the coupling strength; as with the fully connected ferromagnetic model, a mode of failure for the approximation is to over-estimate the bias in variables, and constraints appear to work by mitigating this effect. Bars indicate the 3 quartiles of the distribution, and the extremal values are also plotted. In the Bethe approximation with only diagonal constraints, most λ including the extremal value are negative. With on and off-diagonal constraints λ with a net effect of reducing interaction strength, values diverge approaching the point of algorithmic failure. (right) Adding diagonal constraints not only removes the diagonal self-consistency error, but also reduces the error in the unconstrained (off-diagonal) statistic. Adding both on and off diagonal constraints removes inconsistency for both types. The self consistency errors (13) grow with decreasing temperature.

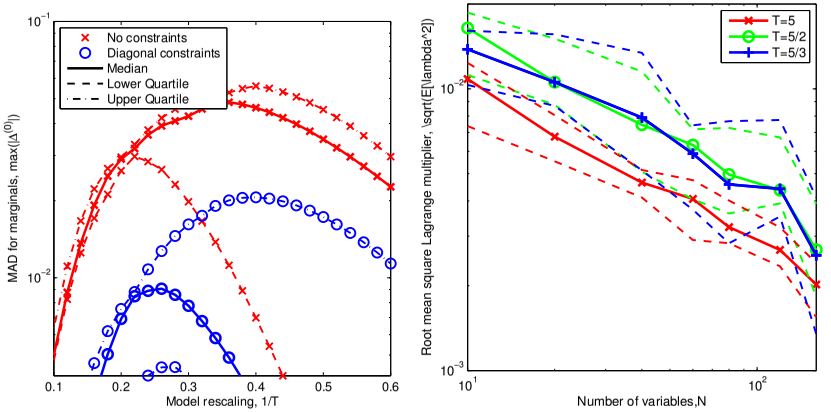


Figure 9: Errors and Lagrange multipliers are shown for the 3-states Potts model on 3-regular random graphs. (left) The diagonal constraint regime yields a significant improvement in maximum absolute deviation of $p(x_i)$ over the raw approximation on graphs with $N = 40$. (right) The Bethe approximation improves as $N \to \infty$ at all temperatures (due to the disappearance of many short loops). The Lagrange multipliers enforcing diagonal consistency typically decrease (in mean squared value) as system size increase. Quartiles are based on at least 20 samples per system size.

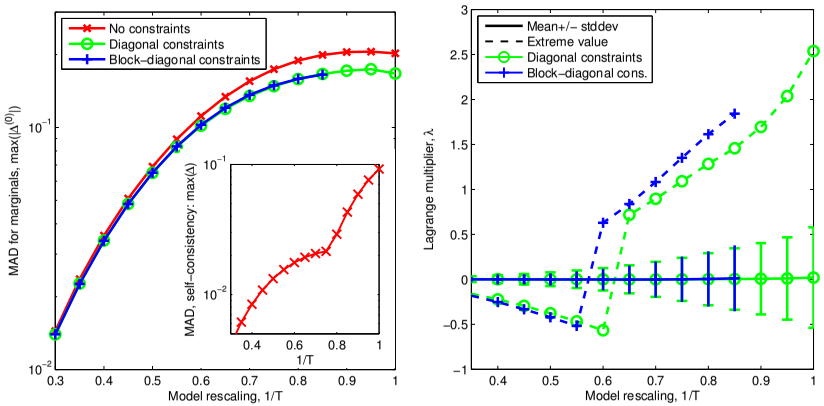


Figure 10: Results for the alarm network. (left) The diagonal and block-diagonal constraint regimes have indistinguishable performance in the MAD for $p(x_i)$, and improve modestly on the unconstrained problem. (inset) At intermediate T there is a qualitative change in the self-consistency error, as some variable subsets become strongly polarized. (right) The values for λ are for the most part small, with extreme values diverging as T approaches 1; a mixture of positive and negative Lagrange multipliers are required to enforce constraints.

5. Discussion

The aim of this paper is to show that for a range of models and variational methods we can have improved accuracy in marginal estimates by inclusion of a set of self-consistent constraints, and to propose this as a general mechanism by which to improve approximations with inconsistency between marginalization (first derivative) and linear response (second derivative) approximations.

We have expanded upon results for rather simple approximations, in easy to understand model frameworks: the marginals are improved in all cases, and in many scenarios by amounts measurable in orders of magnitude. The combination of a weak approximation method with constraints may result in there being no solutions and in marginal cases in a slow convergence of our proposed algorithm. On the other hand, if we are looking for a mechanism by which to leverage a good approximation this approach seems appropriate.

An interesting direction for investigation would be understanding whether partially reducing the violation, rather than eliminating it, might yield comparable results within a more robust algorithmic framework. There are two possibilities: (1) softening the constraint, introducing a penalty term that increases with $|\Delta|$ rather than requiring strict compliance; (2) expanding the range of solution existence by bounding λ. This would presumably maintain a large part of the method advantage by removing some pathologies. Further work could also relate to the choice of constraints, especially for non-Ising models

where the number of statistics available to constrain is large. Two directions are proposed in Appendix B.2.

Some powerful approximation methods are not variational and so the insight we provide cannot be leveraged. In other cases there is no inconsistency to exploit, this is true of adaptive-TAP and some moment matching methods. These have variational free energy frameworks, but by the Gaussian approximations therein used, there is a consistency of pairwise approximations, making redundant the constraints we suggest. This is in line with our thinking that good approximations should not violate these constraints.

We believe our method also sheds light on, or is inclusive of, previous attempts to leverage linear response. Consider the region-based free energy built on the set of regions indicated by Figure 2(d). Every outer region is a hub surrounded by leaves, and if we add constraints over the leaves within each region, a linear expansion in the Lagrange multipliers agrees with the linear expansion obtained in the scheme of Montanari and Rizzo (2005); however, outside of the linearized regime there are important differences.

There are three reasons one might not want to improve a variational approximation by adding self-consistency constraints: 1) The cost of the method which is dominated by a linear response evaluation (a cost that can be as large as $O(N^2)$ using susceptibility propagation), might be prohibitive in some applications. 2) Introducing such constraints may prevent the existence of any solutions (the method may reveal the uncomfortable truth that the approximation used is quite bad). 3) Even where a fixed point exists, it may be slow to reach it using a local iterative scheme for λ such as (20).

6. Conclusion

We have demonstrated that adding covariance constraints, which make the linear response and marginalization estimates consistent, improves the performance of the Bethe approximation for a variety of simple model types. We have argued this is true more generally of variational frameworks, and have provided an algorithmic framework for mean-field and region-based frameworks, generalizing previous results. The regimes of adding all possible constraints (on-and-off diagonal) and adding constraints only over single variable covariances (diagonal) were examined. The former tends to lead to better results, whilst the latter is simpler to implement and can yield solutions across a broader range of models.

The usefulness of this paper is not in the specific algorithm developed, but in the principle of statistical consistency. We hope it might be extended to other variational frameworks in which inconsistencies exist between first and second derivative estimates. We have presented and tested an algorithmic framework; the framework can solve the constrained free energy where solutions exist, but a rather expensive (annealed in T) procedure was exploited to obtain our results. Further work is required to make this reliable and competitive with state of the art marginal estimation.

Acknowledgments

The authors thank Alejandro Lage-Castellanos for comments and Muneki Yasuda for sharing insight on the I-SUSP algorithm.

Appendix A. Exact Expressions for the Fully Connected Ising model

It is straightforward to calculate the exact marginal distributions for (21), for any pair of variables

$$p_T(x_i,x_j) \propto \sum_{n=0}^{N-2} \binom{N-2}{n} \exp\left[\frac{(h+[N-2-2n])^2}{2T} + \frac{1}{T}x_i x_j + \frac{h+[N-2-2n]}{T}(x_i+x_j)\right] .$$

By contrast in the Bethe approximation

$$q_{ij}(x_i,x_j) \propto \exp\left[\frac{1}{T}x_i x_j + \frac{h_{\rightarrow}}{T}(x_i+x_j)\right] , \tag{22}$$

where the log-ratio messages $h_{\rightarrow}$ are defined as the solution to

$$h_{\rightarrow} = h + (N-1)T\text{atanh}\left[\tanh(1/T)\tanh(h_{\rightarrow}/T)\right] ; \qquad h_{\rightarrow} = \frac{1}{2}\log\left(\frac{\mu_{i\rightarrow(ij)}(1)}{\mu_{i\rightarrow(ij)}(-1)}\right) .$$

Since we will place constraints on $\mathbb{V}_q(x_i,x_j)$ a convenient parameterization for the pair probability in our approximation will be

$$q_{ij}(x_i,x_j) = \frac{(1+M_i x_i)(1+M_j x_j) + C_{ij}x_i x_j}{4} . \tag{23}$$

We can restrict attention to symmetric solutions $M_i = M$ and $C_{ij} = C$, similarly at most two distinct Lagrange multiplier values need be considered: $\lambda_{i,i} = \lambda_0$, when diagonal constraints are applied, and $\lambda_{i,j} = \lambda_1$ when off-diagonal constraints are applied. Minimizing the variational free energy (17) with respect to M and C leads to the following pair of equations

$$0 = -[(n-1)M/T + \lambda_0 M] + \text{atanh}(M) + (N-1)\sum_{x_1,x_2}\frac{x_1}{2}q_j(x_2)\log\left(\frac{q_{ij}(x_1,x_2)}{q_i(x_1)}\right) , \tag{24}$$

$$0 = -[1/T - \lambda_1] + \sum_{x_1,x_2}\frac{x_1 x_2}{4}\log q_{ij}(x_1,x_2) . \tag{25}$$

If we solve this pair of equations for $\lambda = 0$, we find an alternative representation of (22). Explicit expressions for the covariance matrix approximation χ are derived in Raymond and Ricci-Tersenghi (2013b) and Raymond and Ricci-Tersenghi (2013a), these can be concisely expressed in the inverse as

$$[\chi^{-1}]_{i,j} = \begin{cases} \frac{1}{1-M^2}\left[1+(n-1)\frac{C^2}{(1-M^2)^2-C^2}\right] - \lambda_0 & \text{if } i=j . \\ -1/T + \sum_{x_1,x_2}\left[\frac{x_1 x_2}{4}\log q_{ij}(x_1,x_2)\right] - \frac{C}{(1-M^2)^2-C^2} & \text{otherwise.} \end{cases}$$

Abbreviating $\chi^{-1}_{i,i} = a$ and $\chi^{-1}_{i,j(j\neq i)} = b$ we can write the components of the covariance matrix approximation

$$\chi_{i,j} = \frac{1}{(a-b)(a+(N-1)b)} \begin{cases} a+(N-2)b & \text{if } i=j . \\ -b & \text{otherwise.} \end{cases}$$

Thus we have up to four parameters $\{M, C, \lambda_0, \lambda_1\}$, depending which constraints are applied. We solve (24) and (25) in combination with either, or both, $(1-M^2 = \chi_{i,i})$ and $(C_{ij} = \chi_{i,j})$. These are non-linear equations in M, C and λ_0; we resort to local search to find solutions; solutions are possible by expansions at small or large T, or for large N. For the solution to be valid, we also check that the Hessian is positive definite, and that $p(x_i,x_j) \geq 0$.

Solutions for the equations are trivial for large or small T, and approach the solutions of the unconstrained approximation. By slowly varying T we can discover the solutions that evolve continuously from these two fixed points. We did not find any solutions appearing discontinuously, nor did we find any evidence for symmetry breaking (that is solutions in which the magnetizations, or correlations, differed by label index).

It is also straightforward to apply the same methodology for the mean-field approximation: there is one saddle-point equation, (24) with $C = 0$; and up to one type of covariance constraint ($\chi_{i,i} = 1-M^2$), where the inverse covariance matrix elements become $[\chi^{-1}]_{i,j} = -1/T$, and $[\chi^{-1}]_{i,i} = 1/(1-M^2) - \lambda_0$.

Appendix B. Algorithm for Finding Optimized Lagrangian Parameters, and the Linear Responses

Our approach to find solutions depends on their existence, this is discussed generically before we consider the impact of constraint selection strategies (and constraint representation) on solution existence and algorithmic stablity. We then show that an exact method for minimizing the Lagrangian exists for some fixed values of the Lagrange parameters λ. The CLBP and CLSP algorithms are then presented in pseudocode. Finally, we discuss heuristics for the calculation of λ.

B.1 Existence

Certain models are known to be solved at leading order in some control parameter by region-based, Bethe or mean-field approximation. Examples are ferromagnetic random graphs or fully connected models in the limit of large number of variables (N), arbitrary models in the weakly interacting limit (also called high temperature T), finite models in the limit of strong biases on the variables (h). In these models, we find, of course, that the deviation between the linear response and marginal estimates are small in the corresponding control parameter. Expanding in this parameter (for example $1/T$, $1/N$) it is straightforward to show the existence of solutions, and quantify the effectiveness of the constraints (see Raymond and Ricci-Tersenghi, 2013a). However, even outside the regime where expansions about the Bethe approximation are appropriate, we find solutions: this includes models where the covariance matrix has divergent terms, due to a mean-field phase transition as described in Raymond and Ricci-Tersenghi (2013b).

However, in experiments, we present it is shown that variation of the temperature (which controls the smoothness of the probability) can lead to models with no solutions, for some constraint sets. In the case of both on and off-diagonal constraints in Ising models defined on some graph $G = \{V, E\}$, with $|V|$ variables and $|E|$ edges, we have $|E|+|V|$ variational parameters but also $|E|+|V|$ constraints—solving a system of non-linear equations where the number of parameters matches the number of constraints seems optimistic; and perhaps it should not be surprising that as the Bethe approximation breaks down solutions fail to

exist; applying all constraints is certainly a marginal case unless some are redundant. If we move to alphabets with more than 2 states per variable the number of pair-statistics for on-and-off diagonal constraints exceeds the number of variables (with no obvious redundancy); certainly such a system can have no solutions. We must think carefully on which constraints to implement, and this is discussed in Appendix B.2.

In the experiments of this paper we have introduced for all models the control parameter T, with all models being well approximated by Bethe (and Mean-field) approximations in the weak coupling limit ($T \to \infty$), and some models also being well approximated in the strong coupling limit ($T \to 0$). To extend the regime in which our algorithms are convergent, an annealing approach was taken—slowly increasing $1/T$ (or T) and following a solution which evolved continuously in the marginals and Lagrange multipliers from the exactly solved regime. In many cases, solutions were found to reach a critical point T^* beyond which they could not be continuously evolved to new solutions. At these points we invariably found no new solutions emerging discontinuously by simply iterating our procedures. By undertaking annealing, it would in principle be possible to miss some solution that emerges discontinuously; the example of Section 4.1 shows this is possible (there is coexistence of two solutions at low temperature for the case of off-diagonal constraints only), but in the other examples of this paper we found no evidence for this. We expect in most practical applications, and for good choices of the constraints, coexistence will be absent.

A common feature in solution discontinuity during annealing is the divergence of some Lagrange multiplier(s); this indicates that the failure was due to inviability of solutions rather than a breakdown of algorithmic dynamics. Unlike the Bethe approximation, which can be forced to converge to a local minimum at any T, replacing LBP by a convex-concave procedure. We speculate that the constrained free energy has no solutions in strongly coupled regimes and that where practical solutions do exist the cavity based algorithms will be sufficient if combined with appropriate damping and/or annealing.

B.2 Selection of Constraints for Best Solutions

In this section we consider reasonable choices for the constraints, assuming a solution existents, and algorithmic convergence is possible.

Since we are applying constraints to discrete models it is clear that the apriori constraints selection should be independent of the labeling convention. Furthermore, we note that we made the rather arbitrary choice in (2) of perturbations in the set of statistics $\{\delta_{x_i,y} : y = 1 \dots Y\}$, Y being the number of states. Whilst this is invariant under relabeling, and spans all possible perturbation directions (and one redundant direction $\sum_y \delta_{x_i,y}$), we might get different covariances according to our choice. It would seem sensible to choose our constraint regime so that any set of statistics that spans the set of perturbations yields the same result. To achieve this we must pick a complete basis $\{\phi\}$, and apply constraints on all-covariance pairs.

The consideration of bases motivates the regimes we have explored: diagonal, off-diagonal, on-and-off diagonal. These schemes are basis-independent. If the condition $C_{(i,\cdot),(j,\cdot)} = \chi_{(i,\cdot),(j,\cdot)}$ is met in one basis for all elements, then a change between two (orthonormal) basis is a rotation, and the identities remain intact.

We found that off-diagonal typically performed worse than on-diagonal, as well as being computationally more challenging. In the final experiment of Section 4.4 we also tried one basis dependent set of constraints, the so-called *pure-diagonal* regime. Despite breaking the paradigm of basis independence, it performed just as well as the *block-diagonal* regime that is basis independent, so this indicates that basis independence is not an essential feature in constraint selection.

Topological distinctions amongst constraints can also be made. In a region-based approximation, we can distinguish the "2-core" from the rest of the graph. The 2-core is the set of variables that are found by recursively removing variables that are involved in at most one "outer region" (generalized interaction). We advise constraining variables on the 2-core only. The reason for this is that the set of variables that are not in the 2-core form a forest—a set of disconnected trees for which the approximation is conditionally exact even without constraints. If we have the core correct, then other constraints will be redundant. It is only in the case of the alarm net (Figure 1, Section 4.4) that the 2-core is distinguished from the full graph and we make this approximation. Other topological distinctions could be applied, distinguishing constraints by the distance between variables might be one possibility (in Figure 2(b) for example there are nearest, and next nearest neighbors within the regions); however, in the Bethe approximation (all experiments we present) only nearest neighbors might be constrained so the example is mute.

Beyond these considerations, we did not attempt to further restrict the set of constraints in a model specific manner, but several options might be worth exploring, in particular: (1) Consider the covariance matrix restricted to a single variable "block", $\chi_{(i,\cdot),(i,\cdot)}$. For a given approximation (for example Bethe) there is a unique orthonormal basis $\{\phi_i\}$ that diagonalizes the covariance matrix. This would provide a natural choice for the statistic basis. The eigenvalue-vector pairs determine which direction is most susceptible to a change in parameters, and so (loosely speaking) most sensitive to approximation errors. We might, therefore, rank constraints by this eigenvalues for inclusion. (2) If we can solve the problem to obtain q_i^*, then we can consider defining only one perturbation statistic per variable as $\phi(x_i) \propto \log(Yq^*(x_i))$. Since we know approximations tend to be overconfident, it would seem a natural choice to constrain in line with the belief, rather than wasting resources on unimportant directions in parameter space.

B.3 Assignment of Constraints to Regions, and Basis selection, for Best Convergence

For our constraint regimes (diagonal, on-diagonal and on-and-off diagonal), the allocation of constraints to specific regions, the choice of statistical basis, the initial condition of the algorithm and the damping, leave the algorithmic fixed points unchanged. Owing to our slow annealing approach (increasing or decreasing T) our results were not very sensitive to the other implementation details. However, these choices do impact convergence significantly.

We find that a non-redundant orthonormal set of statistics leads to faster convergence. Orthonormality ensures that, at leading order, the Lagrange multipliers are independent of one another, which is the criterium that guarantees the success of our method. A redundant statistic is $\phi(x_i) = \sum_{y=0}^{Y-1} \delta_{x_i,y}$, where x_i has Y states, and so only $Y-1$ orthonormal

statistics need be implemented. A nice approach to the selection of an orthonormal basis is discussed in Yasuda (2013), for Ising spins, a unique choice per variable is x_i.

We allocated constraints in a naive manner, assigning each constraint to the first available "outer region". For diagonal constraints an approach where constraints are allocated to single-variable regions was proposed by Yasuda et al., and this allows some simplifications as well as implying a unique allocation (see Yasuda, 2013). Approaches that do not allocate constraints to regions, such as iteration by linear expansion of the covariance matrix have been attempted (29), but those were slower to converge.

As a general rule, we wish to group Lagrange multipliers together as much as possible. In this way, correlations amongst the Lagrange multipliers are accounted for locally. A potentially better approach than our fixed assignment suggestion might be to update all Lagrange within some outer region. Where a constraint could be allocated to more than one region, the results could be averaged, or regions could be selected differently on each cycle to avoid conflicting assignments.

A damping factor that begins as 1 and decreases during the annealing procedure (as required to enable convergence) is applied in the experiments. In simulations without an annealing scheme, a rate $\lesssim 0.5$ could be essential to prevent oscillations that arise from correlated updating of Lagrange multipliers. In particular, off-diagonal values λ_{ij} and diagonal values λ_{ii} can be anti-correlated. If they are not updated on the same region undamped iteration can lead to slow convergence or oscillations.

Initial conditions are chosen as $\lambda = 0$; so that the first iteration is equivalent to an unconstrained approximation. One advantage of the annealing approach (where we first solve the trivial model $T = 0$ or $+\infty$, and then increase or decrease T, initializing by the current solution), is that we do not require the convergence of the unconstrained approximation to find a solution in the constrained regime.

B.4 Convex Concave Procedure

We prove that it is possible to minimize (16) in q by a convex-concave decomposition. In Heskes, Albers, and Kappen (2003) by introducing auxiliary parameters q' over the inner regions, a pair of convex optimization procedures were developed that led to minima of the region based free energy. With the addition of our constraints, we only need to make a minor modification to their argument. Consider an auxiliary free energy $F(q, q')$, where q' are additional parameters in one to one correspondence with the original variational parameters. Heskes et al. outline three requirements of this free energy

1 Convexity of $F(q, q')$ with respect to q.

2 $F(q, q' = q) = F(q)$

3 $F(q, q') \geq F(q)$

that guarantee that the iterative procedure

$$q'_{t+1} = \text{argmin}_q F(q, q'_t) \; , \tag{26}$$

converges to values that are minimizing arguments of $F(q)$.

Since the violation term (13) is a quadratic function of the variational parameters $q(x_\alpha)$, it can be expressed by a constant A_0, a vector of linear coefficients A_1 and quadratic coefficients A_2 that define a symmetric matrix. A_2 can be decomposed as a positive definite and negative semi-definite part (A_2^+ and A_2^-). It is then easy to define a function in vector notation

$$\Delta^+(q, q', q^*_{(\cdot)}) = A_0(q^*_{(\cdot)}) + q^T A_1 + (q')^T A_2 q' + (q - q')^T A_2^+ (q - q') \; . \tag{27}$$

To minimize our region-based free energy with constraints (16) we substitute (27) for (13) and follow the procedure in Heskes, Albers, and Kappen (2003) for the remaining terms. The free energy meets the criteria for convergence under the iterative scheme (26).

The double loop procedure, despite not changing the asymptotic complexity of message passing $O(N)$ is often considered impractical, but, in the framework we are proposing, the largest cost is the evaluation of the linear response, implementing a double loop procedure may be sensible. In practice we have used the variation on loopy belief propagation, rather than the double-loop procedure, to extremize the Lagrangian. Despite the absence of theoretical guarantees, the 3-fold scheme proves to be reliable: solving for variational parameters (by CLBP); solving for the linear response (by CLSP); and update of the Lagrange multipliers.

B.5 Pseudocode for Determining the Fixed Point of q Given λ

We determine q by the constrained loopy belief propagation (CLBP) algorithm, described in Algorithm 1, which is a modified form of the Heskes, Albers, and Kappen (2003) loopy belief propagation algorithm, that uses the convex-concave procedure with guaranteed convergence.

In Algorithm 1 we consider an auxiliary region interaction including both interactions and the Lagrange multiplier terms

$$\psi_\alpha(x_\alpha; \boldsymbol{\lambda}_\alpha, q_\alpha) = \psi_\alpha(x_\alpha) \prod_{[(i,s_1),(j,s_2)] \in \omega_\alpha} \exp\left\{ -\lambda_{(i_1,y_1),(i_2,y_2)} \hat{\phi}_{(i_1,y_1),(i_2,y_2)}(x_\alpha; q_\alpha) \right\} \; ,$$

and

$$\hat{\phi}_{(i_1,y_1),(i_2,y_2)}(x_\alpha; q_\alpha) = \left(\delta_{x_{i_1},y_1} - \mathbb{E}_{q_\alpha}(\delta_{x_{i_1},y_1}) \right) \left(\delta_{x_{i_2},y_2} - \mathbb{E}_{q_\alpha}(\delta_{x_{i_2},y_2}) \right) + \mathbb{E}_{q_\alpha}(\delta_{x_{i_2},y_2}) \mathbb{E}_{q_\alpha}(\delta_{x_{i_1},y_1}) \; .$$

Simplified expressions for the Ising model are presented in the main text. Terms here have an interpretation compatible with Heskes, Albers, and Kappen (2003), with U are the set of *outer regions*, and V are the set of intersection regions (see Figure 2). In the case of a Bethe approximation U are the set of edges, and V are the set of variables.

Algorithm 1 λ compatible Heskes-Albers-Kappen algorithm

while $\neg$ converged **do**

for all $\beta \in V$ **do**

for all $\alpha \in U, \alpha \supset \beta$ **do**

$$q_\alpha(x_\beta) = \sum_{x_{\alpha \setminus \beta}} q_\alpha(x_\alpha)$$

$$\mu_{\alpha \to \beta}(x_\beta) = \frac{q_\alpha(x_\beta)}{\mu_{\beta \to \alpha}(x_\beta)}$$

end for

$$q_\beta^{(num)}(x_\beta) = \prod_{\alpha \supset \beta} \mu_{\alpha \to \beta}(x_\beta)$$

$$q_\beta(x_\beta) = \frac{q_\beta^{(num)}(x_\beta)}{\sum_{x'_\beta} q_\beta^{(num)}(x'_\beta)}$$

for all $\alpha \in U, \alpha \supset \beta$ **do**

$$\mu_{\beta \to \alpha}(x_\beta) = \frac{q_\beta(x_\beta)}{\mu_{\alpha \to \beta}(x_\beta)}$$

while $\neg$ converged (solve self-consistently) **do**

$$q_\alpha^{(num)}(x_\alpha) = \psi_\alpha(x_\alpha; \lambda, q_\alpha) \prod_{\beta \subset \alpha} \mu_{\beta \to \alpha}(x_\beta)$$

$$q_\alpha(x_\alpha) = \frac{q_\alpha^{(num)}(x_\alpha)}{\sum_{x'_\alpha} q_\alpha^{(num)}(x'_\alpha)}$$

end while
end for
end for
end while

B.6 Linear Response about the Fixed Point

By linearizing the CLBP method, we obtain a constrained loopy susceptibility propagation (CLSP) method. Qualitatively similar is the I-SUSP algorithm of Yasuda and Tanaka (2013), which applies only to the case of diagonal constraints and the Bethe approximation.

The implementation we use follows Algorithm 1, but with the following replacements for each equation (in order):

$$\begin{aligned}
\delta\hat{q}_\alpha(x_\beta) &= \sum_{x_{\alpha \setminus \beta}} \frac{q_\alpha(x_\alpha)}{q_\beta(x_\beta)} \delta\hat{q}_\alpha(x_\alpha) \; ; \\
\delta\hat{\mu}_{\alpha \to \beta}(x_\beta) &= \delta\hat{q}_\alpha(x_\beta) - \delta\hat{\mu}_{\beta \to \alpha}(x_\beta) \; ; \\
\delta\hat{q}_\beta^{(num)}(x_\beta) &= \sum_{\alpha \supset \beta} \delta\hat{\mu}_{\alpha \to \beta}(x_\beta) \; ; \\
\delta\hat{q}_\beta(x_\beta) &= \delta\hat{q}_\beta^{(num)}(x_\beta) - \sum_{x_\beta} q_\beta(x_\beta) \delta\hat{q}_\beta^{(num)}(x_\beta) \; ; \\
\delta\hat{\mu}_{\beta \to \alpha}(x_\beta) &= \delta\hat{q}_\beta(x_\beta) - \delta\hat{\mu}_{\alpha \to \beta}(x_\beta) \; ; \\
\delta\hat{q}_\alpha^{(num)}(x_\alpha) &= \frac{\delta_{x_i, y} \mathbb{I}(i \in \alpha)}{k_i} + \sum_{\beta \subset \alpha} \delta\hat{\mu}_{\beta \to \alpha}(x_\beta) \\
&+ \sum_{x'_\alpha} q_\alpha(x'_\alpha) \delta\hat{q}_\alpha(x'_\alpha) \frac{\partial \log \psi(x_\alpha; \boldsymbol{\lambda}_\alpha, q_\alpha)}{\partial q_\alpha(x'_\alpha)} \; ; \\
\delta\hat{q}_\alpha(x_\alpha) &= \delta\hat{q}_\alpha^{(num)}(x_\alpha) - \sum_{x'_\alpha} q_\alpha(x'_\alpha) \delta\hat{q}_\alpha^{(num)}(x'_\alpha) \; ;
\end{aligned}$$

where $\mathbb{I}(i \in \alpha)$ is an indicator function evaluating to one if i is contained in α, and k_i is the number of outer regions containing i. From the converged quantities we can identify

$$q^*_{\alpha,(i,y)}(x_\alpha) = q_\alpha(x_\alpha) \delta\hat{q}_\alpha(x_\alpha) \; . \tag{28}$$

Since the linear response is a linearized method, the innermost do-while loop of Algorithm 1 could be replaced by a method for solving linear equations. Both have been tried with similar outcomes.

Appendix C. Solving for λ, Explicit Expressions and Simplification

The diagonal update scheme of I-SUSP relies on a similar reasoning as that undertaken in Yasuda and Tanaka (2013) and Yasuda (2013). The difference between the two approaches is that in the works by Yasuda et al. only diagonal constraints (restricted to single variable consistency relations) are considered. As such it is possible to choose in general a single variable cavity i, rather than a region cavity α (see Figure 4). To consider statistics not restricted to single variables our scheme is required. The I-SUSP method has an advantage over our method in that there is a simplification of the expressions on a single variable region that allows a closed form for updating the Lagrange multipliers. One advantage of our scheme is that we may fix several Lagrange multipliers on a region simultaneously, thus accounting for correlations in their values, and perhaps reducing the number of updates to convergence.

In practice, damping within the 3-cycle of Figure 3 can be necessary for convergence. In practice, we replace $\lambda^{t+1} = d\lambda^t + (1-d)\lambda^*$, where λ^* is the cavity approximation. As $1/T$ was increased (or decreased) at a constant rate we increased d whenever the solution failed to converge, and this modification extended the region of convergence.

An exact expression for the correlation matrix χ as a function of the variational parameters was derived in Raymond and Ricci-Tersenghi (2013b) and Raymond and Ricci-Tersenghi (2013a), the special case for homogeneous graphs is made explicit (Appendix A). The inverse covariance matrix χ^{-1} is, in the basis where variational parameters are beliefs q, a linear function of λ. Updating a single λ can be achieved by the Shermann-Morrison formula, which in its linearized form becomes

$$\chi_{i,i}\,\delta\lambda_{i,j}\,\chi_{j,j} = (C^*_{i,j} - \chi_{i,j})\ . \tag{29}$$

It is also possible to write a similar expression blockwise (over $\boldsymbol{\lambda_\alpha}$). This scheme and variations were considered, but were found to require more iterations or stronger damping (for convergence) than the linearized cavity formula (20).

In our approach we associate each constraint to a unique region, but some statistics such as $\mathbb{V}_p(x_i, x_i)$ may be approximated by different α for the same variational approach (for example by q_{ij} or q_{ik} if i participates in two interactions). Though these estimates agree finally at the solution point, they disagree at intermediate stages of CLBP. Since updates for λ may be correlated (consider Figure 8 where diagonal and off-diagonal are strongly anti-correlated), the grouping of λ may be very important. Updating non-disjoint sets and averaging over results may be more stable than our scheme where closely correlated λ (for example $\lambda_{i,i}$ and $\lambda_{i,j}$) may be updated on different regions.

References

C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1):5–43, 2003.

M. Chertkov and V.Y. Chernyak. Loop calculus in statistical physics and information science. *Phys. Rev. E*, 73:065102, 2006.

P. Dagum and M. Luby. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artif. Intell.*, 60(1):141–153, March 1993.

R. Dickman and G. Stell. Self-consistent Ornstein-Zernike approximation for lattice gases. *Physical review letters*, 77(6):996, 1996.

E. Domínguez, A. Lage-Castellanos, R. Mulet, F. Ricci-Tersenghi, and T. Rizzo. Characterizing and improving generalized belief propagation algorithms on the 2d Edwards–Anderson model. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(12): P12007, 2011.

R.J. Giordano, T. Broderick, and M.I. Jordan. Linear response methods for accurate covariance estimates from mean field variational Bayes. In *Advances in Neural Information Processing Systems 28*, pages 1441–1449. Curran Associates, Inc., Red Hook, NY, 2015.

T. Heskes, K. Albers, and B. Kappen. Approximate inference and constrained optimization. In *Uncertainty in Artificial Intelligence: Proceedings of the Nineteenth Conference (UAI-2003)*, pages 313–320, San Francisco, CA, 2003. Morgan Kaufmann Publishers.

J. S. Høye and G. Stell. Thermodynamics of the MSA for simple fluids. *J. Chem. Phys.*, 67:439–445, July 1977. doi: 10.1063/1.434887.

H. Huang and Y. Kabashima. Adaptive Thouless-Anderson-Palmer approach to inverse Ising problems with quenched random fields. *Phys. Rev. E*, 87:062129, Jun 2013.

H.J. Kappen and F.B. Rodriguez. Efficient learning in Boltzmann machines using linear response theory. *Neural Comput.*, 20:1137, 1998.

E. Kierlik, M.L. Rosinberg, and G. Tarjus. A self-consistent Ornstein–Zernike approximation for the random field Ising model. *Journal of Statistical Physics*, 94(5-6):805–836, 1999.

A. Lage-Castellanos, R. Mulet, F. Ricci-Tersenghi, and T. Rizzo. Replica cluster variational method: the replica symmetric solution for the 2d random bond Ising model. *Journal of Physics A: Mathematical and Theoretical*, 46(13):135001, 2013.

P.M. Long and R.A. Servedio. Restricted Boltzmann machines are hard to approximately evaluate or simulate. In *Proceedings of the 27th International Conference on Machine Learning*, pages 703–710. Omnipress, 2010.

D.J. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, Cambridge, UK, 2004.

M. Mezard and A. Montanari. *Information, Physics, and Computation*. Oxford University Press, Inc., New York, NY, USA, 2009.

M. Mezard and T. Mora. Constraint satisfaction problems and neural networks: a statistical physics perspective. arXiv:0803.3061, 2008.

J. Miskin and D. J. C. MacKay. *Ensemble Learning for Blind Image Separation and Deconvolution*, pages 123–141. Springer London, London, 2000.

A. Montanari and T. Rizzo. How to compute loop corrections to the Bethe approximation. *J. Stat. Mech.*, page P10011, 2005.

A. Montanari, F. Ricci-Tersenghi, and G. Semerjian. Solving constraint satisfaction problems through belief propagation-guided decimation. In *45th Annual Allerton Conference on Communication, Control and Computing*, Red Hook, NY, USA, 2007. Curran Associates, Inc.

J.M. Mooij. libDAI: A free and open source c++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, August 2010.

J.M. Mooij, B. Wemmenhove, B. Kappen, and T. Rizzo. Loop corrected belief propagation. In *International Conference on Artificial Intelligence and Statistics*, pages 331–338, 2007.

M. Opper and O. Winther. Tractable approximations for probabilistic models: The adaptive Thouless-Anderson-Palmer mean field approach. *Phys. Rev. Lett.*, 86:3695–3699, 2001.

M. Opper and O. Winther. Variational linear response. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2003.

M. Opper and O. Winther. Expectation consistent free energies for approximate inference. In *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA, 2005.

M. Opper, U. Paquet, and O. Winther. Improving on expectation propagation. In *Advances in Neural Information Processing Systems 21*. MIT Press, Cambridge, MA, 2009.

M. Opper, U. Paquet, and O. Winther. Perturbative corrections for approximate inference in Gaussian latent variable models. *Journal of Machine Learning Research*, 14:2857–2898, 2013.

U. Paquet, O. Winther, and M. Opper. Perturbation corrections in approximate inference: Mixture model applications. *Journal of Machine Learning Research*, 10:1263–1304, 2009.

G. Parisi. *Statistical Field Theory*. Addison-Wesley, Boston, MA, USA, 1987.

A. Pelizzola. Cluster variation method in statistical physics and probabilistic graphical models. *J. Phys. A*, 38(33):R309, 2005.

J. Raymond and F. Ricci-Tersenghi. Correcting beliefs in the mean-field and Bethe approximations using linear response. In *Communications Workshops (ICC), 2013 IEEE International Conference on*, pages 1429–1433, 2013a.

J. Raymond and F. Ricci-Tersenghi. Mean-field method with correlations determined by linear response. *Phys. Rev. E*, 87:052111, 2013b.

T. Richardson and R. Urbanke. *Modern Coding Theory*. Cambridge University Press, Cambridge, UK, 2008.

M.J. Wainwright and M.I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, 2008.

M. Welling and Y.W. Teh. Linear response algorithms for approximate inference in graphical models. *Neural Comput.*, 16:197–221, 2004.

O. Winther and M. Opper. Expectation consistent approximate inference. *Journal of Machine Learning Research*, 6:2177–2204, 2005.

M. Yasuda. An generalization of improved susceptibility propagation. In *Inference, Computation, and Spin Glasses (ICSG2013)*, 2013.

M. Yasuda and K. Tanaka. Susceptibility propagation by using diagonal consistency. *Phys. Rev. E*, 87:012134, Jan 2013.

J.S. Yedidia, W.T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *Information Theory, IEEE Transactions on*, 51(7):2282–2312, 2005.

A. L. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: Convergent alternatives to belief propagation. *Neural Computation*, 14:2002, 2002.

Journal of Machine Learning Research 18 (2017) 1-37 Submitted 5/16; Revised 1/17; Published 1/17

Distributed Sequence Memory of Multidimensional Inputs in Recurrent Networks

Adam S. Charles ADAMSC@PRINCETON.EDU
Princeton Neuroscience Institute
Princeton University
Princeton, NJ 08544, USA

Dong Yin DONGYIN@BERKELEY.EDU
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, CA 94720-1776, USA

Christopher J. Rozell CROZELL@GATECH.EDU
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0250, USA

Editor: Samy Bengio

Abstract

Recurrent neural networks (RNNs) have drawn interest from machine learning researchers because of their effectiveness at preserving past inputs for time-varying data processing tasks. To understand the success and limitations of RNNs, it is critical that we advance our analysis of their fundamental memory properties. We focus on echo state networks (ESNs), which are RNNs with simple memoryless nodes and random connectivity. In most existing analyses, the short-term memory (STM) capacity results conclude that the ESN network size must scale linearly with the input size for unstructured inputs. The main contribution of this paper is to provide general results characterizing the STM capacity for linear ESNs with multidimensional input streams when the inputs have common low-dimensional structure: sparsity in a basis or significant statistical dependence between inputs. In both cases, we show that the number of nodes in the network must scale linearly with the information rate and poly-logarithmically with the input dimension. The analysis relies on advanced applications of random matrix theory and results in explicit non-asymptotic bounds on the recovery error. Taken together, this analysis provides a significant step forward in our understanding of the STM properties in RNNs.

Keywords: short-term memory, recurrent neural networks, sparse signal recovery, low-rank recovery, restricted isometry property

©2017 Adam Charles, Dong Yin, and Christopher Rozell.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-270.html.

1. Introduction

Recurrent neural networks (RNNs) have drawn interest from researchers because of their effectiveness at processing sequences of data (Jaeger, 2001; Lukoševičius, 2012; Hinaut et al., 2014). While deep networks have shown remarkable performance improvements at task such as image classification, RNNs have recently been successfully employed as layers in conventional deep neural networks to expand these tools into tasks with time-varying data (Sukhbaatar et al.; Gregor et al., 2015; Graves et al., 2013; Bashivan et al., 2016). This inclusion is becoming increasingly important as neural networks are being applied to a growing variety of inherently temporal high-dimensional data, such as video (Donahue et al., 2015), audio (Graves et al., 2013), EEG data (Bashivan et al., 2016), two-photon calcium imaging (Apthorpe et al., 2016). Despite the growing use of both deep and recurrent networks, theory characterizing the properties of such networks remain relatively unexplored. For deep neural networks, much of the computational power is often attributed to flexibility in learned representations (Mallat, 2016; Vardan et al., 2016; Patel et al., 2015). The power of RNNs, however, is tied to the ability of the recurrent network dynamics to act as a distributed memory substrate, preserving information about past inputs to leverage temporal dependencies for data processing tasks such as classification and prediction. To understand the success and limitations of RNNs, it is critical that we advance our analysis of the fundamental memory properties of these network structures.

There are many types of recurrent network structures that have been employed in machine learning applications, each with varying complexity in the network elements and the training procedures. In this paper we will focus on RNN structures known as echo state networks (ESNs). These networks have discrete time continuous-valued nodes $\boldsymbol{x}[n] \in \mathbb{R}^M$ that evolve at time n in response to the inputs $\boldsymbol{s}[n] \in \mathbb{R}^L$ according to the dynamics:

$$\boldsymbol{x}[n+1] = f(\boldsymbol{W}\boldsymbol{x}[n] + \boldsymbol{Z}\boldsymbol{s}[n] + \boldsymbol{\epsilon}[n]), \tag{1}$$

where $\boldsymbol{W} \in \mathbb{R}^{M\times M}$ is the connectivity matrix defining the recurrent dynamics, $\boldsymbol{Z} \in \mathbb{R}^{M\times L}$ is the weight vector describing how the input drives the network, $f(\cdot) : \mathbb{R}^M \to \mathbb{R}^M$ is an element-wise nonlinearity evaluated at each node and $\boldsymbol{\epsilon}[n] \in \mathbb{R}^M$ represents the error due to potential system imperfections (Jaeger, 2001; Wilson and Cowan, 1972; Amari, 1972; Sompolinsky et al., 1988; Maass et al., 2002). In an ESN, the connectivity matrix $\boldsymbol{W}$ is random and untrained, while the simple individual nodes have a single state variable with no memory. This is in contrast to approaches such as long short-term memory units (Sak et al., 2014; Lipton et al., 2016; Kalchbrenner et al., 2016) which have individual nodes with complex memory properties. As with many other recent papers, we will also focus on linear networks where $f(\cdot)$ is the identity function (Jaeger, 2001; Jaeger and Haas, 2004; White et al., 2004; Ganguli et al., 2008; Ganguli and Sompolinsky, 2010; Charles et al., 2014; Wallace et al., 2013).

The memory capacity of these networks has been studied in both the machine learning and computational neuroscience literature. In the approach of interest, the short-term memory (STM) of a network is characterized by quantifying the relationship between the transient network activity and the recent history of the exogenous input stream driving the network (Jaeger and Haas, 2004; Maass et al., 2002; Ganguli and Sompolinsky, 2010; Wallace et al., 2013; Verstraeten et al., 2007; White et al., 2004; Lukoševičius and Jaeger, 2009; Buonomano and Maass, 2009; Charles et al., 2014). Note that this is in contrast to alternative approaches that characterize long-term memory in RNNs through quantifying the number of distinct network attractors that can be used to stably remember

input patterns with the asymptotic network state. In the vast majority of the existing theoretical analysis of STM, the results conclude that networks with M nodes can only recover inputs of length $N \leq M$ (White et al., 2004; Wallace et al., 2013) when the inputs are unstructured.

However, in any machine-learning problem of interest, the input statistical structure is precisely what we intend to exploit to accomplish meaningful tasks. For one example, many signals are well-known to admit a sparse representation in a transform dictionary (Elad et al., 2010; Davies and Daudet, 2006). In fact, some classes of deep neural networks have been designed to induce sparsity at higher layers that may serve as inputs into the recurrent layers (LeCun et al., 2010; Kavukcuoglu et al., 2010). For another example, a collection of time-varying input streams (e.g., pixels or image features in a video stream) are often heavily correlated. In the specific case of single input streams ($L = 1$) with inputs that are K-sparse in a basis, recent work (Charles et al., 2014) has shown that the STM capacity can scale as favorably as $M = \mathcal{O}(K \log^{\gamma}(N)) \leq N$, where $\gamma \geq 1$ is a constant. In other words, the memory capacity can scale linearly with the information rate in the signal and only logarithmically with the signal dimension, resulting in the potential for recovery of inputs of length $N \gg M$. Unfortunately, existing analyses (Jaeger, 2001; White et al., 2004; Ganguli and Sompolinsky, 2010; Charles et al., 2014) are generally specific to the restricted case of single time-series inputs ($L = 1$) or unstructured inputs (Verstraeten et al., 2010).

Conventional wisdom is that structured inputs should lead to much higher STM capacity, though this has never been addressed with strong analysis in the general case of ESNs with multidimensional input streams. The main contribution of this paper is to provide general results characterizing the STM capacity for linear randomly connected ESNs with multidimensional input streams when the inputs are either sparse in a basis or have significant statistical dependence (with no sparsity assumption). In both cases, we show that the number of nodes in the network must scale linearly with the information rate and poly-logarithmically with the total input dimension. The analysis relies on advanced applications of random matrix theory, and results in non-asymptotic analysis of explicit bounds on the recovery error. Taken together, this analysis provides a significant step forward in our understanding of the STM properties in RNNs. While this paper is primarily focused on network structures in the context of RNNs in machine learning, these results also provide foundation for the theoretical understanding of recurrent network structures in biological neural networks, as well as the memory properties in other network structures with similar dynamics (e.g., opinion dynamics in social networks).

2. Background and Related Work

2.1 Short Term Memory in Recurrent Networks

Many approaches have been used to analyze the STM of randomly connected networks, including nonlinear networks (Sompolinsky et al., 1988; Massar and Massar, 2013; Faugeras et al., 2009; Rajan et al., 2010; Galtier and Wainrib, 2016; Wainrib, 2015) and linear networks (Jaeger, 2001; Jaeger and Haas, 2004; White et al., 2004; Ganguli et al., 2008; Ganguli and Sompolinsky, 2010; Charles et al., 2014; Wallace et al., 2013) with both discrete-time and continuous-time dynamics. These methods can be broadly be classified as either correlation-based methods (White et al., 2004; Ganguli et al., 2008) or uniqueness methods (Jaeger, 2001; Maass et al., 2002; Jaeger and Haas, 2004; Charles et al., 2014; Legenstein and Maass, 2007; Büsing et al., 2010). Correlation methods focus on quantifying the correlation between the network state and recent network inputs. In these studies, the STM is defined as the time of the oldest input where the correlation between the network

state and that input remains above a given threshold (White et al., 2004; Ganguli et al., 2008). These methods have mostly been applied to discrete-time systems, and have resulted in bounds on the STM that scale linearly with the number of nodes (i.e. $M > N$).

In contrast, uniqueness methods instead aim to show that different network states correspond to unique input sequences (i.e. the network dynamics are bijective).[1] For uniqueness methods, the STM is defined as the longest input length where this input-network state bijection still holds. These methods have been used under the term *separability property* for continuous-time liquid state machines (Maass et al., 2002; Vapnik and Chervonenkis, 1971; Legenstein and Maass, 2007; Wallace et al., 2013; Büsing et al., 2010) and under the term *echo-state property* for discrete-time ESNs (Jaeger, 2001; Yildiz et al., 2012; Buehner and Young, 2006; Manjunath and Jaeger, 2013). The echo-state property is the method most related to the approach we take here, and essentially derives the maximum length of the input signal such that the resulting network states remain unique. While this property guarantees a bijection between inputs and network states, it does not take into account input signal structure, does not capture the robustness of the mapping, and does not provide guarantees for stably recovering the input from the network activity.

2.2 Compressed Sensing

The compressed sensing literature and its recent extensions include many tools for studying the effects of random matrices applied to low-dimensional signals. Specifically, in the basic compressed sensing problem we desire to recover the signal $\boldsymbol{s} \in \mathbb{R}^N$ from M measurements[2] generated from a random linear measurement operator,

$$\boldsymbol{x} = \mathcal{A}(\boldsymbol{s}) + \boldsymbol{\epsilon}, \tag{2}$$

where $\boldsymbol{\epsilon} \in \mathbb{R}^M$ represents the potential measurement errors. Typically, $\boldsymbol{s}$ is assumed to have low-dimensional structure and recovery is performed via a convex optimization program. The most common example is a sparsity model where $\boldsymbol{s}$ can be represented as

$$\boldsymbol{s} = \boldsymbol{\Psi}\boldsymbol{a},$$

where $\boldsymbol{\Psi} \in \mathbb{R}^{N \times N}$ is a transform matrix and $\boldsymbol{a} \in \mathbb{R}^N$ is the sparse coefficient representation of $\boldsymbol{s}$ with $K \ll N$ of its entries non-zero. Under this sparsity assumption, the coefficient representation is recoverable if the linear operator $\mathcal{A}$ satisfies the restricted isometry property (RIP) that guarantees uniqueness of the compressed measurements. Specifically, we say that $\mathcal{A}$ satisfies the RIP($2K$,δ) if for every $2K$-sparse signal $\boldsymbol{s}$, the following condition is satisfied:

$$C(1-\delta) \leq ||\mathcal{A}\boldsymbol{s}||_2^2 / \, ||\boldsymbol{s}||_2^2 \leq C(1+\delta),$$

where $0 < \delta < 1$ and $C > 0$ is a positive constant. When $\mathcal{A}$ satisfies the RIP($2K$,δ) the sparse coefficients $\boldsymbol{a}$ can be recovered by solving an ℓ_1-norm based optimization function

$$\boldsymbol{a} = \arg\min_{\boldsymbol{a}} ||\boldsymbol{a}||_1 \quad \text{such that} \quad ||\boldsymbol{x} - \mathcal{A}(\boldsymbol{\Psi}\boldsymbol{a})||_2 \leq ||\boldsymbol{\epsilon}||_2, \tag{3}$$

1. We note that uniqueness-based methods imply recovery-based methods, modulo a recovery algorithm, as often recovery guarantees are based on some semblance of a bijection.
2. In the case of RNNs, the network node values act as the measurements of our system, prompting the use of M as the number of measurements in this section

up to a reconstruction error given by

$$||\boldsymbol{s}-\widehat{\boldsymbol{s}}||_2 \leq \alpha\,||\boldsymbol{\epsilon}||_2 + \beta\frac{\left|\left|\boldsymbol{\Psi}^T\left(\boldsymbol{s}-\boldsymbol{s}_K\right)\right|\right|_1}{\sqrt{K}}, \tag{4}$$

where α and β are constants (Candès et al., 2006). The first term of this recovery error bound depends on the norm of the measurement error $\boldsymbol{\epsilon}$, while the second term depends on the ℓ_1 difference between the true signal and the best K-sparse approximation of the true vector ($\boldsymbol{s}_K$). This term essentially measures how closely the signal matches the sparsity model. The ℓ_1 optimization program in (3) required for recovery can be solved by many efficient algorithms, including neurally plausible architectures (Rozell et al., 2010; Balavoine et al., 2012; Shapero et al., 2014; Charles et al., 2012).

When the data of interest is a matrix $\boldsymbol{S} \in \mathbb{R}^{L\times N}$, other low-dimensional models have also been explored. For example, as an alternative to a sparsity assumption, the successful low-rank model assumes that there are correlations between rows and columns such that $\boldsymbol{S}$ has rank $R < \min\{L,N\}$. We can then write the decomposition of the matrix as

$$\boldsymbol{S} = \boldsymbol{Q}\boldsymbol{V}^*,$$

where $\boldsymbol{Q} \in \mathbb{R}^{L\times R}$ and $\boldsymbol{V}^* \in \mathbb{R}^{R\times N}$. There is a rich and growing literature dedicated to establishing guarantees for recovering low-rank matrices from incomplete measurements. Due to the difficulty of establishing a general matrix-RIP property for observations of a matrix (Recht et al., 2010), the guarantees in this literature more commonly use the optimality conditions for specific optimization procedures to show that the resulting solution has bounded error with high probability. The most common optimization program used for low-rank matrix recovery is the nuclear norm minimization,

$$\boldsymbol{S} = \arg\min_{\boldsymbol{S}} ||\boldsymbol{S}||_* \quad \text{such that} \quad ||\boldsymbol{x}-\mathcal{A}\left(\boldsymbol{S}\right)||_2 \leq ||\boldsymbol{\epsilon}||_2\,, \tag{5}$$

where the nuclear norm $||\boldsymbol{S}||_*$ is defined as the sum of the singular values of $\boldsymbol{S}$ (Candès and Tao, 2010; Candès and Plan, 2010; Recht et al., 2010; Chen and Suter, 2004; Fazel, 2002; Singer and Cucuringu, 2010; Toh and Yun, 2010; Liu and Vandenberghe, 2009; Jaggi et al., 2010). This optimization procedure is similar to the ℓ_1-regularized optimization of Equation (3), however the nuclear-norm induces sparsity in the singular values rather than the matrix entries directly.

The solution to Equation (5) can be shown to satisfy performance guarantees via the dual-certificate approach (Candès and Plan, 2010; Ahmed and Romberg, 2015). This technique is a proof by construction and shows that a dual certificate (i.e., a vector whose projections into and out of the space spanned by the singular vectors of $\boldsymbol{S}$ are bounded) exists. Showing that such a certificate exists demonstrates that Equation (5) converges to a valid solution and is key to deriving accuracy bounds (Candès and Plan, 2010; Ahmed and Romberg, 2015). Specifically, if the dual certificate exists, then the solution to Equation (5) satisfies the recovery bound

$$\left|\left|\widehat{\boldsymbol{S}}-\boldsymbol{S}\right|\right|_F \leq \left(4\sqrt{\min(N,L)\frac{2NL+M}{M}}+2\right)\epsilon, \tag{6}$$

where the Forbenius norm $\|\cdot\|_F^2$ is defined as the sum of the squares of all the matrix entries. This bound demonstrates that perfect recovery is achievable in the case where there is no error ($\epsilon = 0$). We note that alternate optimization programs with similar guarantees have been proposed in the literature for inferring low-rank matrices (i.e. Ahmed and Romberg, 2015), but we will focus on nuclear norm optimization approaches due to the extensive literature on nuclear-norm solvers and the connections to sparse vector inference.

2.3 STM Capacity via the RIP

The ideas and tools from the compressed sensing literature have recently been used to show that *a-priori* knowledge of the input sparsity can lead to improvements recovery-based STM capacity results for ESNs. For a single input stream under a sparsity assumption, Ganguli and Sompolinsky (2010) analyzed an annealed version of the network dynamics to show that the network memory capacity can be larger than the network size. Building on this observation, Charles et al. (2014) provided an analysis of the exact network dynamics in an ESN (for the single input case of $L = 1$), yielding precise bounds on a network's STM capacity captured in the following theorem:

Theorem 1 (Theorem 4.1.1, Charles et al., 2014) *Suppose $N \geq M$, $N \geq K$, $N \geq O(1)$,*[3] *and $L = 1$. Let $\boldsymbol{U}$ be any unitary matrix of eigenvectors (containing complex conjugate pairs) of the connectivity matrix $\boldsymbol{W}$ and for M an even integer, denote the eigenvalues of $\boldsymbol{W}$ by $\{e^{jw_m}\}_{m=1}^{M}$. Let the first $M/2$ eigenvalues ($\{e^{jw_m}\}_{m=1}^{M/2}$) be chosen uniformly at random on the complex unit circle (i.e., $\{w_m\}_{m=1}^{M/2}$ is uniformly distributed over $[0,2\pi)$) and the other $M/2$ eigenvalues as the complex conjugates of these values. Furthermore, let the entries of the input weights $\boldsymbol{z}$ be i.i.d. zero-mean Gaussian random variables with variance $\frac{1}{M}$. Given RIP conditioning δ and failure probability $N^{-\log^4 N} \leq \eta \leq \frac{1}{e}$, if*

$$M \geq C\frac{K}{\delta^2}\mu^2\left(\boldsymbol{\Psi}\right)\log^5\left(N\right)\log(\eta^{-1}),$$

then for a universal constant C, with probability $1-\eta$ the mapping of length-N input sequences into M network state variables satisfies the RIP($2K$, δ).

This theorem proves a rigorous and non-asymptotic bound on the length of the input that can be robustly extracted from the network nodes. By showing the RIP property on the network dynamics, the recovery bound given in Equation (4) establishes the recovery performance for any N-length, K-sparse signal from the resulting network state at time N. In short, the number of required nodes scales linearly with the information rate of the signal (i.e., the sparsity level) and poly-logarithmically with the length of the input. The coherence factor $\mu^2(\boldsymbol{\Psi})$, defined as

$$\mu\left(\boldsymbol{\Psi}\right) = \max_{n=1,\ldots,N}\sup_{t\in[0,2\pi]}\left|\sum_{m=0}^{N-1}\boldsymbol{\Psi}_{m,n}e^{-jtm}\right|,$$

expresses the *types* of sparsity that are efficiently stored in the network. Essentially this coherence factor is large (on the order of $\sqrt{N}$) for inputs sparse in the Fourier basis, and is very low (essentially a small constant) for inputs that are sparse in bases different from the Fourier bases (e.g. wavelet transforms). For the extreme case of Fourier-sparse inputs, the number of nodes must again exceed the number of inputs. When this coherence is low and $K \ll M$, this bound is a clear improvement over existing results as it allows for $N > M$. However, this result is restricted to single input streams with one type of low-dimensional structure. The current paper addresses the much more general problem of multidimensional inputs and other types of low-dimensional structure.

3. We use $O(1)$ notation to indicate that a variable is a finite constant.

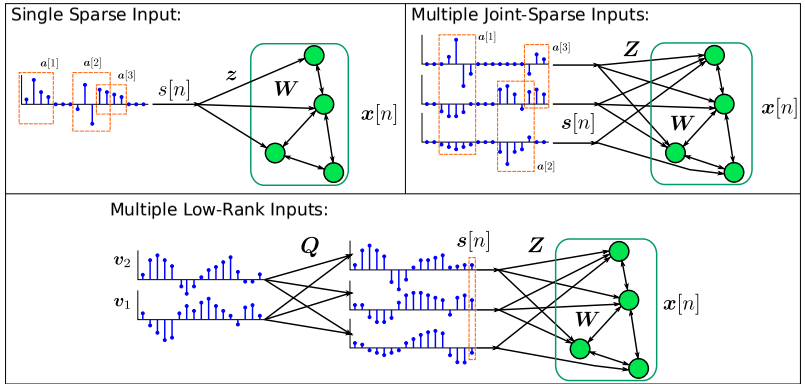


Figure 1: Echo-state networks can efficiently store inputs with a variety of low-dimensional structures. Top-Left: With a single input stream, the coefficients $\boldsymbol{a}$ represent chunks of activity in the input stream $s[n]$ (shown in red boxes). The raw input stream values then drive the network via Equation (1), resulting in a transient network state $\boldsymbol{x}[N]$ that encodes the input stream. Top-Right: In the case of multiple inputs that are jointly sparse, each coefficient can now represent a chunk of activity both across time and across input streams (as depicted by the red boxes). Bottom: When the multiple input streams are instead low-rank, each input stream is instead described by a linear combination of prototypical vectors $\boldsymbol{v}_k$. The matrix $\boldsymbol{Q}$ represents how the prototypical vectors are combined in order to obtain the input sequences fed into the network.

3. STM for Multi-Input Networks

In this work we will use the tools of random matrix theory to establish STM capacity results for recurrent networks under the general conditions of multiple simultaneous input streams and a variety of low-dimensional models. The temporal evolution of the linear network with multiple inputs is similar to the previous ESN definition, with the main difference being that the input at each time-step $\boldsymbol{s}[n] \in \mathbb{R}^L$ is a length L vector that drives the network through a feed-forward matrix $\boldsymbol{Z} \in \mathbb{R}^{M \times L}$ rather than a feed-forward vector,

$$\begin{aligned} \boldsymbol{x}[n] &= \boldsymbol{W}\boldsymbol{x}[n-1] + \sum_{l=1}^{L} \boldsymbol{z}_l s_l[n] + \widetilde{\boldsymbol{\epsilon}}[n] \\ &= \boldsymbol{W}\boldsymbol{x}[n-1] + \boldsymbol{Z}\boldsymbol{s}[n] + \widetilde{\boldsymbol{\epsilon}}[n]. \end{aligned} \tag{7}$$

We denote the columns of $\boldsymbol{Z}$ as $\boldsymbol{z}_l$ to separately notate the vectors mapping each input stream.

We can write the current network state as a linear function of the inputs by iterating Equation (7),

$$\boldsymbol{x}[N] = \sum_{k=1}^{N} \boldsymbol{W}^{N-k}\boldsymbol{Z}\boldsymbol{s}[k] + \boldsymbol{\epsilon},$$

where the error term $\boldsymbol{\epsilon} = \sum_{k=1}^{N} \boldsymbol{W}^{N-k}\widetilde{\boldsymbol{\epsilon}}[k]$ is the accumulated error, and then rewriting sum as a matrix-vector multiply,

$$\boldsymbol{x}[N] = \left[\boldsymbol{Z}, \boldsymbol{W}\boldsymbol{Z}, \cdots, \boldsymbol{W}^{N-1}\boldsymbol{Z}\right] \left[\boldsymbol{s}^T[N], \boldsymbol{s}^T[N-1], \cdots, \boldsymbol{s}^T[1]\right]^T + \boldsymbol{\epsilon}.$$

Depending on the signal statistics in question, we will find it convenient in some cases to express the network dynamics in terms of a linear operator applied to an input matrix, i.e.

$$\boldsymbol{x}[N] = \mathcal{A}(\boldsymbol{S}) + \boldsymbol{\epsilon},$$

where $\boldsymbol{S} = \left[\boldsymbol{s}^T[N], \boldsymbol{s}^T[N-1], \cdots, \boldsymbol{s}^T[1]\right]^T$. In other cases, we find it more convenient to reorganize the columns into an effective measurement matrix applied to a vector of inputs. By defining the eigen-decomposition of $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^{-1}$, we can re-write the dynamics process as

$$\boldsymbol{x}[N] = \boldsymbol{U}\left[\boldsymbol{D}^0\boldsymbol{U}^{-1}\boldsymbol{Z}, \boldsymbol{D}\boldsymbol{U}^{-1}\boldsymbol{Z}, \cdots, \boldsymbol{D}^{N-1}\boldsymbol{U}^{-1}\boldsymbol{Z}\right] \left[\boldsymbol{s}^T[N], \boldsymbol{s}^T[N-1], \cdots, \boldsymbol{s}^T[1]\right]^T + \boldsymbol{\epsilon}.$$

To simplify this expression, we can reorganize the columns of the linear operator (and the rows of the vector of inputs) such that all the inputs corresponding to the l^{th} input vector $\boldsymbol{z}_l$ create a single block. The k^{th} row of the l^{th} block of out matrix is now represented by $\boldsymbol{D}^{k-1}\boldsymbol{U}^{-1}\boldsymbol{z}_l$, which can be written as $\widetilde{\boldsymbol{Z}}_l\boldsymbol{d}_{k-1}$, where $\widetilde{\boldsymbol{Z}}_l = \boldsymbol{U}^{-1}\boldsymbol{z}_l$ and $\boldsymbol{d}_{k-1}$ is the vector of the diagonal elements of $\boldsymbol{D}$ raised to the $(k-1)$ power. We can more concisely by defining the matrix $\boldsymbol{F}$ consisting of the eigenvalues of $\boldsymbol{W}$ raised to different powers (i.e. $\boldsymbol{F}_{i,j} = d_i^{j-1}$), resulting in the expression

$$\boldsymbol{x}[N] = \boldsymbol{U}\left[\widetilde{\boldsymbol{Z}}_1\boldsymbol{F}, \widetilde{\boldsymbol{Z}}_2\boldsymbol{F}, \cdots, \widetilde{\boldsymbol{Z}}_L\boldsymbol{F}\right] \left[\boldsymbol{s}_1^T, \boldsymbol{s}_2^T, \cdots, \boldsymbol{s}_L^T\right]^T + \boldsymbol{\epsilon} = \boldsymbol{A}\widetilde{\boldsymbol{s}} + \boldsymbol{\epsilon}. \tag{8}$$

Since the eigenvalues of $\boldsymbol{W}$ here are restricted to reside on the unit circle, we note that $\boldsymbol{F}$ is a Vandermonde matrix whose rows are Fourier basis vectors. From Equation (8) we see that the current state is simply the sum of L compressed input streams, where the compression for each block essentially performs the same compression as for a single stream, but modulated by the different feed-forward vectors $\boldsymbol{z}_l$.

3.1 Sparse Multiple Inputs

To begin, we consider the direct extension of previous results based on sparsity models to the multi-input setting. In this setting we consider the model where the composite of all input signals is sparse in a basis $\boldsymbol{\Psi} \in \mathbb{R}^{NL \times NL}$ so that $\widetilde{\boldsymbol{s}} = \boldsymbol{\Psi}\widetilde{\boldsymbol{a}}$. This means that each signal stream can be written as $\boldsymbol{s}_l = \sum_{k=1}^{L} \boldsymbol{\Psi}^{l,k}\boldsymbol{a}_k$ where $\boldsymbol{\Psi}^{l,k}$ is the $\{l,k\}^{th}$ $N \times N$ block of $\widetilde{\boldsymbol{\Psi}}$. This signal model captures dependencies between input streams because a given coefficient can influence multiple channels. While in many application the basis $\boldsymbol{\Psi}$ is pre-specified (i.e. wavelet decomposition in image processing; Christopoulos et al., 2000), these bases can also be learned from exemplar data via dictionary learning algorithms (Olshausen and Field, 1996; Aharon et al.). This sparsity model can be a useful model for signals of interest, such as video signals, where similar sparse decompositions have been used for action recognition (Guha and Ward, 2012) and video categorization (Chiang et al., 2013). With this model, we will use a generalized notion of the coherence parameter used in (Charles et al., 2014):

$$\mu_S(\boldsymbol{\Psi}) = \max_{l,k=1,\ldots,L} \max_{n=1,\ldots,N} \sup_{t \in [0,2\pi]} \frac{\left|\sum_{m=0}^{N-1} \boldsymbol{\Psi}_{m,n}^{l,k} e^{-jtm}\right|}{\|\boldsymbol{\Psi}_m^{l,k}\|_2}. \tag{9}$$

In this case, each $N \times N$ block must be different from the Fourier basis to achieve high STM capacity. This restriction is reasonable, since if a single sub-block of $\boldsymbol{\Psi}$ was coherent with the Fourier basis, then at least one input stream could be sparse in a Fourier-like basis and hence would be unrecoverable. Using this network and signal model, we obtain the following theorem on the stability of the network representation:

Theorem 2 *Suppose $NL \geq M$, $N \geq K$ and $N \geq O(1)$. Let $\boldsymbol{U}$ be any unitary matrix of eigenvectors (containing complex conjugate pairs) and the entries of $\boldsymbol{Z}$ be i.i.d. zero-mean Gaussian random variables with variance $\frac{1}{M}$. For M an even integer, denote the eigenvalues of $\boldsymbol{W}$ by $\{e^{jw_m}\}_{m=1}^{M}$. Let the first $M/2$ eigenvalues ($\{e^{jw_m}\}_{m=1}^{M/2}$) be chosen uniformly at random on the complex unit circle (i.e., we chose $\{w_m\}_{m=1}^{M/2}$ uniformly at random from $[0, 2\pi)$) and the other $M/2$ eigenvalues as the complex conjugates of these values. For a given RIP conditioning δ, failure probability $N^{-\log^4 N} \leq \eta \leq \frac{1}{e}$, and coherence $\mu_S(\boldsymbol{\Psi})$ as defined as in Equation* (9)*, if*

$$M \geq C \frac{K}{\delta^2} \mu_S^2(\boldsymbol{\Psi}) \log^5(NL) \log(\eta^{-1}),$$

then $\boldsymbol{A}$ satisfies RIP-$(2K, \delta)$ with probability exceeding $1 - \eta$ for a universal constant C.

The proof of Theorem 2 is provided in Appendix A.1. Note that when $L = 1$, Theorem 2 reduces to Theorem 1. In this result we see that that the number of nodes relies only linearly on the underlying dimensionality (K) and poly-logarithmically on the total size of the input (NL). This means that under favorable coherence and sparsity conditions on the input, the network can again have STM capacities that are higher than the number of nodes in the network. Specifically, showing that $\boldsymbol{A}$ satisfies the RIP property, Theorem 2 ensures that standard recovery guarantees from the sparse inference literature hold. In particular, any K-sparse input is recoverable from the network state at time N up to the error bound of Equation (4) by solving the ℓ_1-regularized least-squares optimization of Equation (3).

3.2 Low Rank Multiple Inputs

Next we consider the case of a very different type of low-dimensional structure where the input signals are correlated but not necessarily sparse. Specifically, in this setting we assume that the inputs arise from a process where R prototypical signals combine linearly to form the various input streams. Such a signal structure could arise, for instance, due to correlations between input streams at spatially neighboring locations. A number of interesting applications display such correlations, including important measurement modalities in neuroscience (e.g. two-photon calcium imaging; Denk et al., 1990; Maruyama et al., 2014 and neural electrophysiological recordings; Berényi et al., 2014; Ahmed and Romberg, 2015), and remote sensing applications (e.g. hyperspectral imagery; Zhang et al., 2014; Veganzones et al., 2016). The applicability of RNNs and machine learning methods to data well described by this low-rank model is also increasingly relevant as there is increasing interest in applying neural network techniques to such data, either for detection (Apthorpe et al., 2016), classification (Chen et al., 2016, 2014), or as samplers via variational auto-encoders (Gao et al., 2016). In this case, we can write out the input matrix in the reduced form $\boldsymbol{S} = \boldsymbol{Q}\boldsymbol{V}^*$, where $\boldsymbol{V}^* \in \mathbb{R}^{R \times N}$ is the matrix whose rows may represent environmental causes generating the data and $\boldsymbol{Q} \in \mathbb{R}^{L \times R}$ represents the mixing matrix that defines the input stream. We

will assume both $L \geq R$ and $N \geq R$, meaning that $\boldsymbol{S}$ is low-rank. With this model we use a definition of coherence given by:

$$\mu_L^2 = R^{-1} \sup_{\omega \in [0, 2\pi]} ||\boldsymbol{V}^* \boldsymbol{f}_\omega||_2^2 . \tag{10}$$

where $\boldsymbol{f}_\omega = [1, e^{-j\omega}, \cdots e^{-j(N-1)\omega}]^T$ is the Fourier vector with frequency ω. This coherence parameter mirrors the coherence used for the sparse-input case. As μ_S measured the similarity between the measurement vectors and the sparsity basis $\boldsymbol{\Psi}$, μ_L measures the similarity between the measurements and the left singular vectors of the measured matrix. The intuition here is that measurements that align with the left singular vectors are unlikely to measure significant information about the $\boldsymbol{S}$.

To analyze the STM of the network dynamics with respect to low-rank signal statistics, we leverage the dual certificate approach (Candès and Plan, 2010, 2011a; Ahmed and Romberg, 2015) to derive the following theorem,

Theorem 3 *Suppose $NL \geq M$, $N \geq R$, $N \geq O(1)$ and $L \geq O(1)$. Let $\boldsymbol{z}$ be i.i.d. zero-mean Gaussian random variables with variance $\frac{1}{M}$. For M an even integer, denote the eigenvalues of $\boldsymbol{W}$ by $\{e^{jw_m}\}_{m=1}^{M}$. Let the first $M/2$ eigenvalues ($\{e^{jw_m}\}_{m=1}^{M/2}$) be chosen uniformly at random on the complex unit circle (i.e., we chose $\{w_m\}_{m=1}^{M/2}$ uniformly at random from $[0, 2\pi)$) and the other $M/2$ eigenvalues as the complex conjugates of these values. For a given coherence μ_L as defined as in Equation* (10)*, if*

$$M \geq cR\left(N + \mu_L^2 L\right) \log^3(LN),$$

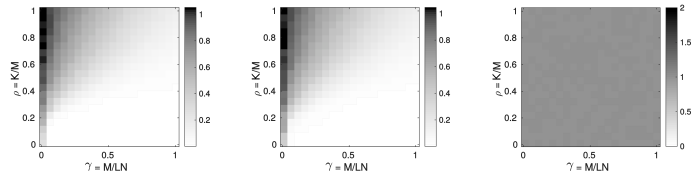
then, with probability at least $1 - O((LN)^{1-\beta}$, the minimization in Equation (5) *recovers the rank-R input matrix $\boldsymbol{S}$ up to the error bound in Equation* (6)*.*

The proof of Theorem 3 is in Appendix A.2 and follows a golfing scheme to find an inexact dual certificate. In fact, we note that since our architecture is extremely similar mathematically to the architecture in (Ahmed and Romberg, 2015), our proof is also very similar. The main difference is that due to the unbounded nature of our distributions (i.e. the feed-forward vectors $\boldsymbol{Z}$ are Gaussian random variables) and the fact that our Fourier vectors are on the unit circle (rather than gridded), we can consider our proof as a generalization of the proof in (Ahmed and Romberg, 2015).

Theorem 3 is qualitatively similar to Theorem 2 in the way the STM capacity scales. In this case, the bound still scales linearly with the information rate as captured by the number of elements in the left and right matrices that compose $\boldsymbol{S}$: $RN + RL$. Interestingly, due to the left singular vectors interacting with the measurement operator first, the coherence term only affects the portion of the bound related to the number of elements in $\boldsymbol{Q}$. Additionally, as before, the number of total inputs LN only impacts the bound poly-logarithmically.

4. Simulation

To empirically verify that these theoretical STM scaling laws are representative of the empirical behavior, we generated a number of random networks and evaluated the recovery of (sparse or low-rank) input sequences in the presence of noise. For each simulation we generate a $M \times M$ random



(a) RMSE: No basis (b) RMSE: Haar wavelet basis (c) RMSE: DCT basis

Figure 2: ESNs can have high STM capacity for multidimensional sparse inputs. Relative mean-squared error (rMSE) of the recovery for canonical sparse inputs (a) and Haar wavelet-sparse inputs (b) is very low for a range of sparsity and network sizes satisfying $M < LN$. The rMSE for DCT-sparse inputs (c), as predicted by our theoretical results, remains high (approximately 100% error).

orthogonal connectivity matrix $\boldsymbol{W}$[4] and a $M \times L$ random Gaussian feed-forward matrix $\boldsymbol{Z}$. In both cases we fixed the number of inputs to $L = 40$ and the number of time-steps to $N = 100$ while varying the network size M and underlying dimensionality of the input (i.e., the sparsity level or the input matrix rank). For the sparse input simulations, inputs were chosen with a uniformly random support pattern with random Gaussian values on the support. For low-rank simulations, the right singular vectors were chosen to be Gaussian random vectors, and the left singular values were chosen at random from a number of different basis sets.

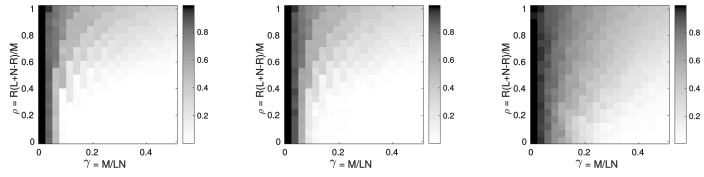
In Figure 2 we show the relative mean-squared error of the input recovery as a function of the sparsity-to-network size ratio $\rho = K/M$ and the network size-to-input ratio $\gamma = M/NL$. Each pixel value represents the average recovery relative mean-squared error (rMSE), as calculated by

$$\text{RMSE} = \frac{\|\widehat{\boldsymbol{s}} - \boldsymbol{s}\|_2^2}{\|\boldsymbol{s}\|_2^2},$$

over 20 randomly generated trials with a noise level of $\|\boldsymbol{\epsilon}\|_2 \approx 0.01$. We show results for recovery of three different types of sparse signals: signals sparse in the canonical basis, signals sparse in a Haar wavelet basis, and signals sparse in a discrete cosine transform (DCT) basis. As our theory predicts, for canonical- and Haar wavelet-sparse signals the network has very favorable STM capacity results. The fact that the capacity achieves $M < NL$ is demonstrated by the area left of the $M = NL$ point ($\gamma = 1$) where the signal is recovered with high accuracy. Likewise, for the DCT-sparse signals we find that the inputs are never recovered well for any $M < NL$. This behavior is also predicted by our theory because of the unfavorable coherence properties of the DCT basis.

For the low-rank trials we see that recovery of low-rank inputs for a range of $M < LN$ is possible as predicted by the theoretical results. As with the sparse input case we consider three types of low-rank inputs. Instead of changing the sparsity basis, however, we change the right singular vectors $\boldsymbol{V}$ of the low-rank input matrix $\boldsymbol{S}$. We explore the cases where the elements of $\boldsymbol{V}$

4. Orthogonal connectivity matrices were obtained by running an orthogonalization procedure on a random Gaussian matrix.



(a) RMSE: No basis (b) RMSE: Haar wavelet basis (c) RMSE: DCT basis

Figure 3: ESNs can have high STM capacity for multidimensional inputs with low-rank structure. The rMSE of the input recovery for inputs with canonical right singular vectors (a) and Haar wavelets for right singular vectors (b) is very low for a range of input rank and network sizes satisfying $M < LN$. The rMSE for inputs with DCT right singular vectors (c) remains high, a behavior predicted by our theoretical results.

are chosen from the canonical basis, the haar basis and the DCT basis. These results are shown in Figure 3 with plots similar to those in Figure 2, but with only showing the range $\gamma < 0.5$ to reduce computational time. As our theory predicts, the recovery of inputs with canonical- and Haar right singular vectors is more accurate for a larger range of ρ, γ pairs than the inputs with DCT right singular vectors.

5. Conclusions

Determining the fundamental limits of memory in recurrent networks is critical to understanding their behavior in machine learning tasks. In this work we show that randomly connected echo-state networks can exploit the low-dimensional structure in multidimensional input streams to achieve very high short-term memory capacity. Specifically, we show non-asymptotic bounds on recovery error for input sequences that have underlying low-dimensional statistics described by either joint sparsity or low-rank properties (with no sparsity). For multiple sparse inputs, we find that the network size must be linearly proportional to the input sparsity and only logarithmically dependent on the total input size (a combination of the input length and number of inputs). For inputs with low-rank structure, we find a similar dependency where the network size depends linearly on the underlying dimension of the inputs (the product of the input rank with the input length and the number of inputs) and logarithmically on the total input size. Both results continue to demonstrate that ESNs can have STM capacities much larger than the network size.

These results are a significant (conceptual and technical) generalization over previous work that provided theoretical guarantees in the case of a single sparse input (Charles et al., 2014). While the linear ESN structure is a simplified model, rigorous analysis of these networks has remained elusive due to the recurrence itself. These results isolate the properties of the transient dynamics due to the recurrent connections, and may provide one foundation for which to explore the analysis of other complex network attributes such as nonlinearities and spiking properties. We also note that knowledge of how well neural networks compresses structured signals could indicate methods to pick the size of recurrent network layers. Specifically, if a task is thought to require a certain time-

frame of an input signal, the overall sparsity (or rank) of the signal in that time-frame can be used in conjunction with the length of that time-frame to give a lower bound for the required number of nodes in the recurrent layer of the network.

While the current paper is restricted to orthogonal connectivity matrices, previous work (Charles et al., 2014) has shown that a number of network structures can satisfy these criteria, including some types of small-world network topologies. Additionally, we explored here low-rank and sparse inputs separately. The methods we have used to prove Theorem 3, however, have also been used to analyze recovery signals with other related structures (e.g. matrices that can be decomposed into the sum of a sparse and low-rank matrix) from compressive measurements (Candès and Plan, 2011b). Our bounds presented here therefore could open up avenues for similar analysis of other low-dimensional signal classes.

While the context of this paper is focused on the role of recurrent networks as a tool in machine learning tasks, these results may also lead to a better understanding of the STM properties in other networked systems (e.g., social networks, biological networks, distributed computing, etc.). With respect to the literature relating recurrent ESN and liquid-state machines to working memory in biological systems, the notion of sparsity has much of the same flavor as the concept of *chunking* (Gobet et al., 2001). Chunking is the concept of humans learning to remember items in highly correlated groups rather than remembering items individually as a way of artificially increasing their working memory. Similarly, the use of sparsity bases allow RNNs to 'chunk' items according to the basis elements. Thus, each basis counts only as one item (the true underlying sparsity) and the network needs only store these elements rather than storing every input separately.

Acknowledgments

The authors are grateful to S. Bahmani, A. Ahmed and J. Romberg for valuable discussions related to this work. This work was supported in part by ONR grant N00014-15-1-2731 and NSF grants CCF-0830456 and CCF-1409422.

A. Appendix

A.1 RIP for Multiple Gaussian Feed-forward Vectors

In this appendix we prove Theorem 2, showing that the matrix representing the network evolution with L inputs and *i.i.d.* Gaussian feed-forward vectors satisfies the RIP. Recall that we have $\boldsymbol{x}[N] = \boldsymbol{A}\widetilde{\boldsymbol{s}}$, where $\boldsymbol{A} \in \mathbb{R}^{M\times NL}$ is derived in Section 3 and that $\widetilde{\boldsymbol{s}}$ (the vectorization of $\boldsymbol{S}$) is sparse with respect to the basis Ψ, meaning that there is a K-sparse signal $\boldsymbol{a}$ such that $\widetilde{\boldsymbol{s}} = \Psi\boldsymbol{a}$. Similar to (Charles et al., 2014), this proof is based on showing conditions on M such that $\boldsymbol{A}$ satisfies the RIP with respect to $\boldsymbol{\Psi}$, i.e.

$$(1-\delta)\left|\left|\boldsymbol{a}\right|\right|_2^2 \leq \left|\left|\boldsymbol{A}\boldsymbol{\Psi}\boldsymbol{a}\right|\right|_2^2 \leq (1+\delta)\left|\left|\boldsymbol{a}\right|\right|_2^2,$$

holds with high probability for all K-sparse $\boldsymbol{a}$. This is equivalent to bounding the following probability of the event

$$\left|\left|(\boldsymbol{A}\boldsymbol{\Psi})^H\boldsymbol{A}\boldsymbol{\Psi} - I\right|\right|_K \leq \delta. \tag{11}$$

where the norm $\left|\left|\boldsymbol{A}\right|\right|_K$ is defined as

$$\left|\left|\boldsymbol{A}\right|\right|_K := \sup_{\boldsymbol{y}\ is\ K-sparse} \frac{\boldsymbol{y}^H\boldsymbol{A}\boldsymbol{y}}{\left|\left|\boldsymbol{y}\right|\right|_2^2}.$$

First we bound the expectation of $\left|\left|(\boldsymbol{A}\boldsymbol{\Psi})^H\boldsymbol{A}\boldsymbol{\Psi} - I\right|\right|_K$, and use the result to bound the tail probability of the event (11).

Proof

A.1.1 EXPECTATION

First we let

$$\widehat{\boldsymbol{A}} = \begin{bmatrix}\widetilde{\boldsymbol{Z}}_1\boldsymbol{F} & \widetilde{\boldsymbol{Z}}_2F & \cdots & \widetilde{\boldsymbol{Z}}_L\boldsymbol{F}\end{bmatrix}\boldsymbol{\Psi},$$

Since $(\boldsymbol{A}\boldsymbol{\Psi})^H\boldsymbol{A}\boldsymbol{\Psi} = \widehat{\boldsymbol{A}}^H\widehat{\boldsymbol{A}}$, then for any $\boldsymbol{a} \in \mathbb{R}^{NL}$, $\left|\left|\boldsymbol{A}\boldsymbol{\Psi}\boldsymbol{a}\right|\right|_2 = \left|\left|\widehat{\boldsymbol{A}}\boldsymbol{a}\right|\right|_2$. Therefore we only need to prove that

$$\left|\left|\widehat{\boldsymbol{A}}^H\widehat{\boldsymbol{A}} - \boldsymbol{I}\right|\right|_K \leq \delta,$$

holds with high probability when M is large enough. Let V_i^{H} denote the ith row of $\widehat{\boldsymbol{A}}$, i.e.,

$$V_i^{\mathrm{H}} = \sum_{l=1}^{L}\tilde{\boldsymbol{z}}_{i,l}\boldsymbol{F}_i^H\boldsymbol{\Psi}_i.$$

Let $\boldsymbol{B}_1 = \sum_{i=1}^{M/2}V_iV_i^H - \boldsymbol{I}/2$ and $\boldsymbol{B}_2 = \sum_{i=M/2+1}^{M}V_iV_i^H - \boldsymbol{I}/2$; it is easy to check that $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$ are complex conjugates, and that $\widehat{\boldsymbol{A}}^H\widehat{\boldsymbol{A}} - \boldsymbol{I} = \boldsymbol{B}_1 + \boldsymbol{B}_2$. Therefore $\left|\left|\widehat{\boldsymbol{A}}^H\widehat{\boldsymbol{A}} - \boldsymbol{I}\right|\right|_K \leq 2\left|\left|\boldsymbol{B}_1\right|\right|_K$, and we only need to show that $\left|\left|\boldsymbol{B}_1\right|\right|_K \leq \delta$ with high probability when M is large enough. We can easily see that when $i \neq j$, $1 \leq i, j \leq M/2$, V_i^H and V_j^H are independent.

First we show that $\mathbb{E}\left[\left|\left|\boldsymbol{B}_1\right|\right|_K\right]$ is small with high probability when M is large enough. We then show that $\left|\left|\boldsymbol{B}_1\right|\right|_K$ is concentrated around its mean with high probability when M is large enough. By Lemma 6.7 in Rauhut (2010), for a Rademacher sequence ϵ_i, $i = 1, ..., M/2$, we have

$$\mathbb{E}\left[\left|\left|\boldsymbol{B}_1\right|\right|_K\right] = \mathbb{E}\left[\left|\left|\sum_{i=1}^{M/2}(V_iV_i^H - \frac{1}{M}\boldsymbol{I})\right|\right|_K\right] \leq 2\mathbb{E}\left[\left|\left|\sum_{i=1}^{M/2}\epsilon_iV_iV_i^H\right|\right|_K\right].$$

We can now apply Lemma 8.2 from Rauhut (2010), giving us

$$\begin{aligned}
\mathbb{E}\left[\left|\left|\boldsymbol{B}_1\right|\right|_K\right] &\leq 2\mathbb{E}\left[\left|\left|\sum_{i=1}^{M/2}\epsilon_iV_iV_i^H\right|\right|_K\right] \\
&\leq 2\mathbb{E}\left[\mathbb{E}\left[C_0V_{\max}\sqrt{K}\log\left(100K\right)\sqrt{\log\left(4NL\right)\ln\left(5M\right)}\right.\right. \\
&\qquad \left.\left.\cdot\sqrt{\left|\left|\sum_{i=1}^{M/2}V_iV_i^H\right|\right|_K}\ \middle|V_i, i = 1, ..., M/2\right]\right] \\
&\leq \sqrt{C_1K\log^4(NL)}\sqrt{\mathbb{E}\left[V_{\max}^2\right]\mathbb{E}\left[\left|\left|\boldsymbol{B}_1\right|\right|_K + \frac{1}{2}\right]},
\end{aligned} \tag{12}$$

where the last inequality results due to $25K \leq NL$, $5M \leq 4NL$, the Cauthy-Schwarz inequality and the triangle inequality, and $V_{\max} = \max_{1\leq l\leq M/2} ||V_l||_\infty$. Note that the pth element of V_l^H can be written

$$V_l^H(p) = \sum_{i=1}^{L} \tilde{\boldsymbol{z}}_{l,i} \boldsymbol{F}_l^H \boldsymbol{\Psi}_i(p).$$

and since we know that

$$\sum_{i=1}^{L} |\boldsymbol{F}_l^H \boldsymbol{\Psi}_i(p)|^2 \leq \sum_{i=1}^{L} ||\boldsymbol{\Psi}_i(p)||_2^2 \, \mu^2(\boldsymbol{\Psi}) = \mu^2(\boldsymbol{\Psi}),$$

and

$$V_{\max} = \max_{\substack{1\leq l\leq M/2 \\ 1\leq p\leq NL}} |V_l(p)|$$

we can now use Corollary 5. Setting $Q = MNL/2$, $\mu_0 = \mu(\boldsymbol{\Psi})$ in Corollary 5 yields

$$\mathcal{P}\left[V_{\max}^2 > \frac{\mu^2(\boldsymbol{\Psi})}{M} \log \frac{MNL}{2\eta}\right] \leq \eta, \tag{13}$$

and

$$\mathbb{E}\left[V_{\max}^2\right] \leq \frac{\mu^2(\boldsymbol{\Psi})}{M}(\log \frac{MNL}{2} + 1) \leq C_2 \frac{\mu^2(\boldsymbol{\Psi})}{M} \log(NL). \tag{14}$$

Returning to the inequality in Equation (12), considering (14), we have

$$\mathbb{E}\left[||\boldsymbol{B}_1||_K\right] \leq a\sqrt{\mathbb{E}\left[||\boldsymbol{B}_1||_K\right] + 1},$$

where $a = \sqrt{C_1 C_2 K \log^5(NL)\mu^2(\boldsymbol{\Psi})/M}$. Then $\mathbb{E}\left[||\boldsymbol{B}_1||_K\right] \leq \frac{a^2}{2} + a\sqrt{\frac{1}{2} + \frac{a^2}{4}}$. When $a \leq 1/2$, we get $\mathbb{E}\left[||\boldsymbol{B}_1||_K\right] \leq a$. Let $0 < \boldsymbol{A}' \leq a \leq 1/2$, and we can conclude that when

$$M \geq \frac{C_3 K \mu^2(\boldsymbol{\Psi}) \log^5(NL)}{\delta'},$$

then

$$\mathbb{E}\left[||\boldsymbol{B}_1||_K\right] \leq \delta'.$$

A.1.2 TAIL BOUND

Now we study the tail bound of $||\boldsymbol{B}_1||_K$. First we construct a second set of random variables V_l', which are independent of V_l and are identically distributed as V_l. Additionally, we let

$$\widetilde{\boldsymbol{B}}_1 = \sum_{i=1}^{M/2} \left(V_i V_i^H - V_i' V_i'^H\right),$$

and then according to Charles et al. (2014), there is

$$\mathbb{E}\left[\left|\left|\widetilde{\boldsymbol{B}}_1\right|\right|_K\right] \leq 2\mathbb{E}\left[||\boldsymbol{B}_1||_K\right], \tag{15}$$

$$\mathcal{P}\left[||\boldsymbol{B}_1||_K > 2\mathbb{E}\left[||\boldsymbol{B}_1||_K\right] + u\right] \leq 2\mathcal{P}\left[\left|\left|\widetilde{\boldsymbol{B}}_1\right|\right|_K > u\right]. \tag{16}$$

Now since we have

$$\left|\left|V_i V_i^H - V_i' V_i'^H\right|\right|_K \leq 2\max\{\left|\left|V_i V_i^H\right|\right|_K, \left|\left|V_i' V_i'^H\right|\right|_K\},$$

and

$$\left|\left|V_i V_i^H\right|\right|_K \leq \sup_{y \text{ is } K-sparse} ||V_i||_\infty^2 \frac{||y||_1^2}{||y||_2^2} \leq K V_{\max}^2,$$

then we know that $\max_{1\leq i\leq M/2} \left|\left|V_i V_i^H\right|\right|_K \leq K V_{\max}^2$ and $\max_{1\leq i\leq M/2} \left|\left|V_i' V_i'^H\right|\right|_K \leq K V_{\max}'^2$, where $V_{\max}' = \max_{1\leq l\leq M/2} ||V_l'||_\infty$ then by Equation (13), we obtain

$$\mathcal{P}\left[\max_{1\leq i\leq M/2} \left|\left|V_i V_i^H\right|\right|_K > \frac{K\mu^2(\boldsymbol{\Psi})}{M} \log \frac{MNL}{2\eta}\right] \leq \mathcal{P}\left[V_{\max}^2 > \frac{\mu^2(\boldsymbol{\Psi})}{M} \log \frac{MNL}{2\eta}\right] \leq \eta.$$

Since the probability theorems depend on bounded random variables, we define $\mathbf{F}$ to denote the following event

$$\mathbf{F} = \left\{\max\left\{\max_{1\leq i\leq M/2} \left|\left|V_i V_i^H\right|\right|_K, \max_{1\leq i\leq M/2} \left|\left|V_i' V_i'^H\right|\right|_K\right\} \leq \frac{K\mu^2(\boldsymbol{\Psi})}{M} \log \frac{MNL}{2\eta}\right\},$$

such that $\mathcal{P}\left[\mathbf{F}^C\right] \leq 2\eta$. Furthermore, we define $\mathbf{I}_F$ as the indicator function of $\mathbf{F}$, and let $\widehat{\boldsymbol{B}}_1 = \sum_{i=1}^{M/2} \xi_i \left(V_i V_i^H - V_i' V_i'^H\right) \mathbf{I}_F$, where $\xi = \{\xi_i\}$, $i = 1, 2, ..., M/2$ is a Rademacher sequence and independent of V_i. The truncated variable $Y_i = \xi_i \left(V_i V_i^H - V_i' V_i'^H\right) \mathbf{I}_F$ has a symmetric distribution and $||Y_i||_K$ is bounded by $B_{\max} := \frac{2K\mu^2(\boldsymbol{\Psi})}{M} \ln \frac{MNL}{2\eta}$. By Proposition 19 in Tropp et al. (2009), we have

$$\mathcal{P}\left[\left|\left|\widehat{\boldsymbol{B}}_1\right|\right|_K > C_4(u\mathbb{E}\left[\left|\left|\widehat{\boldsymbol{B}}_1\right|\right|_K\right] + tB_{\max})\right] \leq e^{-u^2} + e^{-t}, \tag{17}$$

for all $u,\ t \geq 1$. Following Tropp et al. (2009), we find that

$$\mathcal{P}\left[\left|\left|\widetilde{\boldsymbol{B}}_1\right|\right|_K > v\right] \leq \mathcal{P}\left[\left|\left|\widehat{\boldsymbol{B}}_1\right|\right|_K > v\right] + \mathcal{P}\left[\mathbf{F}^C\right], \tag{18}$$

and

$$\mathbb{E}\left[\left|\left|\widehat{\boldsymbol{B}}_1\right|\right|_K\right] \leq \mathbb{E}\left[\left|\left|\widetilde{\boldsymbol{B}}_1\right|\right|_K\right]. \tag{19}$$

By combining Equations (17), (18), and (19), we get

$$\mathcal{P}\left[\left|\left|\widetilde{\boldsymbol{B}}_1\right|\right|_K > C_4(u\mathbb{E}\left[\left|\left|\widetilde{\boldsymbol{B}}_1\right|\right|_K\right] + tB_{\max})\right] \leq e^{-u^2} + e^{-t} + 2\eta.$$

In Equation (20), let $\eta < 1/e$, $u = \sqrt{\log \eta^{-1}}$ and $t = \log \eta^{-1}$. These values yield

$$\mathcal{P}\left[\left|\left|\widetilde{\boldsymbol{B}}_1\right|\right|_K > C_4(\sqrt{\log \eta^{-1}}\mathbb{E}\left[\left|\left|\widetilde{\boldsymbol{B}}_1\right|\right|_K\right] + (\log \eta^{-1})B_{\max})\right] \leq 4\eta. \tag{20}$$

Now we can combine Equations (15), (16), and (20) to get

$$\mathcal{P}\left[||\boldsymbol{B}_1||_K > 2\mathbb{E}\left[||\boldsymbol{B}_1||_K\right] + 2C_4\sqrt{\log\eta^{-1}}\mathbb{E}\left[||\boldsymbol{B}_1||_K\right] + C_4(\log\eta^{-1})B_{\max}\right] \leq 8\eta.$$

By choosing

$$M \geq \frac{C_3 K\mu^2(\boldsymbol{\Psi})\log^5(NL)}{\delta'^2}, \tag{21}$$

where $\delta' < 1/2$, we obtain

$$\mathcal{P}\left[||\boldsymbol{B}_1||_K > 2\delta' + 2C_4\delta'\sqrt{\log\eta^{-1}} + C_5\log\eta^{-1}\frac{\delta'^2\log\left(\frac{1}{2}MNL\eta^{-1}\right)}{\log^5(NL)}\right] \leq 8\eta.$$

We observe now that

$$\frac{\log(\frac{1}{2}MNL\eta^{-1})}{\log^5(NL)} \leq \frac{2\log(NL) + \log\eta^{-1}}{\log^5(NL)},$$

indicating that when $\eta \geq (NL)^{-\log^4(NL)}$, i.e., $\log\eta^{-1} \leq \log^5(NL)$, then

$$\frac{\log(\frac{1}{2}MNL\eta^{-1})}{\log^5(NL)} \leq C_6,$$

, for a constant C_6. This inequality reduces our probability statement to

$$\mathcal{P}\left[||\boldsymbol{B}_1||_K > 2\delta' + 2C_4\delta'\sqrt{\log\eta^{-1}} + C_5C_6\log\eta^{-1}\delta'^2\right] \leq 8\eta.$$

For an arbitrary δ with $0 < \delta < 1/2$, we let

$$\delta' = \frac{\delta}{2C_4C_7\sqrt{\log\eta^{-1}}}, \tag{22}$$

resulting in

$$2C_4\delta'\sqrt{\log\eta^{-1}} = \frac{\delta}{C_7},$$

$$2\delta' = \frac{\delta}{C_4C_7\sqrt{\log\eta^{-1}}} \leq \frac{\delta}{C_4C_7},$$

$$C_5C_6\log\eta^{-1}\delta'^2 = C_5C_6\frac{\delta^2}{4C_4^2C_7^2} \leq \frac{C_5C_6}{8C_4^2C_7^2}\delta.$$

Then we can see that if

$$C_7 \geq \max\left\{3, \frac{3}{C_4}, \sqrt{\frac{3C_5C_6}{8C_4^2}}\right\},$$

then

$$\mathcal{P}\left[||\boldsymbol{B}_1||_K > \frac{\delta}{3} + \frac{\delta}{3} + \frac{\delta}{3}\right] < 8\eta.$$

Now by plugging (22) into (21), we know that when $(NL)^{-\ln^4(NL)} \leq \eta < 1/e$, there exists a constant C such that when

$$M \geq \frac{CK\mu^2(\boldsymbol{\Psi})\log^5(NL)\log(\eta^{-1})}{\delta^2},$$

there is

$$\mathrm{P}(||\boldsymbol{B}_1||_K > \delta) \leq 8\eta,$$

which completes the proof. ■

A.1.3 LEMMAS FOR THEOREM 2

Lemma 4 *Suppose we have n complex Gaussian random variables, $z_1,\ z_2, \cdots\ z_n$, $z_i = x_i + \mathrm{j}y_i$, where x_i and y_i denote the real and imaginary parts of z_i. Let x_i and y_i i.i.d Gaussian r.v.'s with mean 0 and variance $1/2M$. $\phi_1,\ \phi_2,\ \cdots\ \phi_n$ r.v.'s which are independent of z_i for all i and satisfy*

$$\sum_{i=1}^{n}|\phi_i|^2 \leq \mu_0^2,$$

then for $w = \sum_{i=1}^{n} z_i\phi_i$,

$$\mathcal{P}\left[|w|^2 > u\right] \leq e^{-\frac{Mu}{\mu_0^2}}.$$

Proof We use x and y to denote the real and imaginary parts of w, and a_i and b_i to denote the real and imaginary parts of ϕ_i. We have

$$w = \sum_{i=1}^{n}(a_ix_i - b_iy_i) + \mathrm{j}\sum_{i=1}^{n}(a_iy_i + b_ix_i)x + \mathrm{j}y.$$

Then conditioned on a_i and b_i, x and y have distribution $\mathcal{N}(0, \frac{1}{2M}\sum_{i=1}^{n}(a_i^2 + b_i^2))$.

The next step is to prove the conditional independence of x and y. Since

$$\mathrm{Cov}(a_ix_i - b_iy_i, a_iy_i + b_ix_i|a_i, b_i) = \frac{a_ib_i}{2M} - \frac{a_ib_i}{2M} = 0,$$

when $i \neq j$, x_i, y_i are independent of x_j, y_j, and x and y are conditionally independent. Thus, conditioned on a_i and b_i,

$$\frac{2M}{\sum_{i=1}^{n}(a_i^2 + b_i^2)}|w|^2,$$

is χ^2 distributed. We use χ_2 to denote a two-degree χ^2 distributed random variable. According to the results on χ^2 distributions, we have

$$\mathcal{P}\left[|w|^2 > u|a_i, b_i\right] = \mathcal{P}\left[\chi_2 > \frac{2Mu}{\sum_{i=1}^{n}(a_i^2 + b_i^2)}|a_i, b_i\right] = e^{-\frac{Mu}{\sum_{i=1}^{n}(a_i^2+b_i^2)}} \leq e^{-\frac{Mu}{\mu_0^2}}. \tag{23}$$

Noticing that Equation (23) holds for all possible values of a_i and b_i completes the proof. ■

Corollary 5 *For Q r.v.'s, $w_1,\ w_2,\ \cdots,\ w_Q$, let $w_i = \sum_{l=1}^{n} z_{il}\phi_{il}$ and $z_{il} = x_{il} + \mathrm{j}y_{il}$, where x_{il}, y_{il}, $1 \leq i \leq Q$, $1 \leq l \leq n$ are i.i.d. Gaussian distributed with mean 0 and variance $1/2M$. Suppose for any i, there is*

$$\sum_{l=1}^{n} |\phi_{il}|^2 \leq \mu_0^2.$$

And let $w_{\max} = \max_{1\leq i\leq Q} |w_i|$, then for $\eta > 0$, we have

$$\mathcal{P}\left[w_{\max}^2 > \frac{\mu_0^2}{M}\log\frac{Q}{\eta}\right] \leq \eta,$$

and

$$\mathbb{E}\left[w_{\max}^2\right] \leq \frac{\mu_0^2}{M}(\ln Q + 1).$$

Proof According to Lemma 4 and by using union bound, we have

$$\mathcal{P}\left[w_{\max}^2 > u\right] \leq Qe^{-\frac{Mu}{\mu_0^2}}.$$

Let $\eta = Qe^{-\frac{Mu}{\mu_0^2}}$ and there is

$$\mathcal{P}\left[w_{\max}^2 > \frac{\mu_0^2}{M}\log\frac{Q}{\eta}\right] \leq \eta.$$

Then we have

$$\begin{aligned}\mathbb{E}\left[w_{\max}^2\right] &= \int_0^\infty \mathcal{P}\left[w_{\max}^2 > u\right] du \\ &\leq \int_0^{\frac{\mu_0^2}{M}\ln Q} 1du + \int_{\frac{\mu_0^2}{M}\log Q}^{\infty} Qe^{-\frac{Mu}{\mu_0^2}} du \\ &= \frac{\mu_0^2}{M}(\log Q + 1).\end{aligned}$$

■

A.2 Proof of Low-rank Recovery

In this appendix we prove Theorem 3 where a low-rank input matrix $\boldsymbol{S}$ can be recovered from the network state $\boldsymbol{x}[N]$ via nuclear norm optimization (Candès and Tao, 2010; Candès and Plan, 2010; Recht et al., 2010). To prove this theorem we use the dual certificate approach used to prove similar results in (Ahmed and Romberg, 2015; Candès and Plan, 2011a). In this methodology we seek a certificate $\boldsymbol{Y}$ whose projections into and out of the space spanned by the singular vectors of $\boldsymbol{S}$ are bounded appropriately. Specifically if we consider the singular value decomposition of $\boldsymbol{S}$ as

$$\boldsymbol{S} = \boldsymbol{Q}\boldsymbol{\Sigma}\boldsymbol{V}^*$$

and we consider the projection $\mathcal{P}_T$ which projects a matrix into the space T spanned by the left and right singular vectors,

$$\mathcal{P}_T(\boldsymbol{W}) = \boldsymbol{Q}\boldsymbol{Q}^*\boldsymbol{W} + \boldsymbol{W}\boldsymbol{V}\boldsymbol{V}^* - \boldsymbol{Q}\boldsymbol{Q}^*\boldsymbol{W}\boldsymbol{V}\boldsymbol{V}^* \tag{24}$$

the conditions for the dual certificate are that $\mathcal{A}$ is injective on T and there exists a matrix $\boldsymbol{Y}$ which satisfies

$$\left|\left|\mathcal{P}_T(\boldsymbol{Y}) - \boldsymbol{Q}\boldsymbol{V}^H\right|\right|_F \leq \frac{1}{2\sqrt{2}\gamma} \tag{25}$$

$$||\mathcal{P}_{T^\perp}(\boldsymbol{Y})|| \leq \frac{1}{2} \tag{26}$$

where the projection $\mathcal{P}_{T^\perp}$ is the projection onto the perpendicular space to T,

$$\mathcal{P}_{T^\perp}(\boldsymbol{W}) = (\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^*)\boldsymbol{W}(\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^*)$$

The remainder of this proof will be devoted to demonstrating that there does exist a certificate $\boldsymbol{Y}$ by iteratively devising $\boldsymbol{Y}$ via a golfing scheme (Gross, 2011; Candès and Plan, 2011a; Ahmed et al., 2014). The golfing scheme essentially generates an iterative method which defined a series of certificate vectors $\boldsymbol{Y}_k$ for $k \in [1, \cdots, \kappa]$ which converge to a certificate $\boldsymbol{Y}_\kappa$ which satisfies the necessary conditions. As in (Ahmed and Romberg, 2015), we can initialize the 0^{th} iterate to zero, and define the k^{th} iterate in terms of the $\boldsymbol{Y}_{k-1}$ as

$$\boldsymbol{Y}_k = \boldsymbol{Y}_{k-1} + \kappa\mathcal{A}_k^*\mathcal{A}_k(\boldsymbol{Q}\boldsymbol{V}^* - \mathcal{P}_T(\boldsymbol{Y}_{k-1})),$$

where

$$\mathcal{A}(\boldsymbol{W}) = \mathrm{vec}\left(\langle \boldsymbol{A}_n, \boldsymbol{W}\rangle\right). \tag{27}$$

We can see that since every iterate has $\mathcal{A}_k^*$ applied to it, every iteration is projected in to the range of $\mathcal{A}^*$, indicating that the final iteration $\boldsymbol{Y}$ will also be in the range of $\mathcal{A}^*$. In (Ahmed and Romberg, 2015), Asif and Romberg define a simpler iteration

$$\widetilde{\boldsymbol{Y}}_k = (\mathcal{P}_T - \kappa\mathcal{P}_T\mathcal{A}_k^*\mathcal{A}_k\mathcal{P}_T)\widetilde{\boldsymbol{Y}}_{k-1},$$

which is expressed in terms of the modified certificate

$$\widetilde{\boldsymbol{Y}}_k = \mathcal{P}_T(\boldsymbol{Y}_k) - \boldsymbol{Q}\boldsymbol{V}^*.$$

What remains now is to demonstrate that this iterative procedure converges, with high probability, to a certificate which satisfies the desired dual certificate conditions. We start by using Lemma 9 and observing that the Forbenious norm of the k^{th} iterate is well bounded with probability $1 - O((LN)^{-\beta})$ by

$$\begin{aligned}\left|\left|\widetilde{\boldsymbol{Y}}_k\right|\right|_F &\leq \max_k ||\mathcal{P}_T - \kappa\mathcal{P}_T\mathcal{A}_k^*\mathcal{A}_k\mathcal{P}_T|| \left|\left|\widetilde{\boldsymbol{Y}}_{k-1}\right|\right|_F \\ &\leq 2^{-k}\left|\left|\widetilde{\boldsymbol{Y}}_0\right|\right|_F \\ &\leq 2^{-k}||\boldsymbol{Q}\boldsymbol{V}^*||_F \\ &\leq 2^{-k}\sqrt{R},\end{aligned}$$

so long that $M \leq c\beta\kappa R(N+\mu_0^2 L)\log^2(LN)$. As in (Ahmed and Romberg, 2015) we observe that when we choose $\kappa \geq 0.5\log_2(8\gamma^2 R)$, the bound for the Frobenious norm of $\widetilde{\boldsymbol{Y}}_\kappa$ is bounded by $\left|\left|\widetilde{\boldsymbol{Y}}_\kappa\right|\right|_F \leq (2\sqrt{2}\gamma)^{-1}$.

To show that the second condition on the certificate is also satisfies, we apply Lemma 10. We begin with writing the quantity we wish to bound in terms of the past golfing scheme iterate

$$\begin{aligned}
||\mathcal{P}_{T^\perp}(\boldsymbol{Y}_\kappa)|| &\leq \sum_{k=1}^{\kappa}\left|\left|\mathcal{P}_{T^\perp}\left(\kappa\mathcal{A}_k^*\mathcal{A}_k\widetilde{\boldsymbol{Y}}_{k-1}\right)\right|\right| \\
&= \sum_{k=1}^{\kappa}\left|\left|\mathcal{P}_{T^\perp}\left(\kappa\mathcal{A}_k^*\mathcal{A}_k\widetilde{\boldsymbol{Y}}_{k-1}-\widetilde{\boldsymbol{Y}}_{k-1}\right)\right|\right| \\
&\leq \sum_{k=1}^{\kappa}\left|\left|\kappa\mathcal{A}_k^*\mathcal{A}_k\widetilde{\boldsymbol{Y}}_{k-1}-\widetilde{\boldsymbol{Y}}_{k-1}\right|\right| \\
&\leq \sum_{k=1}^{\kappa}\left|\left|\kappa\mathcal{A}_k^*\mathcal{A}_k\widetilde{\boldsymbol{Y}}_{k-1}-\widetilde{\boldsymbol{Y}}_{k-1}\right|\right|_F \\
&\leq \sum_{k=1}^{\kappa}\max_{k\in[1,\dots\kappa]}\left|\left|\kappa\mathcal{A}_k^*\mathcal{A}_k\widetilde{\boldsymbol{Y}}_{k-1}-\widetilde{\boldsymbol{Y}}_{k-1}\right|\right|_F \\
&\leq \sum_{k=1}^{\kappa}\frac{1}{2}2^{-k} \\
&\leq \frac{1}{2}
\end{aligned}$$

We use Lemma 10 to bound the maximum spectral norm of $\kappa\mathcal{A}_k^*\mathcal{A}_k\widetilde{\boldsymbol{Y}}_{k-1}-\widetilde{\boldsymbol{Y}}_{k-1}$ with probability $1-O((LN)^{1-\beta}$. Taking $\kappa \geq \log(LN)$ shows that the final certificate $\boldsymbol{Y}_\kappa$ satisfies all the desired properties, completing the proof.

A.2.1 MATRIX BERNSTEIN INEQUALITY AND OLICZ NORM

The lemmas required in our main result depend heavily on the matrix Bernstein inequality (Tropp, 2012). This inequality uses the variance measure and Oricz norm of a matrix to bound the largest singular value of the matrix. The matrix Bernstein inequality is summarized as

Theorem 6 (Matrix Bernstein's Inequality) *Let $\boldsymbol{X}_i \in \mathbb{R}^{L,N}$, $i \in [1,\dots,M]$ be M random matrices such that $\mathbb{E}[\boldsymbol{X}_i]=0$ and $||\boldsymbol{X}_i||_{\psi_\alpha} < U_\alpha < \infty$ for some $\alpha \geq 1$. Then with probability $1-e^{-t}$, the spectral norm of the sum is bounded by*

$$\left|\left|\sum_{i=1}^{M}\boldsymbol{X}_i\right|\right| \leq C\max\left\{\sigma_X\sqrt{t+\log(L+N)}, U_\alpha\log^{1/\alpha}\left(\frac{MU_\alpha^2}{\sigma_X^2}\right)(t+\log(L+N))\right\},$$

for some constant C and the variance parameter defined by

$$\sigma_X = \max\left\{\left|\left|\sum_{i=1}^{M}\mathbb{E}[\boldsymbol{X}_i\boldsymbol{X}_i^*]\right|\right|^{1/2}, \left|\left|\sum_{i=1}^{M}\mathbb{E}[\boldsymbol{X}_i^*\boldsymbol{X}_i]\right|\right|^{1/2}\right\}.$$

where Orlicz-α norm $||X||_{\psi_\alpha}$ is defined as

$$||X||_{\psi_\alpha} = \inf\left\{y>0|\mathbb{E}\left[e^{||X||^\alpha/y^\alpha}\right]\leq 2\right\}. \tag{28}$$

In particular we will use the matrix Bernstein inequality with the Orlicz-1 and Orlicz-2 norms, since subgaussian and subexponential random variables have bounded Orlicz-2 and -1 norms, respectively. To calculate these norms, we find the following lemmas from (Tropp, 2012; Ahmed and Romberg, 2015) useful:

Lemma 7 (Lemma 5.14, Tropp, 2012) *A random variable X is subgaussian iff X^2 is subexponential. Furthermore,*

$$||X||_{\psi_2}^2 \leq \left|\left|X^2\right|\right|_{\psi_1} \leq 2||X||_{\psi_2}^2.$$

Lemma 8 (Lemma 7, Ahmed and Romberg, 2015) *Let X_1 and X_2 be two subgaussian ranfom variables. Then the product X_1X_2 is a subexponential random variable with*

$$||X_1X_2||_{\psi_1} \leq c||X_1||_{\psi_2}||X_2||_{\psi_2}.$$

Lemma 7 relates the Orlicz-1 and -2 norms for a random variable and it's square. Lemma 8 allows us to factor an Orlicz-1 norm of a sub-exponential random variable as the product of two subgaussian random variables. Finally we find useful the following calculation for the Orlicz-1 norm of the norm of a random Gaussian vector $\boldsymbol{z}_n$ with *i.i.d.* zero-mean and variance σ^2 entries:

$$\begin{aligned}
\left|\left|||\boldsymbol{z}_n||_2^2\right|\right|_{\psi_1} &= \inf\left\{y:\mathbb{E}\left[e^{||\boldsymbol{z}_n||_2^2/y}\right]\leq 2\right\} \\
&= \inf\left\{y:\frac{1}{\sqrt{2\pi\sigma}}\int_{\mathbb{R}}e^{-z_n^2(1/2\sigma^2-1/y)}dz_n \leq 2^{\frac{1}{M}}\right\} \\
&= \frac{2\sigma^2}{1-4^{-\frac{1}{M}}}.
\end{aligned} \tag{29}$$

A.2.2 BOUND ON $||\kappa\mathcal{P}_T\mathcal{A}_k^*\mathcal{A}_k\mathcal{P}_T-\mathcal{P}_T||$

Lemma 9 *Let $\mathcal{P}_T$ be defined as in Equation (24) and $\mathcal{A}_k$ be the restricted measurement operator as defined in Equation (27). Then if the number of nodes scale as*

$$M \geq c\beta\kappa R\left(N+\mu_0^2L\right)\log^2(LN),$$

for a constant $\beta>1$, then with probability greater then $1-O(\kappa(LN)^{-\beta}$, we have

$$\max_{k\in[1,\dots,\kappa]}||\kappa\mathcal{P}_T\mathcal{A}^*\mathcal{A}\mathcal{P}_T-\mathcal{P}_T|| \leq \frac{1}{2}.$$

Proof

Lemma 9 bounds the operator norm

$$||\kappa\mathcal{P}_T\mathcal{A}_k^*\mathcal{A}_k\mathcal{P}_T-\mathcal{P}_T||.$$

Since $\mathbb{E}\left[\mathcal{A}_k^*\mathcal{A}_k\right] = \frac{1}{\kappa}\mathcal{I}$, this norm is equivalent to

$$\begin{aligned}\kappa\mathcal{P}_T\mathcal{A}_k^*\mathcal{A}_k\mathcal{P}_T - \mathcal{P}_T &= \kappa\mathcal{P}_T\mathcal{A}_k^*\mathcal{A}_k\mathcal{P}_T - \mathbb{E}\left[\kappa\mathcal{P}_T\mathcal{A}_k^*\mathcal{A}_k\mathcal{P}_T\right] \\ &= \kappa\sum_{n\in\Gamma_k}\left(\mathcal{P}_T(\boldsymbol{A}_n)\otimes\mathcal{P}_T(\boldsymbol{A}_n) - \mathbb{E}\left[\mathcal{P}_T(\boldsymbol{A}_n)\otimes\mathcal{P}_T(\boldsymbol{A}_n)\right]\right).\end{aligned}$$

We can also define here $\mathcal{L}_n(\boldsymbol{C}) = \langle\mathcal{P}_T(\boldsymbol{A}_n), \boldsymbol{C}\rangle\mathcal{P}_T(\boldsymbol{A}_n)$ which has $||\mathcal{L}_n|| = ||\mathcal{P}_T(\boldsymbol{A}_n)||_F^2$ which gives us

$$\kappa\mathcal{P}_T\mathcal{A}_k^*\mathcal{A}_k\mathcal{P}_T - \mathbb{E}\left[\kappa\mathcal{P}_T\mathcal{A}_k^*\mathcal{A}_k\mathcal{P}_T\right] = \kappa\sum_{n\in\Gamma_k}\left(\mathcal{L}_n - \mathbb{E}\left[\mathcal{L}_n\right]\right).$$

To calculate the variance, we can use the symmetry of $\mathcal{L}_n$ to only calculate

$$\kappa^2\left|\left|\sum_{n\in\Gamma_k}\mathbb{E}\left[\mathcal{L}_n^2\right] - \mathbb{E}\left[\mathcal{L}_n\right]^2\right|\right| \leq \kappa^2\left|\left|\sum_{n\in\Gamma_k}\mathbb{E}\left[\mathcal{L}_n^2\right]\right|\right| = \kappa^2\left|\left|\mathbb{E}\left[\sum_{n\in\Gamma_k}\|\mathcal{P}_T(\boldsymbol{A}_n)\|_F^2\mathcal{L}_n\right]\right|\right|.$$

We now need to bound $\|\mathcal{P}_T(\boldsymbol{A}_n)\|_F^2$, which can be done by the following:

$$\begin{aligned}||\mathcal{P}_T(\boldsymbol{A}_n)||_F^2 &= \langle\mathcal{P}_T(\boldsymbol{A}_n), \boldsymbol{A}_n\rangle \\ &= \langle\boldsymbol{Q}\boldsymbol{Q}^*\boldsymbol{z}_n\boldsymbol{f}_n^*, \boldsymbol{z}_n\boldsymbol{f}_n^*\rangle + \langle\boldsymbol{z}_n\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*, \boldsymbol{z}_n\boldsymbol{f}_n^*\rangle - \langle\boldsymbol{Q}\boldsymbol{Q}^*\boldsymbol{z}_n\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*, \boldsymbol{z}_n\boldsymbol{f}_n^*\rangle \\ &= \|\boldsymbol{f}_n\|_2^2\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2 + \|\boldsymbol{z}_n\|_2^2\|\boldsymbol{V}^*\boldsymbol{f}_n\|_2^2 - \|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2\|\boldsymbol{V}^*\boldsymbol{f}_n\|_2^2 \\ &\leq N\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2 + \|\boldsymbol{z}_n\|_2^2\|\boldsymbol{V}^*\boldsymbol{f}_n\|_2^2.\end{aligned}$$

Using this calculation we can write

$$\begin{aligned}\left|\left|\mathbb{E}\left[\sum_{n\in\Gamma_k}||\mathcal{P}_T\left(\boldsymbol{A}_n\right)||_F^2\mathcal{L}_n\right]\right|\right| &\leq \left|\left|\sum_{n\in\Gamma_k}\mathbb{E}\left[\left(N\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2 + \|\boldsymbol{z}_n\|_2^2\|\boldsymbol{V}^*\boldsymbol{f}_n\|_2^2\right)\mathcal{L}_n\right]\right|\right| \\ &\leq N\left|\left|\sum_{n\in\Gamma_k}\mathbb{E}\left[\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2\mathcal{L}_n\right]\right|\right| \\ &\quad + \sup\|\boldsymbol{V}^*\boldsymbol{f}_n\|_\infty\left|\left|\sum_{n\in\Gamma_k}\mathbb{E}\left[\|\boldsymbol{z}_n\|_2^2\mathcal{L}_n\right]\right|\right| \\ &\leq N\left|\left|\sum_{n\in\Gamma_k}\mathbb{E}\left[\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2\mathcal{L}_n\right]\right|\right| + R\mu_0^2\left|\left|\sum_{n\in\Gamma_k}\mathbb{E}\left[\|\boldsymbol{z}_n\|_2^2\mathcal{L}_n\right]\right|\right|\end{aligned}$$

We now need to bound these two quantities. First we look to bound the first quantity

$$\begin{aligned}\left|\left|\sum_{n\in\Gamma_k}\mathbb{E}\left[\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2(\mathcal{P}_T(\boldsymbol{A}_n)\otimes\mathcal{P}_T(\boldsymbol{A}_n))\right]\right|\right| &\leq \|\mathcal{P}_T\|\left|\left|\mathbb{E}\left[\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2(\boldsymbol{A}_n\otimes\boldsymbol{A}_n)\right]\right|\right|\|\mathcal{P}_T\| \\ &\leq \left|\left|\mathbb{E}\left[\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2\{z_n[\alpha]z_n^*[\beta]\boldsymbol{f}_n\boldsymbol{f}_n^*\}_{\alpha,\beta}\right]\right|\right|.\end{aligned}$$

Expanding, we have:

$$\begin{aligned}* &= \left|\left|\mathbb{E}\left[\left(\sum_{l=1}^{L}\|\boldsymbol{q}_l\|_2^2|z_n[l]|^2 + 2\sum_{l\neq m}\mathrm{Re}(\langle\boldsymbol{q}_l, \boldsymbol{q}_m\rangle z_n[l]z_n^*[m])\right)z_n[\alpha]z_n^*[\beta]\boldsymbol{I}_N\right]_{\alpha,\beta}\right|\right| \\ &= \left|\left|\left\{\frac{1}{M^2}\sum_{l=1}^{L}\|\boldsymbol{q}_l\|_2^2\boldsymbol{I}_N\delta_{\alpha=\beta} + \frac{2}{M^2}\langle\boldsymbol{q}_\alpha, \boldsymbol{q}_\beta\rangle\boldsymbol{I}_N\delta_{\alpha\neq\beta}\right\}_{\alpha,\beta}\right|\right| \\ &= \frac{1}{M^2}\left|\left|\left\{\|\boldsymbol{Q}\|_F^2\boldsymbol{I}_N\delta_{\alpha=\beta} + 2\langle\boldsymbol{q}_\alpha, \boldsymbol{q}_\beta\rangle\boldsymbol{I}_N\delta_{\alpha\neq\beta}\right\}_{\alpha,\beta}\right|\right|,\end{aligned}$$

giving us

$$\left|\left|\sum_{n\in\Gamma_k}\mathbb{E}\left[\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2(\mathcal{P}_T(\boldsymbol{A}_n)\otimes\mathcal{P}_T(\boldsymbol{A}_n))\right]\right|\right| \leq \frac{1}{M\kappa}\|\boldsymbol{Q}\|_F^2 + \|\boldsymbol{Q}\boldsymbol{Q}^*\| \leq \frac{R+1}{M\kappa}.$$

Similarly, for the second term we can take $\boldsymbol{Q} = \boldsymbol{I}_L$ to get

$$\left|\left|\sum_{n\in\Gamma_k}\mathbb{E}\left[\|\boldsymbol{z}_n\|_2^2(\mathcal{P}_T(\boldsymbol{A}_n)\otimes\mathcal{P}_T(\boldsymbol{A}_n))\right]\right|\right| \leq \frac{1}{M\kappa}\|\boldsymbol{I}\|_F^2 = \frac{L}{M\kappa}.$$

Putting the pieces together, we get

$$\sigma_X^2 = \kappa R\frac{N + \mu_0^2 L}{M}.$$

To use the matrix Bernstein inequality, it now remains to bound the following Orlicz-1 norm: $\kappa\,||\mathcal{L}_n - \mathbb{E}\left[\mathcal{L}_n\right]||_{\psi_1}$. By the PSD quality of $\mathcal{L}_n$ and its expectation,

$$||\mathcal{L}_n - \mathbb{E}\left[\mathcal{L}_n\right]||_{\psi_1} \leq \max\left\{||\mathcal{L}_n||_{\psi_1} - ||\mathbb{E}\left[\mathcal{L}_n\right]||_{\psi_1}\right\}.$$

The norm of $||\mathbb{E}\left[\mathcal{L}_n\right]||$ can be calculated via

$$||\mathbb{E}\left[\mathcal{L}_n\right]|| = ||\mathbb{E}\left[\mathcal{P}_T\left(\boldsymbol{A}_n\right)\right]||_F^2 = \mathbb{E}\left[\sum_m|z_n[m]|^2|f_n[m]|^2\right] = \frac{1}{M}\|\boldsymbol{f}_n\|_2^2 = \frac{N}{M},$$

indicating that the second term is simply $||\mathbb{E}\left[\mathcal{L}_n\right]||_{\psi_1} = N/(M\log(2))$. To calculate $||\mathbb{E}\left[\mathcal{L}_n\right]||_{\psi_1}$, we use the definition of the Orlitcz-1 norm in Equation (28) to see that

$$||\mathcal{L}_n||_{\psi_1} = \left|\left|||\mathcal{P}_T\left(\boldsymbol{A}_n\right)||_2^2\right|\right|_{\psi_1} \leq N\left|\left|\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2\right|\right|_{\psi_1} + RN\mu_0^2\left|\left|\|\boldsymbol{z}_n\|_2^2\right|\right|_{\psi_1},$$

where the inequality stems form the fact that $\|\boldsymbol{V}^*\boldsymbol{f}_n\|_2^2 \leq RN\mu_0^2$ and $\|\boldsymbol{f}_n\|_2^2 \leq N$. Using the result in Equation (29) with $\sigma^2 = 1/M$ in the first term and in $\sigma^2 = R/M$ in the second term yields

$$\begin{aligned}||\mathbb{E}\left[\mathcal{L}_n\right]||_{\psi_1} &\leq N\left|\left|\|\boldsymbol{Q}^*\boldsymbol{z}_n\|_2^2\right|\right|_{\psi_1} + RN\mu_0^2\left|\left|\|\boldsymbol{z}_n\|_2^2\right|\right|_{\psi_1} \\ &\leq \frac{1}{M}\left(\frac{N}{M(1-4^{-\frac{1}{R}})} + \frac{R\mu_0^2}{1-4^{-\frac{1}{L}}}\right) \\ &\leq \frac{2R\left(N + L\mu_0^2\right)}{\log(2)M}.\end{aligned}$$

We now have appropriate bounds on both the variance and Orliscz norm, permitting a bound on the largest singular value via the Matrix Bernstein inequality. Specifically, we can see that the first term in Theorem 6 with $t = \beta \log(LN) > \log(N+L)$ is bounded as

$$\sigma_X \sqrt{t + \log(L+N)} \leq \sqrt{2\kappa R\beta \frac{N+\mu_0^2 L}{M} \log(LN)}.$$

Likewise we can bound the second term:

$$\begin{aligned} U_1 \log\left(\frac{MU_1^2}{\sigma_X^2}\right)(t + \log(L+N)) &\leq 2\beta U_1 \log\left(\frac{4\Delta\kappa R(N+\mu_0^2 L)}{\log^2(2)M}\right)\log(LN) \\ &\leq c\frac{\beta\kappa R(N+\mu_0^2 L)}{M}\log^2(LN). \end{aligned}$$

Thus to appropriately bound

$$||\kappa \mathcal{P}_T \mathcal{A}_k^* \mathcal{A}_k \mathcal{P}_T - \mathcal{P}_T|| \leq c\max\left\{\sqrt{\frac{\kappa R\beta(N+\mu_0^2 L)}{M}\log(LN)}, \frac{\beta\kappa R(N+\mu_0^2 L)}{M}\log^2(LN)\right\},$$

we can see that we would need

$$M \geq C\beta\kappa R(N+\mu_0^2 L)\log^2(LN).$$

Taking the union bound over the κ partitions completes the proof of the lemma. ■

A.2.3 BOUND ON $\|(\mathcal{A}^*\mathcal{A} - \mathcal{I})(\boldsymbol{G})\|$

Lemma 10 *Let $\mathcal{A}_k$ be defined as in Equation (27), $\kappa < M$ be the number of steps in the golfing scheme and assume that $M \leq LN$. Then as long as*

$$M \geq c\beta\kappa \max\left(N + \mu_0^2 L\right)\log^2(NL),$$

where μ_k^2 is the coherence term defined by

$$\mu_k^2 = R^{-1} \sup_{\omega\in[0,2\pi]} \left|\left|\widetilde{\boldsymbol{Y}}_k^* \boldsymbol{f}_\omega\right|\right|_2^2, \tag{30}$$

then with probability at least $1 - O(M(LN)^{-\beta})$, we have

$$\max_k \left|\left|\kappa \mathcal{A}_k^* \mathcal{A}_k(\widetilde{\boldsymbol{Y}}_{k-1}) - \widetilde{\boldsymbol{Y}}_{k-1}\right|\right| \leq 2^{-(k+1)}.$$

Proof

Lemma 10 essentially bounds the operator norm of $\kappa\mathcal{A}^*\mathcal{A} - \mathcal{I}$. In particular, to prove Theorem 2, the reduced version with $\kappa = 1$ is needed. Lemma 2 essentially uses the matrix Bernstein inequality to accomplish this task, taking

$$X_n = \kappa(\langle \boldsymbol{G}, \boldsymbol{A}_n\rangle \boldsymbol{A}_n - \mathbb{E}\left[\langle \boldsymbol{G}, \boldsymbol{A}_n\rangle \boldsymbol{A}_n\right]),$$

and we just need to control $||\sum \mathbb{E}\left[X_n X_n^*\right]||$ and $||\sum \mathbb{E}\left[X_n^* X_n\right]||$. To bound the second of these, we can calculate

$$\begin{aligned} \left|\left|\sum_{n\in\Gamma_k} \mathbb{E}\left[X_n^* X_n\right]\right|\right| &\leq \kappa^2 \left|\left|\sum_{n\in\Gamma_k} \mathbb{E}\left[|\langle \boldsymbol{G}, \boldsymbol{A}_n\rangle|^2 \boldsymbol{A}_n \boldsymbol{A}_n^*\right]\right|\right| \\ &= \kappa^2 \left|\left|\sum_{n\in\Gamma_k} \mathbb{E}\left[||\boldsymbol{f}_n||_2^2 |\langle \boldsymbol{G}, \boldsymbol{A}_n\rangle|^2 \boldsymbol{z}_n \boldsymbol{z}_n^*\right]\right|\right| \\ &\leq \frac{3N\kappa}{M}||\boldsymbol{G}||_F^2, \end{aligned}$$

where the second inequality is due to Lemma 11 and $||\boldsymbol{f}_n||_2^2 \leq N$. For the other expectation

$$\begin{aligned} \left|\left|\sum_{n\in\Gamma_k} \mathbb{E}\left[X_n X_n^*\right]\right|\right| &\leq \kappa^2 \left|\left|\sum_{n\in\Gamma_k} \mathbb{E}\left[|\langle \boldsymbol{G}, \boldsymbol{A}_n\rangle|^2 \boldsymbol{A}_n^* \boldsymbol{A}_n\right]\right|\right| \\ &= \frac{L\kappa^2}{M^2}\left|\left|\sum_{n\in\Gamma_k} \mathbb{E}\left[\|\boldsymbol{G}\boldsymbol{f}_n\|_2^2 \boldsymbol{f}_n \boldsymbol{f}_n^*\right]\right|\right| \\ &\leq \frac{L\kappa^2}{M^2}\sup_\omega(\|\boldsymbol{G}\boldsymbol{f}_\omega\|_2^2)\left|\left|\sum_{n\in\Gamma_k} \mathbf{1}_N\right|\right| \\ &\leq \frac{L\kappa}{M}\sup_\omega(\|\boldsymbol{G}\boldsymbol{f}_\omega\|_2^2) \\ &= \frac{L\kappa}{M}\mu^2\|\boldsymbol{G}\|_F^2. \end{aligned}$$

Using these bounds, we can write

$$\sigma_X^2 \leq \frac{\kappa}{M}\|\boldsymbol{G}\|_F^2 \max\left\{\mu_0 L, 3N\right\},$$

and to use Proposition 1 we just need to bound $\|X\|_{\psi_2}$. To start, we can see that

$$\begin{aligned} U_1 &= ||X||_{\psi_1} \leq 2\kappa\,||\langle \boldsymbol{G}, \boldsymbol{A}_n\rangle \boldsymbol{A}_n||_{\psi_1} \\ &\leq c\kappa\,||\langle \boldsymbol{G}, \boldsymbol{A}_n\rangle||_{\psi_2}\,||||\boldsymbol{A}_n||_F||_{\psi_2} \\ &\leq c\kappa\,||\langle \boldsymbol{G}, \boldsymbol{A}_n\rangle||_{\psi_2}\sqrt{\left|\left|||\boldsymbol{f}_n||_2^2 ||\boldsymbol{z}_n||_2^2\right|\right|_{\psi_2}} \\ &= c\kappa\sqrt{\frac{N}{M\left(1-4^{-1/L}\right)}}\,|||\text{trace}(\boldsymbol{f}_n \boldsymbol{z}_n^* \boldsymbol{G})||_{\psi_2} \\ &\leq c\kappa\sqrt{\frac{N}{M^2\left(1-4^{-1/L}\right)}}\sum_{l=1}^L ||\langle \boldsymbol{g}_l, \boldsymbol{f}_n\rangle||_{\psi_2} \\ &\leq c\kappa\sqrt{\frac{N^2 L\mu_0^2\|\boldsymbol{G}\|_F^2}{M^2}}. \end{aligned}$$

We can now apply the matrix Bernstein theorem with the calculated values of U_1 and σ_X. Again using $t = \beta \log(LN)$, the first portion of the bound is

$$\sigma_X \sqrt{t + \log(L+N)} \leq c \left|\left|\boldsymbol{G}\right|\right|_F \sqrt{\frac{\kappa\beta}{M} \max\{\mu_k^2 L, N\} \log(LN)},$$

and the second portion of the bound is

$$\begin{aligned} U_1 \log\left(\frac{\Delta U_1^2}{\sigma_X}\right)&(t + \log(L+N)) \\ &\leq c \left|\left|\boldsymbol{G}\right|\right|_F \kappa \sqrt{\frac{LN\mu_k^2}{M}} \log\left(\frac{LN\mu_k^2}{M^2} \frac{Mc \left|\left|\boldsymbol{G}\right|\right|_F^2 \kappa^2}{\kappa \left|\left|\boldsymbol{G}\right|\right|_F^2 \max\{\mu_k^2 L, N\}}\right) \beta \log(LN) \\ &\leq c \left|\left|\boldsymbol{G}\right|\right|_F \kappa \sqrt{\frac{LN\mu_k^2}{M}} \log\left(\frac{c\Delta\kappa LN\mu_k^2}{M \max\{\mu_k^2 L, N\}}\right) \beta \log(LN) \\ &\leq c\beta \left|\left|\boldsymbol{G}\right|\right|_F \kappa \sqrt{\frac{LN\mu_k^2}{M}} \log\left(\min\{\mu_k^2 L, N\}\right) \log(LN). \end{aligned}$$

We can now use Lemma 12 to bound $\mu_k^2 \leq \mu_0^2$ with probability $1 - O(M(LN)^{-\beta})$ and Lemma 9 to bound $\left|\left|\boldsymbol{G}_k\right|\right|_F \leq 2^{-k}\sqrt{R}$, which gives us a bound of

$$\begin{aligned} \left|\left|(\mathcal{A}^*\mathcal{A} - \mathcal{I})\boldsymbol{G}\right|\right| \leq c 2^{-k/2} \max\Bigg\{ &\sqrt{\frac{\kappa\beta R \log(LN)}{M} \log(\max\{\mu_0^2 L, N\})}, \\ &\sqrt{\mu_0^2 LN} \frac{\beta\kappa}{M} \log(LN) \log(\min\{\mu_0^2 L, N\}) \Bigg\}. \end{aligned}$$

Simplifying the bound using $R \leq \min\{L, N\}$,

$$\left|\left|(\mathcal{A}^*\mathcal{A} - \mathcal{I})\boldsymbol{G}\right|\right| \leq c 2^{-k/2} \max\left\{ \sqrt{\frac{\kappa\beta R \log(LN)}{M} \log(LN)}, \sqrt{\mu_0^2 LN} \frac{\beta\kappa}{M} \log^2(LN)) \right\}$$

Taking

$$M \geq c\beta\kappa R \max\{N, L\mu_0^2\} \log^2(LN),$$

proves the lemma. To simplify the bound on the probability, we note that Lemma 12 holds with probability $1 - O(M(LN)^{-\beta})$ and this lemma holds with probability $1 - O(\kappa(LN)^{-\beta})$. Since $\kappa < M$ and assuming that $M \leq LN$, we can write that the result holds with probability $1 - O((LN)^{1-\beta})$. Additionally, since Lemma 12 holds when

$$M \geq c\beta\kappa R \left(N + L\mu_0^2\right) \log^2(LN) \geq c\beta\kappa R \max\{N, L\mu_0^2\} \log^2(LN),$$

Then both lemmas hold under the same condition. ■

A.2.4 BOUND ON $\mathbb{E}\left[|\langle \boldsymbol{C}, \boldsymbol{A}_n\rangle|^2 \boldsymbol{z}_n \boldsymbol{z}_n^*\right]$

Lemma 11 bounds the spectrum of the expected matrix

$$\mathbb{E}\left[|\langle \boldsymbol{G}, \boldsymbol{A}_n\rangle|^2 \boldsymbol{z}_n \boldsymbol{z}_n^*\right]:$$

Lemma 11 *Suppose $\boldsymbol{A}_n = \boldsymbol{z}_n \boldsymbol{f}_n^*$ be defined as the outer product of an* i.i.d. *random Gaussian vector $\boldsymbol{z}_n$ with zero mean and variance $1/M$ and a random Fourier vector $\boldsymbol{f}_n$. Then the operator $|\langle \boldsymbol{C}, \boldsymbol{A}_n\rangle|^2 \boldsymbol{z}_n \boldsymbol{z}_n^*$ satisfies*

$$\mathbb{E}_z\left[|\langle \boldsymbol{C}, \boldsymbol{A}_n\rangle|^2 \boldsymbol{z}_n \boldsymbol{z}_n^*\right] \preceq \frac{3}{M^2} \|\boldsymbol{C}^* \boldsymbol{f}_n\|_2^2 \boldsymbol{I}_M,$$

and

$$\mathbb{E}_{z,f}\left[|\langle \boldsymbol{C}, \boldsymbol{A}_n\rangle|^2 \boldsymbol{z}_n \boldsymbol{z}_n^*\right] \preceq \frac{3}{M^2} \|\boldsymbol{C}\|_F^2 \boldsymbol{I}_M.$$

Proof

To begin the proof, we look at the expectation of each element of the matrix. We first calculate the expectation with respect to $\boldsymbol{z}_n$,

$$\begin{aligned} * &= \mathbb{E}_z\left[\left|\sum_{l=1}^L z_n[l] \boldsymbol{c}_l^* \boldsymbol{f}_n\right|^2 z_n[\alpha] z_n^*[\beta]\right] \\ &= \mathbb{E}_z\left[\left(\sum_{l=1}^L z_n[l] \boldsymbol{c}_l^* \boldsymbol{f}_n\right)^* \left(\sum_{l=1}^L z_n[l] \boldsymbol{c}_l^* \boldsymbol{f}_n\right) z_n[\alpha] z_n^*[\beta]\right] \\ &= \mathbb{E}_z\left[\sum_{l=1}^L |z_n[l]|^2 |\boldsymbol{c}_l^* \boldsymbol{f}_n|^2 z_n[\alpha] z_n^*[\beta] + 2\sum_{k \neq l} \mathrm{Re}\left(z_n^*[l] z_n[k] \langle \boldsymbol{c}_l^* \boldsymbol{f}_n, \boldsymbol{c}_k^* \boldsymbol{f}_n\rangle\right) z_n[\alpha] z_n^*[\beta]\right] \\ &= \left(\frac{3}{2M^2} |\boldsymbol{c}_\alpha^* \boldsymbol{f}_n|^2 + \frac{1}{M^2} \|\boldsymbol{C}\boldsymbol{f}_n\|_2^2\right) \delta_{\alpha=\beta} + \frac{2}{M^2} \langle \boldsymbol{c}_\alpha^* \boldsymbol{f}_n, \boldsymbol{c}_\beta^* \boldsymbol{f}_n\rangle \delta_{\alpha\neq\beta}. \end{aligned}$$

We can then use the matrix formulation

$$\begin{aligned} \mathbb{E}_z\left[|\langle \boldsymbol{C}, \boldsymbol{A}_n\rangle|^2 \boldsymbol{z}_n \boldsymbol{z}_n^*\right] &= \frac{3}{2M^2} \mathrm{diag}(\boldsymbol{C}\boldsymbol{f}_n \boldsymbol{f}_n^* \boldsymbol{C}^*) + \frac{1}{M^2} \|\boldsymbol{C}\boldsymbol{f}_n\|_2^2 \boldsymbol{I}_M \\ &\quad + \frac{2}{M^2} \boldsymbol{C}\boldsymbol{f}_n \boldsymbol{f}_n^* \boldsymbol{C}^* + \frac{2}{M^2} \mathrm{diag}(\boldsymbol{C}\boldsymbol{f}_n \boldsymbol{f}_n^* \boldsymbol{C}^*) \\ &= \frac{1}{M^2} \|\boldsymbol{C}\boldsymbol{f}_n\|_2^2 \boldsymbol{I}_M + 2\boldsymbol{C}\boldsymbol{f}_n \boldsymbol{f}_n^* \boldsymbol{C}^* - \frac{1}{2} \mathrm{diag}(\boldsymbol{C}\boldsymbol{f}_n \boldsymbol{f}_n^* \boldsymbol{C}^*) \\ &\preceq \frac{3}{M^2} \|\boldsymbol{C}\boldsymbol{f}_n\|_2^2 \boldsymbol{I}_M, \end{aligned}$$

where to obtain the result we first use the linearity of the expectation along with the with the positive-semidefinite property of $\mathrm{diag}(\boldsymbol{C}\boldsymbol{f}_n \boldsymbol{f}_n^* \boldsymbol{C}^*)$, proving the fist portion of the Lemma. To prove the second portion we simply take an expectation with respect to $\boldsymbol{f}_n$:

$$\mathbb{E}_{z,f}\left[|\langle \boldsymbol{C}, \boldsymbol{A}_n\rangle|^2 \boldsymbol{z}_n \boldsymbol{z}_n^*\right] \preceq \mathbb{E}_f\left[\frac{3}{M^2} \|\boldsymbol{C}\boldsymbol{f}_n\|_2^2 \boldsymbol{I}_M\right] \preceq \frac{3}{M^2} \|\boldsymbol{C}\|_F^2 \boldsymbol{I}_{,M},$$

completing the proof. ■

A.2.5 CONTRACTIVE PROPERTY OF μ_k^2

Lemma 12 *Let μ_k^2 be the coherence factor as defined in Equation* (30)*, and additionally assume that $L > 1$ and that $LN > R\mu_0^4$. If*

$$M \geq c\beta\kappa R\left(N + L\mu_0^2\right)\log^2(LN),$$

then with probability at least $1 - O(\kappa(LN)^{-\beta})$,

$$\mu_k^2 \leq 2^{-1}\mu_{k-1}^2,$$

for all $k \in [1, \cdots, \kappa]$.

Proof

In Lemma 12 we show that the coherence term reduces at each golfing iteration. Observe that

$$\mu_k^2 = \frac{1}{R}\sup_{\omega}\sum_{l=1}^{L}\langle\widetilde{\boldsymbol{Y}}_k, \boldsymbol{e}_l\boldsymbol{f}^*\rangle^2$$

$$\begin{aligned}
&= \frac{1}{R}\sup_{\omega}\sum_{l=1}^{L}\left(\sum_{n\in\Gamma_k}\kappa\langle\mathcal{P}_T\left(\boldsymbol{A}_n\right), \boldsymbol{e}_l\boldsymbol{f}^*\rangle\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle - \langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{e}_l\boldsymbol{f}^*\rangle\right)^2 \\
&= \frac{1}{R}\sup_{\omega}\sum_{l=1}^{L}\left(\sum_{n\in\Gamma_k}\kappa\langle\mathcal{P}_T\left(\boldsymbol{A}_n\right), \boldsymbol{e}_l\boldsymbol{f}^*\rangle\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle - \mathbb{E}\left[\kappa\langle\mathcal{P}_T\left(\boldsymbol{A}_n\right), \boldsymbol{e}_l\boldsymbol{f}^*\rangle\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle\right]\right)^2.
\end{aligned}$$

To bound this quantity we use the scalar Bernstein inequality on each of the inner quantities

$$\sum_{n\in\Gamma_k}X_n = \sum_{n\in\Gamma_k}\kappa\langle\mathcal{P}_T\left(\boldsymbol{A}_n\right), \boldsymbol{e}_l\boldsymbol{f}^*\rangle\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle - \mathbb{E}\left[\kappa\langle\mathcal{P}_T\left(\boldsymbol{A}_n\right), \boldsymbol{e}_l\boldsymbol{f}^*\rangle\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle\right].$$

As in the matrix Bernstein formulation, we require both the variance and Orlicz norm. First we find the variance,

$$\begin{aligned}
\sum_{n\in\Gamma_k}\mathbb{E}\left[X_nX_n^*\right] &= \kappa^2\sum_{n\in\Gamma_k}\mathbb{E}\left[|\langle\mathcal{P}_T\left(\boldsymbol{A}_n\right), \boldsymbol{e}_l\boldsymbol{f}^*\rangle|^2|\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle|^2\right] \\
&\qquad -|\mathbb{E}\left[\langle\mathcal{P}_T\left(\boldsymbol{A}_n\right), \boldsymbol{e}_l\boldsymbol{f}^*\rangle\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle\right]|^2 \\
&\leq \kappa^2\sum_{n\in\Gamma_k}\mathbb{E}\left[|\langle\boldsymbol{Q}\boldsymbol{Q}^*\boldsymbol{z}_n\boldsymbol{f}_n^*, \boldsymbol{e}_l\boldsymbol{f}^*\rangle + \langle\boldsymbol{z}_n\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}, \boldsymbol{e}_l\boldsymbol{f}^*\rangle\right. \\
&\qquad \left.+\langle\boldsymbol{Q}\boldsymbol{Q}^*\boldsymbol{z}_n\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*, \boldsymbol{e}_l\boldsymbol{f}^*\rangle|^2|\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle|^2\right] \\
&\leq \kappa^2\sum_{n\in\Gamma_k}\mathbb{E}\left[\left(|\langle\boldsymbol{Q}\boldsymbol{Q}^*\boldsymbol{z}_n\boldsymbol{f}_n^*, \boldsymbol{e}_l\boldsymbol{f}^*\rangle|^2 + |\langle\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*, \boldsymbol{f}^*\rangle z_n[l]|^2\right.\right. \\
&\qquad \left.\left.+|\langle\boldsymbol{Q}\boldsymbol{Q}^*\boldsymbol{z}_n\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*, \boldsymbol{e}_l\boldsymbol{f}^*\rangle|^2\right)|\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle|^2\right].
\end{aligned}$$

This sum consists of three terms. The first of which can be bounded using Lemma 11,

$$\begin{aligned}
&\sum_{n\in\Gamma_k}\mathbb{E}\left[|\langle\boldsymbol{Q}\boldsymbol{Q}^*\boldsymbol{z}_n\boldsymbol{f}_n^*, \boldsymbol{e}_l\boldsymbol{f}^*\rangle|^2|\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle|^2\right] \\
&\qquad = \sum_{n\in\Gamma_k}\mathbb{E}\left[|\boldsymbol{f}_n^*\boldsymbol{f}\langle\boldsymbol{q}_l, \boldsymbol{Q}^*\boldsymbol{z}_n\rangle|^2|\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle|^2\right] \\
&\qquad \leq 3\sum_{n\in\Gamma_k}\mathbb{E}\left[\boldsymbol{f}^*\boldsymbol{f}_n\boldsymbol{f}_n^*\boldsymbol{f}\boldsymbol{q}_l^*\boldsymbol{Q}^*\left|\left|\widetilde{\boldsymbol{Y}}_{k-1}\boldsymbol{f}_n\right|\right|_2^2\boldsymbol{I}_L\boldsymbol{Q}\boldsymbol{q}_l\right] \\
&\qquad \leq \frac{3R\mu_{k-1}^2}{M^3}\left|\left|\boldsymbol{q}_l\right|\right|_2^2\sum_{n\in\Gamma_k}\boldsymbol{f}^*\mathbb{E}\left[\boldsymbol{f}_n\boldsymbol{f}_n^*\right]\boldsymbol{f} \\
&\qquad = \frac{3NR\mu_{k-1}^2}{\kappa M^2}\left|\left|\boldsymbol{q}_l\right|\right|_2^2.
\end{aligned}$$

For the second term we have

$$\begin{aligned}
&\sum_{n\in\Gamma_k}\mathbb{E}\left[|\langle\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*, \boldsymbol{f}^*\rangle|^2|z_n[l]|^2|\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle|^2\right] \\
&\qquad = \sum_{n\in\Gamma_k}\mathbb{E}\left[|\langle\boldsymbol{V}^*\boldsymbol{f}_n, \boldsymbol{V}^*\boldsymbol{f}\rangle|^2|z_n[l]|^2|\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle|^2\right] \\
&\qquad = \sum_{n\in\Gamma_k}\mathbb{E}\left[\boldsymbol{f}^*\boldsymbol{V}\boldsymbol{V}^*\boldsymbol{f}_n\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*\boldsymbol{f}|\langle\widetilde{\boldsymbol{Y}}_{k-1}, z_n[l]\boldsymbol{z}_n\boldsymbol{f}_n^*\rangle|^2\right].
\end{aligned}$$

Using the fact that $|z_n[l]|^2 = \boldsymbol{e}_l^*\boldsymbol{z}_n\boldsymbol{z}_n^*\boldsymbol{e}_l$ and Lemma 11, we obtain

$$\begin{aligned}
&\sum_{n\in\Gamma_k}\mathbb{E}\left[|\langle\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*, \boldsymbol{f}^*\rangle|^2|z_n[l]|^2|\langle\widetilde{\boldsymbol{Y}}_{k-1}, \boldsymbol{A}_n\rangle|^2\right] \\
&\qquad \leq \frac{3}{M^2}\sum_{n\in\Gamma_k}\mathbb{E}\left[\boldsymbol{f}^*\boldsymbol{V}\boldsymbol{V}^*\boldsymbol{f}_n\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*\boldsymbol{f}\boldsymbol{e}_l^*\left|\left|\widetilde{\boldsymbol{Y}}_{k-1}\boldsymbol{f}_n\right|\right|_2^2\boldsymbol{I}_L\boldsymbol{e}_l\right] \\
&\qquad \leq \frac{3R\mu_{k-1}^2}{M^2}\sum_{n\in\Gamma_k}\mathbb{E}\left[\boldsymbol{f}^*\boldsymbol{V}\boldsymbol{V}^*\boldsymbol{f}_n\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*\boldsymbol{f}\right] \\
&\qquad \leq \frac{3R\mu_{k-1}^2}{M^2}|\Gamma_k|\left|\left|\boldsymbol{V}^*\boldsymbol{f}\right|\right|_2^2 \\
&\qquad \leq \frac{3R^2\mu_{k-1}^2\mu_0^2}{\kappa M^1}.
\end{aligned}$$

Finally, for the third term, we have

$$\begin{aligned}
&\sum_{n\in\Gamma_k}\mathbb{E}\left[|\langle \boldsymbol{Q}\boldsymbol{Q}^*\boldsymbol{z}_n\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*, \boldsymbol{e}_l\boldsymbol{f}^*\rangle|^2|\langle\widetilde{\boldsymbol{Y}}_{k-1},\boldsymbol{A}_n\rangle|^2\right]\\
&\quad=\sum_{n\in\Gamma_k}\mathbb{E}\left[|\langle \boldsymbol{V}^*\boldsymbol{f}_n,\boldsymbol{V}^*\boldsymbol{f}\rangle|^2|\langle \boldsymbol{q}_l,\boldsymbol{Q}^*\boldsymbol{z}_n\rangle|^2|\langle\widetilde{\boldsymbol{Y}}_{k-1},\boldsymbol{A}_n\rangle|^2\right]\\
&\quad=\sum_{n\in\Gamma_k}\mathbb{E}\left[\boldsymbol{f}^*\boldsymbol{V}\boldsymbol{V}^*\boldsymbol{f}_n\boldsymbol{f}_n^*\boldsymbol{V}\boldsymbol{V}^*\boldsymbol{f}|\langle \boldsymbol{q}_l,\boldsymbol{Q}^*\boldsymbol{z}_n\rangle|^2|\langle\widetilde{\boldsymbol{Y}}_{k-1},\boldsymbol{A}_n\rangle|^2\right]\\
&\quad\le\sum_{n\in\Gamma_k}\mathbb{E}\left[||\boldsymbol{V}^*\boldsymbol{f}||_2^2\,\boldsymbol{q}_l^*\boldsymbol{Q}^*\boldsymbol{z}_n\boldsymbol{z}_n^*|\langle\widetilde{\boldsymbol{Y}}_{k-1},\boldsymbol{A}_n\rangle|^2\boldsymbol{Q}\boldsymbol{q}_l\right]\\
&\quad\le\frac{3}{M^2}\,||\boldsymbol{q}_l||_2^2\,||\boldsymbol{V}^*\boldsymbol{f}||_2^2\sum_{n\in\Gamma_k}\mathbb{E}\left[\left|\left|\widetilde{\boldsymbol{Y}}_{k-1}\boldsymbol{f}_n\right|\right|_2^2\right]\\
&\quad\le\frac{3R^2\mu_0^2\mu_{k-1}^2}{\kappa M}\,||\boldsymbol{q}_l||_2^2\,.
\end{aligned}$$

Summing the three bounds and using $||\boldsymbol{q}_l||\le 1$ yields

$$\sigma_X^2\le 9\kappa\left(\frac{NR\mu_{k-1}^2}{M^2}\,||\boldsymbol{q}_l||_2^2+2\frac{R^2\mu_0^2\mu_{k-1}^2}{M}\right).$$

To use the Bernstein inequality it remains to find the Orlicz-1 norm of X_n. From Lemma 10 we have

$$\left|\left|\langle\widetilde{\boldsymbol{Y}}_{k-1},\boldsymbol{A}_n\rangle\right|\right|_{\psi_2}^2=\left|\left|\boldsymbol{z}_n^*\widetilde{\boldsymbol{Y}}_{k-1}\boldsymbol{f}_n\right|\right|_{\psi_2}^2\le\left|\left|\,||\boldsymbol{Q}^*\boldsymbol{z}_n||_2\,||\boldsymbol{\Lambda_{k-1}}\boldsymbol{V}^*\boldsymbol{f}_n||_2\right|\right|_{\psi_2}^2\le c\frac{R\mu_{k-1}^2}{M}.$$

For the first term we have

$$\begin{aligned}
||\boldsymbol{f}_n^*\boldsymbol{f}\langle \boldsymbol{q}_l,\boldsymbol{Q}^*\boldsymbol{z}_n\rangle||_{\psi_2}^2 &\le ||||\boldsymbol{f}_n^*||_2\,||\boldsymbol{f}||_2\,|\langle \boldsymbol{q}_l,\boldsymbol{Q}^*\boldsymbol{z}_n\rangle||_{\psi_2}^2\\
&\le N^2\,||\boldsymbol{q}_l^*\boldsymbol{Q}^*\boldsymbol{z}_n||_{\psi_2}^2\\
&\le c\frac{N^2\,||\boldsymbol{q}_l||_2^2}{M^2}.
\end{aligned}$$

For the second term we have

$$||\langle \boldsymbol{V}^*\boldsymbol{f}_n,\boldsymbol{V}^*\boldsymbol{f}\rangle z_n[l]||_{\psi_2}^2\le||||\boldsymbol{V}^*\boldsymbol{f}_n||_2\,||\boldsymbol{V}^*\boldsymbol{f}||_2\,z_n[l]||_{\psi_2}^2\le c\frac{R^2}{M}\mu_0^4.$$

Similarly, for the final term we have

$$||\langle \boldsymbol{V}^*\boldsymbol{f}_n,\boldsymbol{V}^*\boldsymbol{f}\rangle\langle \boldsymbol{q}_l,\boldsymbol{Q}^*\boldsymbol{z}_n\rangle||_{\psi_2}^2\le||||\boldsymbol{V}^*\boldsymbol{f}_n||_2\,||\boldsymbol{V}^*\boldsymbol{f}||_2\,\boldsymbol{q}_l^*\boldsymbol{Q}^*\boldsymbol{z}_n||_{\psi_2}^2\le c\frac{R^2}{M}\mu_0^4\,||\boldsymbol{q}_l||_2^2\,.$$

Now we can calculate the total Orlicz norm as

$$\begin{aligned}
||X_n||_{\psi_1}^2 &\le c\kappa^2\frac{R\mu_{k-1}^2}{M}\left(\frac{N^2\,||\boldsymbol{q}_l||_2^2}{M^2}+\frac{R^2}{M}\mu_0^4+\frac{R^2}{M}\mu_0^4\,||\boldsymbol{q}_l||_2^2\right)\\
&\le c\frac{\kappa^2RN^2}{M^3}\,||\boldsymbol{q}_l||_2^2\,\mu_{k-1}^2+c\frac{2\kappa^2R^3}{M^2}\mu_0^4\mu_{k-1}^2.
\end{aligned}$$

Since we wish to bound the square of the sum of terms, we calculate the square values of the two terms in the Bernstein inequality. The first term is bounded by

$$t\sigma_X^2\le c\beta\kappa\frac{R}{M}\mu_{k-1}^2\left(\frac{N}{M}\,||\boldsymbol{q}_l||_2^2+2R\mu_0^2\right)\log(LN),$$

and the second term is bounded by

$$t^2U_\alpha^2\log^2\left(\frac{|\Gamma_k|U_\alpha^2}{\sigma_X^2}\right)\le t^2U_\alpha^2\log^2\left(c\frac{|\Gamma_k|M\kappa^2R\mu_{k-1}^2\left(\frac{N^2}{M}\,||\boldsymbol{q}_l||_2^2+2R^2\mu_0^4\right)}{M^2\kappa R\mu_{k-1}^2\left(\frac{N}{M}\,||\boldsymbol{q}_l||_2^2+2R\mu_0^2\right)}\right)$$

$$\begin{aligned}
&\le t^2c\kappa^2\frac{R}{M^2}\mu_{k-1}^2\left(\frac{N^2}{M}\,||\boldsymbol{q}_l||_2^2+2R^2\mu_0^4\right)\log^2\left(c\frac{N^2\,||\boldsymbol{q}_l||_2^2+2MR^2\mu_0^4}{N\,||\boldsymbol{q}_l||_2^2+2RM\mu_0^2}\right)\\
&\le c\beta^2\kappa^2\frac{R}{M^2}\mu_{k-1}^2\left(\frac{N^2}{M}\,||\boldsymbol{q}_l||_2^2+2R^2\mu_0^4\right)\log^4(LN)\,,
\end{aligned}$$

where the last step assumes $L>1$ and $LN>R\mu_0^4$. Each summand is then bounded by the maximum of these two quantities with probability $1-O(|\Gamma_k|(LN)^{-\beta})$, the $|\Gamma_k|$ term coming from the union bound over all terms in each inner sum.

Using this bound on each summand, we obtain the total bound by taking a union bound, summing over $l\in[1,\cdots,L]$, and dividing by R, yielding a bound of the maximum of

$$t\sigma_X^2\le c\beta\kappa\frac{R}{M}\mu_{k-1}^2\left(\frac{N}{M}+2L\mu_0^4\right)\log(LN),$$

and

$$t^2U_\alpha^2\log^2\left(\frac{|\Gamma_k|U_\alpha^2}{\sigma_X^2}\right)\le\beta^2c\kappa^2\frac{R}{M^2}\mu_{k-1}^2\left(\frac{N}{M}+2RL\mu_0^2\right)\log^4(LN)\,,$$

with probability $1-O(M(NL)^{-\beta})$. To complete the proof, we note that if we have

$$M\ge c\beta\kappa R\left(N+L\mu_0^2\right)\log^2(LN),$$

then both terms in this bound are less than μ_{k-1}^2. ■

References

M. Aharon, M. Elad, A. Bruckstein, and Y. Katz. K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representations. *IEEE Transactions on Signal Processing*, 54(11): 4311–4322, 2006.

A. Ahmed and J. Romberg. Compressive multiplexing of correlated signals. *IEEE Transactions on Information Theory*, 61(1):479–498, 2015.

A. Ahmed, B. Recht, and J. Romberg. Blind deconvolution using convex programming. *IEEE Transactions on Information Theory*, 60(3):1711–1732, 2014.

S.-I. Amari. Characteristics of random nets of analog neuron-like elements. *IEEE Transactions on Systems, Man and Cybernetics*, (5):643–657, 1972.

N. J. Apthorpe, A. J. Riordan, R. E. Aguilar, J. Homann, Y. Gu, D. W. Tank, and S. H. Seung. Automatic neuron detection in calcium imaging data using convolutional networks. *Advances in neural information processing systems (NIPS)*, pages 3270–3278, Dec. 2016.

A. Balavoine, J. Romberg, and C. J. Rozell. Convergence and rate analysis of neural networks for sparse approximation. *IEEE Transactions on Neural Networks and Learning Systems*, 23: 1377–1389, Sept. 2012.

P. Bashivan, I. Rish, M. Yeasin, and N. Codella. Learning representations from eeg with deep recurrent-convolutional neural networks. *International Conference on Learning Representations*, 2016.

A. Berényi, Z. Somogyvári, A. J. Nagy, L. Roux, J. D. Long, S. Fujisawa, E. Stark, A. Leonardo, T. D. Harris, and G. Buzsáki. Large-scale, high-density (up to 512 channels) recording of local circuits in behaving animals. *Journal of neurophysiology*, 111(5):1132–1149, 2014.

M. Buehner and P. Young. A tighter bound for the echo state property. *IEEE Transactions on Neural Networks*, 17(3):820–824, 2006.

D. V Buonomano and W. Maass. State-dependent computations: spatiotemporal processing in cortical networks. *Nature Reviews Neuroscience*, 10(2):113–125, 2009.

L. Büsing, B. Schrauwen, and R. Legenstein. Connectivity, dynamics, and memory in reservoir computing with binary and analog neurons. *Neural computation*, 22(5):1272–1311, 2010.

E. Candès and Y. Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

E. J. Candès and Y. Plan. A probabilistic and ripless theory of compressed sensing. *IEEE Transactions on Information Theory*, 57(11):7235–7254, 2011a.

E. J. Candès and Y. Plan. Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements. *IEEE Transactions on Information Theory*, 57(4):2342–2359, 2011b.

E. J. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

E. J. Candès, J. Romberg, and Tao T. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52 (2):489–509, 2006.

A. S. Charles, P. Garrigues, and C. J. Rozell. A common network architecture efficiently implements a variety of sparsity-based inference problems. *Neural Computation*, 24(12):3317–3339, 2012.

A. S. Charles, H. L. Yap, and C. J. Rozell. Short-term memory capacity in networks via the restricted isometry property. *Neural computation*, 26(6):1198–1235, 2014.

P. Chen and D. Suter. Recovering the missing components in a large noisy low-rank matrix: Application to sfm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):1051–1063, 2004.

Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu. Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(6): 2094–2107, 2014.

Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 54(10):6232–6251, 2016.

C.-K. Chiang, C.-H. Liu, C.-H. Duan, and S.-H. Lai. Learning component-level sparse representation for image and video categorization. *IEEE Transactions on Image Processing*, 22(12): 4775–4787, 2013.

C. Christopoulos, A. Skodras, and T. Ebrahimi. The jpeg2000 still image coding system: an overview. *IEEE transactions on consumer electronics*, 46(4):1103–1127, 2000.

M. E. Davies and L. Daudet. Sparse audio representations using the mclt. *Signal processing*, 86(3): 457–470, 2006.

W. Denk, J. H. Strickler, and W. W. Webb. Two-photon laser scanning fluorescence microscopy. *Science*, 248(4951):73–6, 1990.

J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015.

M. Elad, M. A. T. Figueiredo, and Y. Ma. On the role of sparse and redundant representations in image processing. *Proceedings of the IEEE*, 98(6)972–982, 2010.

O. Faugeras, J. Touboul, and B. Cessac. A constructive mean-field analysis of multi-population neural networks with random synaptic weights and stochastic inputs. *Frontiers in computational neuroscience*, 3(1):1–26, Feb. 2009.

M. Fazel. Matrix rank minimization with applications. *PhD Thesis, Stanford University*, 2002.

M. Galtier and G. Wainrib. A local echo state property through the largest Lyapunov exponent. *Neural Networks*, 76:39–45, 2016.

S. Ganguli and H. Sompolinsky. Short-term memory in neuronal networks through dynamical compressed sensing. *Advances in Neural Information Processing Systems (NIPS)*, pages 667–675, 2010.

S. Ganguli, D. Huh, and H. Sompolinsky. Memory traces in dynamical systems. *Proceedings of the National Academy of Sciences*, 105(48):18970, 2008.

Y. Gao, E. Archer, L. Paninski, and J. P. Cunningham. Linear dynamical neural population models through nonlinear embeddings. *Advances in Neural Information Processing Systems (NIPS)*, pages 163–171, Dec. 2016.

F. Gobet, P. C. R. Lane, S. Croker, P. C. H. Cheng, G. Jones, I. Oliver, and J. M. Pine. Chunking mechanisms in human learning. *Trends in Cognitive Sciences*, 5(6):236–243, 2001.

A. Graves, A.-R. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6645–6649, 2013.

K. Gregor, I. Danihelka, A. Graves, and D. Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

D. Gross. Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory*, 57(3):1548–1566, 2011.

T. Guha and R. K. Ward. Learning sparse representations for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1576–1588, 2012.

X. Hinaut, M. Petit, G. Pointeau, and P. F. Dominey. Exploring the acquisition and production of grammatical constructions through human-robot interaction with echo state networks. *Frontiers in Neurorobotics*, 8(16):1–17, 2014.

H. Jaeger. Short term memory in echo state networks. *GMD Report 152 German National Research Center for Information Technology*, 5:1–60, 2001.

H. Jaeger and H. Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.

M. Jaggi, M. Sulovsk, et al. A simple algorithm for nuclear norm regularized problems. *International Conference on Machine Learning (ICML)*, pages 471–478, 2010.

N. Kalchbrenner, A. Graves, and I. Danihelka. Grid long short-term memory. *International Conference on Learning Representations*, 2016.

K. Kavukcuoglu, P. Sermanet, Y. L. Boureau, K. Gregor, M. Mathieu, and Y. LeCun. Learning convolutional feature hierarchies for visual recognition. *Advances in Neural Information Processing Systems (NIPS)*, pages 1090–1098, 2010.

Y. LeCun, K. Kavukcuoglu, C. Farabet, et al. Convolutional networks and applications in vision. IEEE International Symposium on Circuits and Systems *ISCAS*, pages 253–256, 2010.

R. Legenstein and W. Maass. Edge of chaos and prediction of computational performance for neural circuit models. *Neural Networks*, 20(3):323–334, 2007.

Z. Lipton, D. Kale, C. Elkan, and R. Wetzel. Learning to diagnose with lstm recurrent neural networks. *International Conference on Learning Representations*, 2016.

Z. Liu and L. Vandenberghe. Interior-point method for nuclear norm approximation with application to system identification. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1235–1256, 2009.

M. Lukoševičius. A practical guide to applying echo state networks. *Neural Networks: Tricks of the Trade*, pages 659–686. 2012.

M. Lukoševičius and H. Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, 2009.

W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

S. Mallat. Understanding deep convolutional networks. *Philosophical Transactions of the Royal Society A*, 374(2065):20150203, 2016.

G. Manjunath and H. Jaeger. Echo state property linked to an input: Exploring a fundamental characteristic of recurrent neural networks. *Neural Computation*, 25(3):671–696, 2013.

R. Maruyama, K. Maeda, H. Moroda, I. Kato, M. Inoue, H. Miyakawa, and T. Aonishi. Detecting cells using non-negative matrix factorization on calcium imaging data. *Neural Networks*, 55: 11–19, 2014.

M. Massar and S. Massar. Mean-field theory of echo state networks. *Physical Review E*, 87(4): 042809, 2013.

B. A. Olshausen and D. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(13):607–609, Jun 1996.

A. B. Patel, T. Nguyen, and R. G. Baraniuk. A probabilistic theory of deep learning. *arXiv preprint arXiv:1504.00641*, 2015.

K. Rajan, L. F. Abbott, and H. Sompolinsky. Stimulus-dependent suppression of chaos in recurrent neural networks. *Physical Review E*, 82(1):011903, 2010.

H. Rauhut. Compressive sensing and structured random matrices. *Theoretical Found. and Numerical Methods for Sparse Recovery*, 9:1–92, 2010.

B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

C. J. Rozell, D. H. Johnson, R. G. Baraniuk, and B. A. Olshausen. Sparse coding via thresholding and local competition in neural circuits. *Neural Computation*, 20(10):2526–2563, Oct 2010.

H. Sak, A. Senior, and F. Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. *Conference of International Speech Communication Association (INTERSPEECH)*, 2014.

S. Shapero, M. Zhu, P. Hasler, and C. J. Rozell. Optimal sparse approximation with integrate and fire neurons. *International Journal of Neural Systems*, 24(05):1440001, Aaug 2014.

A. Singer and M. Cucuringu. Uniqueness of low-rank matrix completion by rigidity theory. *SIAM Journal on Matrix Analysis and Applications*, 31(4):1621–1641, 2010.

H. Sompolinsky, A. Crisanti, and H. J. Sommers. Chaos in random neural networks. *Physical Review Letters*, 61(3):259, 1988.

S. Sukhbaatar, J. Weston, R. Fergus, et al. End-to-end memory networks. *Advances in Neural Information Processing Systems (NIPS)*, pages 2431–2439.

K.-C. Toh and S. Yun. An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems. *Pacific Journal of Optimization*, 6(615-640):15, 2010.

J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

J.A. Tropp, J.N. Laska, M.F. Duarte, J.K. Romberg, and R.G. Baraniuk. Beyond Nyquist: efficient sampling of sparse bandlimited signals. *IEEE Transactions on Information Theory*, 56(1), Jan. 2009. ISSN 0018-9448.

V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971.

P. Vardan, Y. Romano, and M. Elad. Convolutional neural networks analyzed via convolutional sparse coding. *arXiv preprint arXiv:1607.08194*, 2016.

M. A. Veganzones, M. Simoes, G. Licciardi, N. Yokoya, J.M. Bioucas-Dias, and J. Chanussot. Hyperspectral super-resolution of locally low rank images from complementary multisource data. *IEEE Transactions on Image Processing*, 25(1):274–288, 2016.

D. Verstraeten, B. Schrauwen, M. dHaene, and D. Stroobandt. An experimental unification of reservoir computing methods. *Neural Networks*, 20(3):391–403, 2007.

D. Verstraeten, J. Dambre, X. Dutoit, and B. Schrauwen. Memory versus non-linearity in reservoirs. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2010.

G. Wainrib. Context dependent representation in recurrent neural networks. *arXiv preprint http://arxiv.org/pdf/1506.06602.pdf*, 2015.

E. Wallace, R. M. Hamid, and P. E. Latham. Randomly connected networks have short temporal memory. *Neural Computation*, 25:1408–1439, 2013.

O. L. White, D. D. Lee, and H. Sompolinsky. Short-term memory in orthogonal neural networks. *Physical Review Letters*, 92(14):148102, 2004.

H. R. Wilson and J. D. Cowan. Excitatory and inhibitory interactions in localized populations of model neurons. *Biophysical Journal*, 12(1):1, 1972.

I. B. Yildiz, H. Jaeger, and S. J. Kiebel. Re-visiting the echo state property. *Neural networks*, 35: 1–9, 2012.

H. Zhang, W. He, L. Zhang, H. Shen, and Q. Yuan. Hyperspectral image restoration using low-rank matrix recovery. *IEEE Transactions on Geoscience and Remote Sensing*, 52(8):4729–4743, 2014.

Journal of Machine Learning Research 18 (2017) 1-35 Submitted 7/16; Published 2/17

Persistence Images: A Stable Vector Representation of Persistent Homology

Henry Adams adams@math.colostate.edu
Tegan Emerson emerson@math.colostate.edu
Michael Kirby kirby@math.colostate.edu
Rachel Neville neville@math.colostate.edu
Chris Peterson peterson@math.colostate.edu
Patrick Shipman shipman@math.colostate.edu
Department of Mathematics
Colorado State University
1874 Campus Delivery
Fort Collins, CO 80523-1874

Sofya Chepushtanova sofya.chepushtanova@wilkes.edu
Department of Mathematics and Computer Science
Wilkes University
84 West South Street
Wilkes-Barre, PA 18766, USA

Eric Hanson eric.hanson@tcu.edu
Department of Mathematics
Texas Christian University
Box 298900
Fort Worth, TX 76129

Francis Motta motta@math.duke.edu
Department of Mathematics
Duke University
Durham, NC 27708, USA

Lori Ziegelmeier lziegel1@macalester.edu
Department of Mathematics, Statistics, and Computer Science
Macalester College
1600 Grand Avenue
Saint Paul, MN 55105, USA

Editor: Michael Mahoney

Abstract

Many data sets can be viewed as a noisy sampling of an underlying space, and tools from topological data analysis can characterize this structure for the purpose of knowledge discovery. One such tool is persistent homology, which provides a multiscale description of the homological features within a data set. A useful representation of this homological information is a *persistence diagram* (PD). Efforts have been made to map PDs into spaces with additional structure valuable to machine learning tasks. We convert a PD to a finite-dimensional vector representation which we call a *persistence image* (PI), and prove the stability of this transformation with respect to small perturbations in the inputs. The discriminatory power of PIs is compared against existing methods, showing significant performance gains. We explore the use of PIs with vector-based machine learning tools, such as linear sparse support vector machines, which identify features containing discriminating topological information. Finally, high accuracy inference of parameter values from the dynamic output of a discrete dynamical system (the *linked twist map*) and a partial differential equation (the *anisotropic Kuramoto-Sivashinsky equation*) provide a novel application of the discriminatory power of PIs.

Keywords: topological data analysis, persistent homology, persistence images, machine learning, dynamical systems

1. Introduction

In recent years, the field of topology has grown to include a large set of computational tools (Edelsbrunner and Harer, 2010). One of the fundamental tools is persistent homology, which tracks how topological features appear and disappear in a nested sequence of topological spaces (Edelsbrunner and Harer, 2008; Zomorodian and Carlsson, 2005). This multiscale information can be represented as a *persistence diagram* (PD), a collection of points in the plane where each point (x, y) corresponds to a topological feature that appears at scale x and disappears at scale y. We say the feature has a persistence value of $y - x$. This compact summary of topological characteristics by finite multi-sets of points in the plane is responsible, in part, for the surge of interest in applying persistent homology to the analysis of complex, often high-dimensional data. Computational topology has been successfully applied to a broad range of data-driven disciplines (Perea and Harer, 2013; Dabaghian et al., 2012; Chung et al., 2009; Heath et al.; Singh et al., 2008; Topaz et al., 2015; Pearson et al., 2015).

Concurrent with this revolution in computational topology, a growing general interest in data analysis has driven advances in data mining, pattern recognition, and machine learning (ML). Since the space of PDs can be equipped with a metric structure (*bottleneck* or *Wasserstein* (Mileyko et al., 2011; Turner et al., 2014)), and since these metrics reveal the stability of PDs under small perturbations of the data they summarize (Cohen-Steiner et al., 2007, 2010; Chazal et al., 2014), it is possible to perform a variety of ML techniques using PDs as a statistic for clustering data sets. However, many other useful ML tools and techniques (e.g., support vector machines (SVM), decision tree classification, neural networks, feature selection, and dimension reduction methods) require more than a metric structure. In addition, the cost of computing the bottleneck or Wasserstein distance grows quickly as the number of off-diagonal points in the diagrams increases (Di Fabio and Ferri, 2015). To resolve these issues, considerable effort (which we review in §2) has been made to map PDs into spaces which are suitable for other ML tools (Bubenik, 2015; Reininghaus et al., 2015; Rouse et al., 2015; Adcock et al., 2016; Donatini et al., 1998; Ferri et al., 1997; Chung et al., 2009; Pachauri et al., 2011; Bendich et al., 2016; Chen et al., 2015; Carrière et al., 2015; Di Fabio and Ferri, 2015). With the benefits and drawbacks of these approaches in mind, we pose the following question:

Problem Statement: How can we represent a persistence diagram so that

(i) the output of the representation is a vector in $\mathbb{R}^n$,

(ii) the representation is stable with respect to input noise,

©2017 Adams, et al.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-337.html.

(iii) the representation is efficient to compute,

(iv) the representation maintains an interpretable connection to the original PD, and

(v) the representation allows one to adjust the relative importance of points in different regions of the PD?

The main contribution of this paper is to study a finite-dimensional-vector representation of a PD called a *persistence image* (PI). We first map a persistence diagram B to an integrable function $\rho_B \colon \mathbb{R}^2 \to \mathbb{R}$ called a *persistence surface*. The surface ρ_B is defined as a weighted sum of Gaussian functions,[1] one centered at each point in the PD. The idea of persistence surfaces has appeared even prior to the development of persistent homology, in Donatini et al. (1998) and Ferri et al. (1997). Taking a discretization of a subdomain of ρ_B defines a grid. A persistence image, i.e., a matrix of pixel values, can be created by computing the integral of ρ_B on each grid box. This PI is a "vectorization" of the PD, and provides a solution to the problem statement above.

Criterion (i) is the primary motivation for developing PIs. A large suite of ML techniques and statistical tools (means and variances) already exist to work with data in $\mathbb{R}^n$. Additionally, such a representation allows for the use of various distance metrics (p-norms and angle based metrics) and other measures of (dis)similarity. The remaining criteria of the problem statement (ii-v) further ensure the usefulness of this representation.

The desired flexibility of (v) is accomplished by allowing one to build a PI as a weighted sum of Gaussians, where the weightings may be chosen from a broad class of weighting functions.[2] For example, a typical interpretation is that points in a PD of high persistence are more important than points of low persistence (which may correspond to noise). One may therefore build a PI as a weighted sum of Gaussians where the weighting function is non-decreasing with respect to the persistence value of each PD point. However, there are situations in which one may prefer different measures of importance. Indeed, Bendich et al. (2016) find that, in their regression task of identifying a human brain's age from its arterial geometry, the points of medium persistence (not high persistence) best distinguish the data. In such a setting, one may choose a weighting function with largest values for the points of medium persistence. In addition, the Homology Inference Theorem (Cohen-Steiner et al., 2007) states that when given a sufficiently dense finite sample from a space X, the points in the PD with sufficiently small birth times (and sufficiently high persistence) recover the homology groups of the space; hence one may choose a weighting function that emphasizes points near the death-axis and away from the diagonal, as indicated in the leftmost yellow rectangle of Figure 2.4 in Bendich (2009). A potential disadvantage of the flexibility in (v) is that it requires a choice; however, prior knowledge of one's particular problem may inform that choice. Moreoever, our examples illustrate the effectiveness of a standard choice of weighting function that is non-decreasing with the persistence value.

The remainder of this article is organized as follows. Related work connecting topological data analysis and ML is reviewed in §2, and §3 gives a brief introduction to persistent homology, PDs from point cloud data, PDs from functions, and the bottleneck and Wasserstein metrics. PIs are defined in §4 and their stability with respect to the 1-Wasserstein distance

1. In general, ρ_B can be a weighted sum of probability density functions
2. Weighting functions are restricted only to the extent necessary for our stability results in §5.

between PDs is proved in §5. Lastly, §6 contains examples of ML techniques applied to PIs generated from samples of common topological spaces, an applied dynamical system modeling turbulent mixing, and a partial differential equation describing pattern formation in extended systems driven far from equilibrium. Our code for producing PIs is publicly available at `https://github.com/CSU-TDA/PersistenceImages`.

2. Related Work

The space of PDs can be equipped with the bottleneck or Wasserstein metric (defined in §3), and one reason for the popularity of PDs is that these metrics are stable with respect to small deviations in the inputs (Cohen-Steiner et al., 2007, 2010; Chazal et al., 2014). Furthermore, the bottleneck metric allows one to define Fréchet means and variances for a collection of PDs (Mileyko et al., 2011; Turner et al., 2014). However, the structure of a metric space alone is insufficient for many ML techniques, and a recent area of interest in the topological data analysis community has been encoding PDs in ways that broaden the applicability of persistence. For example, Adcock et al. (2016) study a ring of algebraic functions on the space of persistence diagrams, and Verovšek (2016) identifies tropical coordinates on the space of diagrams. Ferri and Landi (1999) and Di Fabio and Ferri (2015) encode a PD using the coefficients of a complex polynomial that has the points of the PD as its roots.

Bubenik (2015) develops the notion of a persistence landscape, a stable functional representation of a PD that lies in a Banach space. A persistence landscape (PL) is a function $\lambda \colon \mathbb{N} \times \mathbb{R} \to [-\infty, \infty]$, which can equivalently be thought of as a sequence of functions $\lambda_k \colon \mathbb{R} \to [-\infty, \infty]$. For $1 \leq p \leq \infty$ the p-landscape distance between two landscapes λ and λ' is defined as $\|\lambda - \lambda'\|_p$; the ∞-landscape distance is stable with respect to the bottleneck distance on PDs, and the p-landscape distance is continuous with respect to the p-Wasserstein distance on PDs. One of the motivations for defining persistence landscapes is that even though Fréchet means of PDs are not necessarily unique (Mileyko et al., 2011), a set of persistence landscapes does have a unique mean. Unique means are also a feature of PIs as they are vector representations. An advantage of PLs over PIs is that the map from a PD to a PL is easily invertible; an advantage of PIs over PLs is that PIs live in Euclidean space and hence are amenable to a broader range of ML techniques. In §6, we compare PDs, PLs, and PIs in a classification task on synthetic data sampled from common topological spaces. We find that PIs behave comparably or better than PDs when using ML techniques available to both representations, but PIs are significantly more efficient to compute. Also, PIs outperform PLs in the majority of the classification tasks and are of comparable computational efficiency.

A vector representation of a PD, due to Carrière et al. (2015), can be obtained by rearranging the entries of the distance matrix between points in a PD. In their Theorem 3.2, they prove that both the L^∞ and L^2 norms between their resulting vectors are stable with respect to the bottleneck distance on PDs. They remark that while the L^∞ norm is useful for nearest-neighbor classifiers, the L^2 norm allows for more elaborate algorithms such as SVM. However, though their stability result for the L^∞ norm is well-behaved, their constant for the L^2 norm scales undesirably with the number of points in the PD. We provide this as motivation for our Theorem 10, in which we prove the L^∞, L^1, and L^2 norms for PI vectors

are stable with respect to the 1-Wasserstein distance between PDs, and in which none of the constants depend on the number of points in the PD.

By superimposing a grid over a PD and counting the number of topological features in each bin, Rouse et al. (2015) create a feature vector representation. An advantage of this approach is that the output is easier to interpret than other more complicated representations, but a disadvantage is that the vectors are not stable for two reasons:

(i) an arbitrarily small movement of a point in a PD may move it to another bin, and

(ii) a PD point emerging from the diagonal creates a discontinuous change.

Source (i) of instability can be improved by first smoothing a PD into a surface. This idea has appeared multiple times in various forms—even prior to the development of persistent homology, Donatini et al. (1998) and Ferri et al. (1997) convert size functions (closely related to 0-dimensional PDs) into surfaces by taking a sum of Gaussians centered on each point in the diagram. This conversion is not stable due to (ii), and we view our work as a continued study of these surfaces, now also in higher homological dimensions, in which we introduce a weighting function[3] to address (ii) and obtain stability. Chung et al. (2009) produce a surface by convolving a PD with the characteristic function of a disk, and Pachauri et al. (2011) produce a surface by centering a Gaussian on each point, but both of these methods lack stability again due to (ii). Surfaces produced from random PDs are related to the empirical intensity plots of Edelsbrunner et al. (2012).

Reininghaus et al. (2015) produce a stable surface from a PD by taking the sum of a positive Gaussian centered on each PD point together with a negative Gaussian centered on its reflection below the diagonal; the resulting surface is zero along the diagonal. This approach is similar to PIs, and indeed we use a result of Reininghaus et al. (2015, Theorem 3) to show that persistence surfaces are stable only with respect to the 1-Wasserstein distance (Remark 6). Nevertheless, we propose our independently-developed surfaces as an alternative stable representation of PDs with the following potential advantages. First, the sum of non-negatively weighted Gaussians in PIs may be easier to interpret than a sum including negative Gaussians. Second, we produce vectors from persistence surfaces with well-behaved stability bounds, allowing one to use vector-based learning methods such as linear SVM. Indeed, Zeppelzauer et al. (2016) report that while the kernel of Reininghaus et al. (2015) can be used with nonlinear SVMs, in practice, this becomes inefficient for a large number of training vectors because the entire kernel matrix must be computed. Third, while the surface of Reininghaus et al. (2015) weights persistence points further from the diagonal more heavily, there are situations in which one may prefer different weightings, as discussed in §1 and item (v) of our Problem Statement. Hence, one may want weightings on PD points that are non-increasing or even decreasing when moving away from the diagonal, an option available in the PI approach.

We produce a persistence surface from a PD by taking a weighted sum of Gaussians centered at each point. We create vectors, or PIs, by integrating our surfaces over a grid, allowing ML techniques for finite-dimensional vector spaces to be applied to PDs. Persistence images are stable, and distinct homology dimensions may be concatenated together into a single vector to be analyzed simultaneously. Persistence surfaces are studied from the

3. Our weighting function is continuous and zero for points of zero persistence, i.e., points along the diagonal.

statistical point of view by Chen et al. (2015); their applications in Section 4 use the L^1 norm between these surfaces, which can be justified as a reasonable notion of distance due to Theorem 9 that proves the L^1 distance between such surfaces is stable.

Zeppelzauer et al. (2016) apply PIs to 3D surface analysis for archeological data, in which the machine learning task is to distinguish scans of natural rock surfaces from those containing ancient human-made engravings. The authors state they select PIs over other topological methods because PIs are computationally efficient and can be used with a broader set of ML techniques. PIs are compared to an aggregate topological descriptor for a PD: the first entry of this vector is the number of points in the diagram, and the remaining entries are the minimum, maximum, mean, standard deviation, variance, 1st-quartile, median, 3rd-quartile, sum of square roots, sum, and sum of squares of all the persistence values. In their three experiments, the authors find the following.

- When classifying natural rock surfaces from engravings using PDs produced from the sublevel set filtration, PIs outperform the aggregate descriptor.
- When the natural rock and engraved surfaces are first preprocessed using the completed local binary pattern (CLBP) operator for texture classificiation (Guo et al., 2010), PIs outperform the aggregate descriptor.
- The authors added PIs and the aggregate descriptor to eleven different non-topological baseline descriptors, and found that the classification accuracy of the baseline descriptor was improved more by the addition of PIs than by the addition of the aggregate descriptor.

Furthermore, Zeppelzauer et al. (2016, Table 1) demonstrate that for their machine learning task, PIs have low sensitivity to the parameter choices of resolution and variance (§4).

3. Background on Persistent Homology

Homology is an algebraic topological invariant that, roughly speaking, describes the holes in a space. The k-dimensional holes (connected components, loops, trapped volumes, etc.) of a topological space X are encoded in an algebraic structure called the k-th homology group of X, denoted $H_k(X)$. The rank of this group is referred to as the *k-th Betti number*, β_k, and counts the number of independent k-dimensional holes. For a comprehensive study of homology, see the textbook by Hatcher (2002).

Given a nested sequence of topological spaces $X_1 \subseteq X_2 \subseteq \ldots \subseteq X_n$, the inclusion $X_i \subseteq X_{i'}$ for $i \leq i'$ induces a linear map $H_k(X_i) \to H_k(X_{i'})$ on the corresponding k-th homology for all $k \geq 0$. The idea of *persistent homology* is to track elements of $H_k(X_i)$ as the scale (or "time") parameter i increases (Edelsbrunner and Harer, 2008; Zomorodian and Carlsson, 2005; Edelsbrunner and Harer, 2010). A standard way to represent persistent homology information is a *persistence diagram* (PD),[4] which is a multiset of points in the Cartesian plane $\mathbb{R}^2$. For a fixed choice of homological dimension k, each homological feature is represented by a point (x, y), whose *birth* and *death* indices x and y are the scale parameters at which that feature first appears and disappears, respectively. Since all topological

4. Another standard representation is the barcode (Ghrist, 2008).

features die after they are born, necessarily each point appears on or above the diagonal line $y = x$. A PD is a multiset of such points, as distinct topological features may have the same birth and death coordinates.[5] Points near the diagonal are often considered to be noise while those further from the diagonal represent more robust topological features.

In this paper, we produce PDs from two different types of input data:

(i) When the data is a point cloud, i.e., a finite set of points in some space, then we produce PDs using the Vietoris–Rips filtration.

(ii) When the data is a real-valued function, we produce PDs using the sublevel set filtration.[6]

For setting (i), point cloud data often comes equipped with a metric or a measure of internal (dis)similarity and is rich with latent geometric content. One approach to identifying geometric shapes in data is to consider the data set as the vertices of a simplicial complex and to add edges, triangles, tetrahedra, and higher-dimensional simplices whenever their diameter is less than a fixed choice of scale. This topological space is called the Vietoris–Rips simplicial complex, which we introduce in more detail in §A.2. The homology of the Vietoris–Rips complex depends crucially on the choice of scale, but persistent homology eliminates the need for this choice by computing homology over a range of scales (Carlsson, 2009; Ghrist, 2008). In §6.1–6.4.1, we obtain PDs from point cloud data using the Vietoris–Rips filtered simplicial complex, and we use ML techniques to classify the point clouds by their topological features.

In setting (ii), our input is a real valued function $f\colon X \to \mathbb{R}$ defined on some domain X. One way to understand the behavior of map f is to understand the topology of its sublevel sets $f^{-1}((-\infty, \epsilon])$. By letting ϵ increase, we obtain an increasing sequence of topological spaces, called the sublevel set filtration, which we introduce in more detail in §A.3. In §6.4.2, we obtain PDs from surfaces $u\colon [0,1]^2 \to \mathbb{R}$ produced from the Kuramoto-Sivashinsky equation, and we use ML techniques to perform parameter classification.

In both settings, the output of the persistent homology computation is a collection of PDs encoding homological features of the data across a range of scales. Let $\mathcal{D}$ denote the set of all PDs. The space $\mathcal{D}$ can be endowed with metrics as studied by Cohen-Steiner et al. (2007) and Mileyko et al. (2011). The *p-Wasserstein distance* defined between two PDs B and B' is given by

$$W_p(B, B') = \inf_{\gamma: B \to B'} \Big(\sum_{u \in B} ||u - \gamma(u)||_\infty^p \Big)^{1/p},$$

where $1 \leq p < \infty$ and γ ranges over bijections between B and B'. Another standard choice of distance between diagrams is $W_\infty(B, B') = \inf_{\gamma: B \to B'} \sup_{u \in B} ||u - \gamma(u)||_\infty$, referred to as the *bottleneck distance*. These metrics allow us to measure the (dis)similarity between the homological characteristics of two data sets.

5. By convention, all points on the diagonal are taken with infinite multiplicity. This facilitates the definitions of the p-Wasserstein and bottleneck distances below.
6. As explained in §A.3, (i) can be viewed as a special case of (ii).

4. Persistence Images

We propose a method for converting a PD into a vector while maintaining an interpretable connection to the original PD. Figure 1 illustrates the pipeline from data to PI starting with spectral and spatial information in $\mathbb{R}^5$ from an immunofluorescent image of a circulating tumor cell (Emerson et al., 2015).

Precisely, let B be a PD in birth-death coordinates.[7] Let $T\colon \mathbb{R}^2 \to \mathbb{R}^2$ be the linear transformation $T(x, y) = (x, y - x)$, and let $T(B)$ be the transformed multiset in birth-persistence coordinates,[8] where each point $(x, y) \in B$ corresponds to a point $(x, y - x) \in T(B)$. Let $\phi_u\colon \mathbb{R}^2 \to \mathbb{R}$ be a differentiable probability distribution with mean $u = (u_x, u_y) \in \mathbb{R}^2$. In all of our applications, we choose this distribution to be the normalized symmetric Gaussian $\phi_u = g_u$ with mean u and variance σ^2 defined as

$$g_u(x, y) = \frac{1}{2\pi\sigma^2} e^{-[(x-u_x)^2 + (y-u_y)^2]/2\sigma^2}.$$

We fix a nonnegative weighting function $f\colon \mathbb{R}^2 \to \mathbb{R}$ that is zero along the horizontal axis, continuous, and piecewise differentiable. With these ingredients, we transform the PD into a scalar function over the plane.

Definition 1 *For B a PD, the corresponding* persistence surface $\rho_B\colon \mathbb{R}^2 \to \mathbb{R}$ *is the function*

$$\rho_B(z) = \sum_{u \in T(B)} f(u)\phi_u(z).$$

The weighting function f is critical to ensure the transformation from a PD to a persistence surface is stable, which we prove in §5.

Finally, the surface $\rho_B(z)$ is reduced to a finite-dimensional vector by discretizing a relevant subdomain and integrating $\rho_B(z)$ over each region in the discretization. In particular, we fix a grid in the plane with n boxes (pixels) and assign to each the integral of ρ_B over that region.

Definition 2 *For B a PD, its* persistence image *is the collection of pixels $I(\rho_B)_p = \iint_p \rho_B \, dydx$.*

PIs provide a convenient way to combine PDs of different homological dimensions into a single object. Indeed, suppose in an experiment the PDs for H_0, H_1, …, H_k are computed. One can concatenate the PI vectors for H_0, H_1, …, H_k into a single vector representing all homological dimensions simultaneously, and then use this concatenated vector as input into ML algorithms.

When generating a PI, the user makes three choices: the resolution, the distribution (and its associated parameters), and the weighting function. A strength of PIs is that they are flexible; a weakness is that these choices are noncanonical.

7. We omit points that correspond to features with infinite persistence, e.g., the H_0 feature corresponding to the connectedness of the complete simplicial complex.
8. Instead of birth-persistence coordinates, one could also use other choices such as birth-death or (average size)-persistence coordinates. Our stability results (§5) still hold with only a slight modification to the constants.

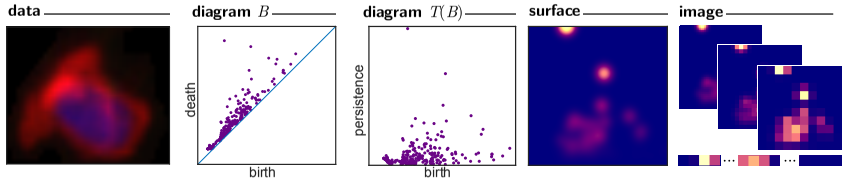


Figure 1: Algorithm pipeline to transform data into a persistence image.

Resolution of the image: The resolution of the PI corresponds to the grid being overlaid on the PD. The classification accuracy in the PI framework appears to be fairly robust to choice of resolution, as discussed in §6.2 and by Zeppelzauer et al. (2016).

The Distribution: Our method requires the choice of a probability distribution associated to each point in the PD. The examples in this paper use a Gaussian centered at each point, but other distributions may be used. The Gaussian distribution depends on a choice of variance: we leave this choice as an open problem, though the experiments in §6.2 and those of Zeppelzauer et al. (2016) show a low sensitivity to the choice of variance.

The Weighting Function: In order for our stability results in §5 to hold, our weighting function $f\colon \mathbb{R}^2 \to \mathbb{R}$ must be zero along the horizontal axis (the analogue of the diagonal in birth-persistence coordinates), continuous, and piecewise differentiable. A simple choice is a weighting function that depends only on the vertical persistence coordinate y. In order to weight points of higher persistence more heavily, functions which are nondecreasing in y, such as sigmoidal functions, are a natural choice. However, in certain ML tasks such as the work of Bendich et al. (2016) the points of small or medium persistence may perform best, and hence one may choose to use more general weighting functions. In our experiments in §6, we use a piecewise linear weighting function $f\colon \mathbb{R}^2 \to \mathbb{R}$ which only depends on the persistence coordinate y. Given $b > 0$, define $w_b\colon \mathbb{R} \to \mathbb{R}$ via

$$w_b(t) = \begin{cases} 0 & \text{if } t \le 0, \\ \frac{t}{b} & \text{if } 0 < t < b, \text{ and} \\ 1 & \text{if } t \ge b. \end{cases}$$

We use $f(x, y) = w_b(y)$, where b is the persistence value of the most persistent feature in all trials of the experiment.

In the event that the birth coordinate is zero for all points in the PD, as is often the case for H_0, it is possible to generate a 1-dimensional (instead of 2-dimensional) PI using 1-dimensional distributions. This is the approach we adopt. Appendix B displays examples of PIs for the common topological spaces of a circle and a torus with various parameter choices.

5. Stability of Persistence Surfaces and Images

Due to the unavoidable presence of noise or measurement error, tools for data analysis ought to be stable with respect to small perturbations of the inputs. Indeed, one reason for the popularity of PDs in topological data analysis is that the transformation of a data

set to a PD is stable (Lipschitz) with respect to the bottleneck metric and—given some mild assumptions about the underlying data—is also stable with respect to the Wasserstein metrics (Edelsbrunner and Harer (2010)). In §5.1, we show that persistence surfaces and images are stable with respect to the 1-Wasserstein distance between PDs. In §5.2, we prove stability with improved constants when the PI is constructed using the Gaussian distribution.

5.1 Stability for general distributions

For $h\colon \mathbb{R}^2 \to \mathbb{R}$ differentiable, define $|\nabla h| = \sup_{z \in \mathbb{R}^2} \|\nabla h(z)\|_2$ to be the maximal norm of the gradient vector of h, i.e., the largest directional derivative of h. It follows by the fundamental theorem of calculus for line integrals that for all $u, v \in \mathbb{R}^2$, we have

$$|h(u) - h(v)| \le |\nabla h| \, \|u - v\|_2. \tag{1}$$

Recall $\phi_u\colon \mathbb{R}^2 \to \mathbb{R}$ is a differentiable probability distribution with mean $u = (u_x, u_y) \in \mathbb{R}^2$. We may safely denote $|\nabla \phi_u|$ by $|\nabla \phi|$ and $\|\phi_u\|_\infty$ by $\|\phi\|_\infty$ since the maximal directional derivative and supremum of a fixed differentiable probability distribution are invariant under translation. Note that

$$\|\phi_u - \phi_v\|_\infty \le |\nabla \phi| \, \|u - v\|_2 \tag{2}$$

since for any $z \in \mathbb{R}^2$ we have $|\phi_u(z) - \phi_v(z)| = |\phi_u(z) - \phi_u(z + u - v)| \le |\nabla \phi| \, \|u - v\|_2$.

Recall that our nonnegative weighting function $f\colon \mathbb{R}^2 \to \mathbb{R}$ is defined to be zero along the horizontal axis, continuous, and piecewise differentiable.

Lemma 3 *For $u, v \in \mathbb{R}^2$, we have $\|f(u)\phi_u - f(v)\phi_v\|_\infty \le \big(\|f\|_\infty |\nabla \phi| + \|\phi\|_\infty |\nabla f|\big)\|u - v\|_2$.*

Proof For any $z \in \mathbb{R}^2$, we have

$$\begin{aligned} |f(u)\phi_u(z) - f(v)\phi_v(z)| &= \big|f(u)\big(\phi_u(z) - \phi_v(z)\big) + \big(f(u) - f(v)\big)\phi_v(z)\big| \\ &\le \|f\|_\infty \, |\phi_u(z) - \phi_v(z)| + \|\phi\|_\infty |f(u) - f(v)| \\ &\le \|f\|_\infty |\nabla \phi| \, \|u - v\|_2 + \|\phi\|_\infty |\nabla f| \, \|u - v\|_2 \qquad \text{by (2) and (1)} \\ &= \big(\|f\|_\infty |\nabla \phi| + \|\phi\|_\infty |\nabla f|\big)\|u - v\|_2. \end{aligned}$$

■

Theorem 4 *The persistence surface ρ is stable with respect to the 1-Wasserstein distance between diagrams: for $B, B' \in \mathcal{D}$ we have*

$$\|\rho_B - \rho_{B'}\|_\infty \le \sqrt{10}\big(\|f\|_\infty |\nabla \phi| + \|\phi\|_\infty |\nabla f|\big) W_1(B, B').$$

Proof Since we assume B and B' consist of finitely many points, there exists a matching γ that achieves the infimum in the Wasserstein distance. Then

$$\begin{aligned}
\|\rho_B - \rho_{B'}\|_\infty &= \| \sum_{u\in T(B)} f(u)\phi_u - \sum_{u\in T(B)} f(\gamma(u))\phi_{\gamma(u)}\|_\infty \\
&\le \sum_{u\in T(B)} \|f(u)\phi_u - f(\gamma(u))\phi_{\gamma(u)}\|_\infty \\
&\le \big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big) \sum_{u\in T(B)} \|u-\gamma(u)\|_2 \qquad \text{by Lemma 3.} \\
&\le \sqrt{2}\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big) \sum_{u\in T(B)} \|u-\gamma(u)\|_\infty \quad \text{since } \|\cdot\|_2 \le \sqrt{2}\|\cdot\|_\infty \text{ in } \mathbb{R}^2 \\
&\le \sqrt{10}\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big) \sum_{u\in B} \|u-\gamma(u)\|_\infty \quad \text{since } \|T(\cdot)\|_2 \le \sqrt{5}\|\cdot\|_\infty \\
&= \sqrt{10}\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big) W_1(B,B').
\end{aligned}$$

The step transforming from a sum over all $u \in T(B)$ to one over all $u \in B$ is necessary because the Wasserstein distance is defined using birth-death coordinates, not birth-persistence coordinates. The bound $\|T(\cdot)\|_2 \le \sqrt{5}\|\cdot\|_\infty$ follows from the fact the unit ball with respect to the L^∞ norm in $\mathbb{R}^2$ (i.e., a square) gets mapped under T to a parallelogram contained inside a ball with respect to the L^2 norm of radius $\sqrt{5}$. ■

It follows that persistence images are also stable.

Theorem 5 *The persistence image $I(\rho_B)$ is stable with respect to the 1-Wasserstein distance between diagrams. More precisely, if A is the maximum area of any pixel in the image, A' is the total area of the image, and n is the number of pixels in the image, then*

$$\begin{aligned}
\|I(\rho_B) - I(\rho_{B'})\|_\infty &\le \sqrt{10}A\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B,B') \\
\|I(\rho_B) - I(\rho_{B'})\|_1 &\le \sqrt{10}A'\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B,B') \\
\|I(\rho_B) - I(\rho_{B'})\|_2 &\le \sqrt{10n}A\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B,B').
\end{aligned}$$

The constant for the L^2 norm bound containing $\sqrt{n}$ goes to infinity as the resolution of the image increases. For this reason, in Theorem 10 we provide bounds with better constants in the specific case of Gaussian distributions.

Proof Note for any pixel p with area $A(p)$ we have

$$\begin{aligned}
|I(\rho_B)_p - I(\rho_{B'})_p| &= \Big|\iint_p \rho_B\, dydz - \iint_p \rho_{B'}\, dydx\Big| \\
&= \Big|\iint_p \rho_B - \rho_{B'}\, dydx\Big| \\
&\le A(p)\|\rho_B - \rho_{B'}\|_\infty \\
&\le \sqrt{10}A(p)\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B,B') \qquad \text{by Theorem 4.}
\end{aligned}$$

Hence we have

$$\begin{aligned}
\|I(\rho_B) - I(\rho_{B'})\|_\infty &\le \sqrt{10}A\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B,B') \\
\|I(\rho_B) - I(\rho_{B'})\|_1 &\le \sqrt{10}A'\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B,B') \\
\|I(\rho_B) - I(\rho_{B'})\|_2 &\le \sqrt{n}\|I(\rho_B) - I(\rho_{B'})\|_\infty \\
&\le \sqrt{10n}A\big(\|f\|_\infty|\nabla\phi| + \|\phi\|_\infty|\nabla f|\big)W_1(B,B').
\end{aligned}$$

■

Remark 6 *Recall $\mathcal{D}$ is the set of all PDs. The kernel $k\colon \mathcal{D}\times\mathcal{D}\to\mathbb{R}$ defined by $k(B,B') = \langle I(\rho_B), I(\rho_{B'})\rangle_{\mathbb{R}^n}$ is non-trivial and additive, and hence Theorem 3 of Reininghaus et al. (2015) implies that k is not stable with respect to W_p for any $1 < p \le \infty$. That is, when $1 < p \le \infty$ there is no constant c such that for all $B, B' \in \mathcal{D}$ we have $\|I(\rho_B) - I(\rho_{B'})\|_2 \le cW_p(B,B')$.*

5.2 Stability for Gaussian distributions

In this section, we provide stability results with better constants in the case of Gaussian distributions. With Gaussian distributions, we can control not only the L^∞ distance but also the L^1 distance between two persistence surfaces.

Our results for 2-dimensional Gaussians will rely on the following lemma for 1-dimensional Gaussians.

Lemma 7 *For $u, v \in \mathbb{R}$, let $g_u, g_v\colon \mathbb{R}\to\mathbb{R}$ be the normalized 1-dimensional Gaussians, defined via $g_u(z) = \dfrac{1}{\sigma\sqrt{2\pi}}e^{-(z-u)^2/2\sigma^2}$. If $a, b > 0$, then*

$$\|ag_u - bg_v\|_1 \le |a-b| + \sqrt{\frac{2}{\pi}}\frac{\min\{a,b\}}{\sigma}|u-v|.$$

Proof Let $\mathrm{Erf}(t) = \frac{2}{\sqrt{\pi}}\int_0^t e^{-u^2}du$. We show in Appendix C that

$$\|ag_u - bg_v\|_1 = F(v-u), \tag{3}$$

where $F\colon \mathbb{R}\to\mathbb{R}$ is defined by

$$F(z) = \begin{cases} |a-b| & \text{if } z = 0 \\ \left|a\mathrm{Erf}\left(\frac{z^2+2\sigma^2\ln(a/b)}{z\sigma 2\sqrt{2}}\right) - b\mathrm{Erf}\left(\frac{-z^2+2\sigma^2\ln(a/b)}{z\sigma 2\sqrt{2}}\right)\right| & \text{otherwise.}\end{cases}$$

The roots of F'' are $z = \pm\sigma\sqrt{2\ln(a/b)}$ and $z = \pm i\sigma\sqrt{2\ln(a/b)}$. If $a > b$, the unique positive real root is $z_a \equiv \sigma\sqrt{2\ln(a/b)}$ while if $b > a$, the unique positive real root is $z_b \equiv -i\sigma\sqrt{2\ln(a/b)}$. Since $F'(z_a) = b\sqrt{2/\pi}/\sigma$ and $F'(z_b) = a\sqrt{2/\pi}/\sigma$, we conclude that

$$\|F'\|_\infty = \sqrt{\frac{2}{\pi}}\frac{\min\{a,b\}}{\sigma}, \quad \text{and hence} \quad F(z) \le |a-b| + \sqrt{\frac{2}{\pi}}\frac{\min\{a,b\}}{\sigma}|z|.$$

The result follows by letting $z = v - u$. ■

Lemma 8 *For $u, v \in \mathbb{R}^2$, let $g_u, g_v \colon \mathbb{R}^2 \to \mathbb{R}$ be normalized 2-dimensional Gaussians. Then*

$$\|f(u)g_u - f(v)g_v\|_1 \leq \left(|\nabla f| + \sqrt{\frac{2}{\pi}} \frac{\min\{f(u), f(v)\}}{\sigma} \right) \|u - v\|_2.$$

The proof of Lemma 8 is shown in Appendix C and uses a similar construction to that of Lemma 7. We are prepared to prove the stability of persistence surfaces with Gaussian distributions.

Theorem 9 *The persistence surface ρ with Gaussian distributions is stable with respect to the 1-Wasserstein distance between diagrams: for $B, B' \in \mathcal{D}$ we have*

$$\|\rho_B - \rho_{B'}\|_1 \leq \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}} \frac{\|f\|_\infty}{\sigma} \right) W_1(B, B').$$

Proof Since we assume B and B' consist of finitely many off-diagonal points, there exists a matching γ that achieves the infimum in the Wasserstein distance. Then

$$\begin{aligned}
&\|\rho_B - \rho_{B'}\|_1 = \left\| \sum_{u \in T(B)} f(u) g_u - \sum_{u \in T(B)} f(\gamma(u)) g_{\gamma(u)} \right\|_1 \\
&\leq \sum_{u \in T(B)} \|f(u)g_u - f(\gamma(u))g_{\gamma(u)}\|_1 \\
&\leq \left(|\nabla f| + \sqrt{\frac{2}{\pi}} \frac{\|f\|_\infty}{\sigma} \right) \sum_{u \in T(B)} \|u - \gamma(u)\|_2 \quad \text{by Lemma 8, where } \min\{f(u), f(v)\} \leq \|f\|_\infty \\
&\leq \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}} \frac{\|f\|_\infty}{\sigma} \right) \sum_{u \in B} \|u - \gamma(u)\|_\infty \quad \text{since } \|T(\cdot)\|_2 \leq \sqrt{5}\|\cdot\|_\infty \\
&= \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}} \frac{\|f\|_\infty}{\sigma} \right) W_1(B, B').
\end{aligned}$$

■

It follows that persistence images are also stable.

Theorem 10 *The persistence image $I(\rho_B)$ with Gaussian distributions is stable with respect to the 1-Wasserstein distance between diagrams. More precisely,*

$$\|I(\rho_B) - I(\rho_{B'})\|_1 \leq \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}} \frac{\|f\|_\infty}{\sigma} \right) W_1(B, B')$$

$$\|I(\rho_B) - I(\rho_{B'})\|_2 \leq \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}} \frac{\|f\|_\infty}{\sigma} \right) W_1(B, B')$$

$$\|I(\rho_B) - I(\rho_{B'})\|_\infty \leq \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}} \frac{\|f\|_\infty}{\sigma} \right) W_1(B, B').$$

Proof We have

$$\begin{aligned}
\|I(\rho_B) - I(\rho_{B'})\|_1 &= \sum_p \left| \iint_p \rho_B \, dydz - \iint_p \rho_{B'} \, dydx \right| \leq \iint_{\mathbb{R}^2} |\rho_B - \rho_{B'}| \, dydz \\
&= \|\rho_B - \rho_{B'}\|_1 \leq \left(\sqrt{5}|\nabla f| + \sqrt{\frac{10}{\pi}} \frac{\|f\|_\infty}{\sigma} \right) W_1(B, B')
\end{aligned}$$

by Theorem 9. The claim follows since $\|\cdot\|_2 \leq \|\cdot\|_1$ and $\|\cdot\|_\infty \leq \|\cdot\|_1$ for vectors in $\mathbb{R}^n$. ■

6. Experiments

We perform several experiments in order to assess the added value of our vector representation of PDs. First, in §6.1, we compare the performance of PDs, PLs, and PIs in a classification task for a synthetic data set consisting of point clouds sampled from six different topological spaces using K-medoids, which requires only a metric space (instead of a vector space) structure. We find that PIs produce consistently high classification accuracy, and furthermore, the computation time for PIs is significantly faster than computing bottleneck or Wasserstein distances between PDs. In §6.2, we explore the impact that the choices of parameters determining our PIs have on classification accuracy. We find that the accuracy is insensitive to the particular choices of PI resolution and distribution variance. In §6.3, we combine PIs with a sparse support vector machine classifier to identify the most strongly differentiating pixels for classification; this is an example of a ML task which is facilitated by the fact that PIs are finite vectors. Finally, as a novel machine learning application, we illustrate the utility of PIs to infer dynamical parameter values in both continuous and discrete dynamical systems: a discrete time system called the linked twist map in §6.4.1, and a partial differential equation called the anisotropic Kuramoto-Sivashinsky equation in §6.4.2.

6.1 Comparison of PDs, PLs, and PIs using K-medoids Classification

Our synthetic data set consists of six shape classes: a unit cube, a circle of diameter one, a sphere of diameter one, three clusters with centers randomly chosen in the unit cube, three clusters within three clusters (where the centers of the minor clusters are chosen as small—i.e., $<$0.1—random perturbations from the major cluster centers), and a torus with a major diameter of one and a minor diameter of one half. We produce 25 point clouds of 500 points sampled uniformly at random from each of the six shapes, and then add a level of Gaussian noise. This gives 150 point clouds in total.

We then compute the H_0 and H_1 PDs for the Vietoris–Rips filtration (§A.2) built from each point cloud which have been endowed with the ambient Euclidean metric on $\mathbb{R}^3$.

Our goal is to compare various methods for transforming PDs into distance matrices to be used to establish proximity of topological features extracted from data. We create $3^2 \cdot 2^2 = 36$ distance matrices of size 150×150, using three choices of representation (PDs,

PLs, PIs), three choices of metric (L^1, L^2, L^∞),[9] two choices of Gaussian noise ($\eta = 0.05$, 0.1), and two homological dimensions (H_0, H_1). For example, the PD, H_1, L^2, $\eta = 0.1$, distance matrix contains the 2-Wasserstein distances between the H_1 PDs for the random point clouds with noise level 0.1. By contrast, the PI, H_1, L^2, $\eta = 0.1$ distance matrix contains all pairwise L^2 distances between the PIs[10] produced from the H_1 PDs with noise level 0.1.

We first compare these distance matrices based on how well they classify the random point clouds into shape classes via K-medoids clustering (Kaufman and Rousseeuw, 1987; Park and Jun, 2009). K-medoids produces a partition of a metric space into K clusters by choosing K points from the data set called *medoids* and assigning each metric space point to its closest medoid. The *score* of such a clustering is the sum of the distances from each point to its closest medoid. The desired output of K-medoids is the clustering with the minimal clustering score. Unfortunately, an exhaustive search for the global minimum is often prohibitively expensive. A typical approach to search for this global minimum is to choose a large selection of K random initial medoids, improve each selection of medoids iteratively in rounds until the clustering score stabilizes and then return the identified final clustering with the lowest score for each initialization. In our experiments, we choose 1,000 random initial selections of $K = 6$ medoids (as there are six shape classes) for each distance matrix, improve each selection of medoids using the Voronoi iteration method (Park and Jun, 2009), and return the clustering with the lowest classification score. To each K-medoids clustering we assign an accuracy which is equal to the percentage of random point clouds identifed with a medoid of the same shape class. In Table 1, we report the classification accuracy of the K-medoids clustering with the lowest clustering score, for each distance matrix.

Our second criterion for comparing methods to produce distance matrices is computational efficiency. In Table 1, we report the time required to produce each distance matrix, starting with 150 precomputed PDs as input. In the case of PLs and PIs, this time includes the intermediate step of transforming each PD into the alternate representation, as well as computing the pairwise distance matrix. All timings are computed on a laptop with a 1.3 GHz Intel Core i5 processor and 4 GB of memory. We compute bottleneck, 1-Wasserstein, and 2-Wasserstein distance matrices using the software of Kerber et al. (2016). For PL computations, we use the Persistence Landscapes Toolbox by Bubenik and Dlotko (2016). Our MATLAB code for producing PIs is publically available at `https://github.com/CSU-TDA/PersistenceImages`.

We see in Table 1 that PI distance matrices have higher classification accuracy than nearly every PL distance matrix, and higher classification accuracy than PDs in half of the trials.

Furthermore, the computation times for PI distance matrices are significantly lower than the time required to produce distance matrices from PDs using the bottleneck or p-Wasserstein metrics. There is certainly no guarantee that PIs will outperform PDs or PLs in any given machine learning task. However, in this experiment, persistent images provide

9. The L^1, L^2, L^∞ distances on PDs are more commonly known as the 1-Wasserstein, 2-Wasserstein, and bottleneck distances.

10. For PIs in this experiment, we use variance $\sigma = 0.1$, resolution 20×20, and the weighting function defined in §4.

Table 1: Comparing classification accuracy and times of PDs, PLs, and PIs. The timings contain the computation time in seconds for producing a 150×150 distance matrix from 150 precomputed PDs. In the case of PLs and PIs, this requires first transforming each PD into its alternate representation and then computing a distance matrix. We consider 36 distinct distance matrices: three representations (PDs, PLs, PIs), two homological dimensions (H_0, H_1), three choices of metric (L^1, L^2, L^∞), and two levels of Gaussian noise ($\eta = 0.05$, 0.1).

Distance Matrix	Accuracy (Noise 0.05)	Time (Noise 0.05)	Accuracy (Noise 0.1)	Time (Noise 0.1)
PD, H_0, L^1	96.0%	37346s	96.0%	42613s
PD, H_0, L^2	91.3%	24656s	91.3%	25138s
PD, H_0, L^∞	60.7%	1133s	63.3%	1149s
PD, H_1, L^1	100%	657s	96.0%	703s
PD, H_1, L^2	100%	984s	97.3%	1042s
PD, H_1, L^∞	81.3%	527s	66.7%	564s
PL, H_0, L^1	92.7%	29s	96.7%	33s
PL, H_0, L^2	77.3%	29s	82.0%	34s
PL, H_0, L^∞	60.7%	2s	63.3%	2s
PL, H_1, L^1	83.3%	36s	80.7%	48s
PL, H_1, L^2	83.3%	50s	66.7%	69s
PL, H_1, L^∞	74.7%	8s	66.7%	9s
PI, H_0, L^1	93.3%	9s	95.3%	9s
PI, H_0, L^2	92.7%	9s	95.3%	9s
PI, H_0, L^∞	94.0%	9s	96.0%	9s
PI, H_1, L^1	100%	17s	95.3%	18s
PI, H_1, L^2	100%	17s	96.0%	18s
PI, H_1, L^∞	100%	17s	96.0%	18s

a representation of persistent diagrams which is both useful for the classification task and also computationally efficient.

6.2 Effect of PI Parameter Choice

In any system that relies on multiple parameters, it is important to understand the effect of parameter values on the system. As such, we complete a search of the parameter space used to generate PIs on the shape data set described in §6.1 and measure K-medoids classification accuracy as a function of the parameters. We use 20 different resolutions (ranging from 5×5 to 100×100 in increments of 5), a Gaussian function with 20 different choices of variance (ranging from 0.01 to 0.2 in increments of 0.01), and the weighting function described in §4. For each set of parameters, we compute the classification accuracy of the K-medoids

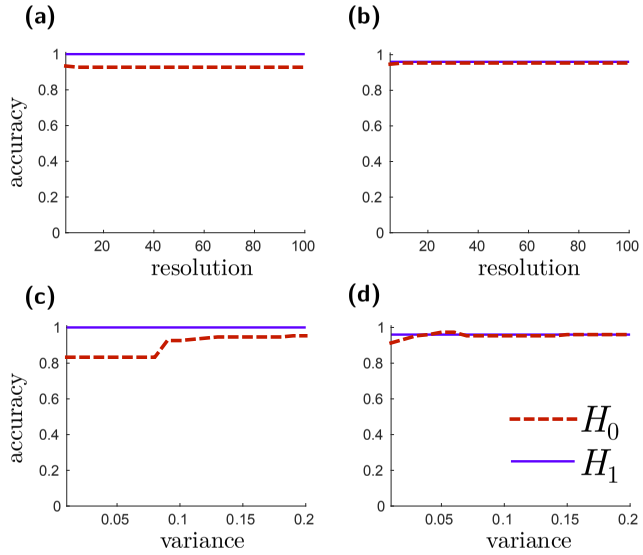


Figure 2: K-medoids classification accuracy as a function of resolution and variance for the data set of six shape classes. First column: noise level $\eta = 0.05$. Second column: noise level $\eta = 0.1$. First row: fixed variance 0.1 with resolutions ranging from 5×5 to 100×100 in increments of 5. Second row: fixed resolution 20×20 with variances ranging from 0.01 to 0.2 in increments of 0.01.

clustering with the minimum clustering score on the two sets of noise levels for the homology dimensions H_0 and H_1. We observe that the classification accuracy is insensitive to the choice of resolution and variance.

The plots in Figure 2 are characteristic of the 2-dimensional accuracy surface over all combinations of parameters in the ranges of variances and resolutions we tested. In an application to archeology, Zeppelzauer et al. (2016) find a similar robustness of PIs to the choices of resolution and variance.

6.3 Differentiating Homological Features by Sparse Support Vector Machine

The 1-norm regularized *linear* support vector machine (SVM), a.k.a. sparse SVM (SSVM) classifies data by generating a separating hyperplane that depends on very few input space features (Bradley and Mangasarian, 1998; Zhu et al., 2004; Zhang and Zhou, 2010). Such a model can be used for reducing data dimension or selecting discriminatory features. Note that linear SSVM feature selection is implemented on vectors and therefore, can be used on our PIs to select discriminatory pixels during classification. Other PD representations in the literature (Reininghaus et al., 2015; Pachauri et al., 2011) are designed to use kernel ML methods, such as *kernel* (nonlinear) SVMs. However, constructing kernel SVM classifiers

using the 1-norm results in minimizing the number of kernel functions, not the number of features in the input space (i.e., pixels in our application) (Fung and Mangasarian, 2004). Hence, for the purpose of feature selection or more precisely, PI pixel selection, we employ the linear SSVM.

We adopt the one-against-all (OAA) SSVM on the sets of H_0 and H_1 PIs from the six class shape data. In a one-against-all SSVM, there is one binary SSVM for each class to separate members of that class from members of all other classes. The PIs were generated using resolution 20×20, variance 0.0001, and noise level 0.05. Note that because of the resolution parameter choice of 20×20, each PI is a 400-dimensional vector, and the selected features will be a subset of indices corresponding to pixels within the PI. Using 5-fold cross-validated SSVM resulted in 100% accuracy comparing six sparse models with indications of the discriminatory features. Feature selection is performed by retaining the features (again, in this application, pixels) with non-zero SSVM weights, determined by magnitude comparison using weight ratios; for details see Chepushtanova et al. (2014). Figure 3 provides two examples, indicating the pixels of H_1 PIs that discriminate circles and tori from the other classes in the synthetic data set.

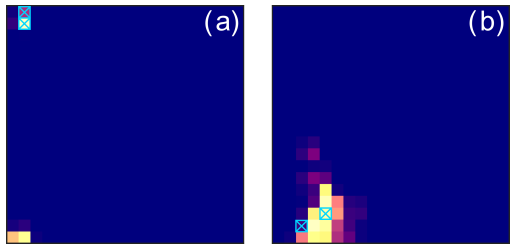


Figure 3: SSVM-based feature (pixel) selection for H_1 PIs from two classes of the synthetic data. Selected pixels are marked by blue crosses. (a) A noisy circle with the two selected pixels (indices 21 and 22 out of 400). (b) A noisy torus with the two selected pixels (indices 59 and 98 out of 400). The PI parameters used are resolution 20×20 and variance 10^{-4}, for noise level 0.05.

Feature selection produces highly interpretable results. The discriminatory pixels in the H_1 PIs that separate circles from the other classes correspond to the region where highly persistent H_1 topological features exist across all samples of a noisy circle (highlighted in Figure 3a). Alternatively, the discriminatory pixels in H_1 PIs that separate tori from the other classes correspond to points of short to moderate persistence (see Figure 3b). In this way, Figure 3b reiterates an observation of Bendich et al. (2016) that points of short to moderate persistence can contain important discriminatory information. Similar conclusions can be drawn from the discriminatory pixels of others classes (Appendix D). Our classification accuracy of 100% is obtained using only those pixels selected by SSVM (a cumulative set of only 10 distinct pixels).

6.4 Application: Determination of Dynamical System Parameters

Models of dynamic physical phenomenon rarely agree perfectly with the reality they represent. This is often due to the presence of poorly-resolved (or poorly-understood) processes which are parameterized rather than treated explicitly. As such, determination of the influence of a model parameter—which may itself be an incompletely-described conglomeration of several physical parameters—on model dynamics is a mainstay of dynamical system analysis. In the case of fitting a dynamic model to data, i.e., explicit determination of optimal model parameters, a variety of techniques exist for searching through parameter space, which often necessitate costly simulations. Furthermore, such approaches struggle when applied to models exhibiting sensitivity to initial conditions. We recast this problem as a machine-learning exercise based on the hypotheses that model parameters will be reflected directly in dynamic data in a way made accessible by persistent homology.

6.4.1 A discrete dynamical model

We approach a classification problem with data arising from the linked twist map, a discrete dynamical system modeling fluid flow. Hertzsch et al. (2007) use the linked twist map to model flows in DNA microarrays with a particular interest in understanding turbulent mixing. This demonstrates a primary mechanism giving rise to chaotic advection. The linked twist map is a Poincaré section of *eggbeater-type flow* (Hertzsch et al., 2007) in continuous dynamical systems. The Poincaré section captures the behavior of the flow by viewing a particle's location at discrete time intervals. The linked twist map is given by the discrete dynamical system

$$\begin{aligned} x_{n+1} &= x_n + r y_n(1-y_n) \mod 1 \\ y_{n+1} &= y_n + r x_n(1-x_n) \mod 1, \end{aligned}$$

where r is a positive parameter. For some values of r, the orbits $\{(x_n, y_n) : n = 0, \ldots, \infty\}$ are dense in the domain. However, for other parameter values, voids form. In either case, the truncated orbits $\{(x_n, y_n) : n = 0, \ldots, N \in \mathbb{N}\}$ exhibit complex structure.

For this experiment, we choose a set of parameter values, $r = 2.5$, 3.5, 4.0, 4.1 and 4.3, which produce a variety of orbit patterns. For each parameter value, 50 randomly-chosen initial conditions are selected, and 1000 iterations of the linked twist map are used to generate point clouds in $\mathbb{R}^2$. Figure 4 shows examples of typical orbits generated for each parameter value. The goal is to classify the trials by parameter value using PIs to capitalize on distinguishing topological features of the data. We use resolution 20×20 and a Gaussian with variance $\sigma = 0.005$ to generate the PIs. These parameters were chosen after a preliminary parameter search and classification effort. Similar results hold for a range of PI parameter values.

For a fixed r parameter value and a large number of points (many thousands), the patterns in the distributions of iterates show only small visible variations for different choices of the initial condition (x_0, y_0). However, with few points, such as in Figure 5, there are more significant variations in the patterns for different choices of initial conditions, making classification more difficult.

We perform classification and cross-validation with a discriminant subspace ensemble. This ML algorithm trains many "weak" learners on randomly chosen subspaces of the data

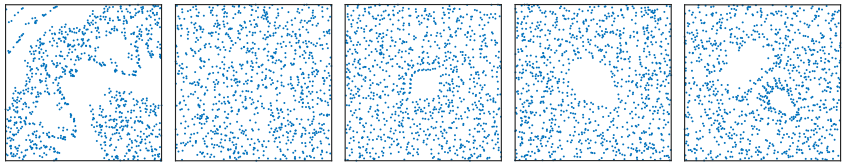


Figure 4: Examples of the first 1000 iterations, $\{(x_n, y_n) : n = 0, \ldots, 1000\}$, of the linked twist map with parameter values $r = 2$, 3.5, 4.0, 4.1 and 4.3, respectively.

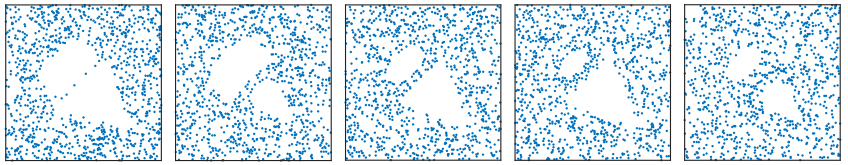


Figure 5: Truncated orbits, $\{(x_n, y_n) : n = 0, \ldots, 1000\}$, of the linked twist map with fixed $r = 4.3$ for different initial conditions (x_0, y_0).

(of a fixed dimension), and classifies and assigns a score to each point based on the current subspace. The final classification arises from an average of the scores of each data point over all learners (Ho, 1998). We perform 10 trials and average the classification accuracies. For the concatenated H_0 and H_1 PIs, this method achieves a classification accuracy of 82.5%; compared to 49.8% when using only H_0 PIs and 65.7% when using H_1 PIs. This experiment highlights two strengths of PIs: they offer flexibility in choosing a ML algorithm that is well suited to the data under consideration, and homological information from multiple dimensions may be leveraged simultaneously for greater discriminatory power.

This application is a brief example of the utility of PIs in classification of data from dynamical systems and modeling real-world phenomena, which provides a promising direction for further applications of PIs.

6.4.2 A partial differential equation

The Kuramoto-Sivashinsky (KS) equation is a partial differential equation for a function $u(x, y, t)$ of spatial variables x, y and time t that has been independently derived in a variety of problems involving pattern formation in extended systems driven far from equilibrium. Applications involving surface dynamics include surface nanopatterning by ion-beam erosion (Cuerno and Barabási, 1995; Motta et al., 2012), epitaxial growth (Villain, 1991; Wolf, 1991; Rost and Krug, 1995), and solidification from a melt (Golovin and Davis, 1998). In these applications, the nonlinear term in the KS equation may be anisotropic, resulting in the anisotropic Kuramoto-Sivashinsky (aKS) equation

$$\frac{\partial}{\partial t} u = -\nabla^2 u - \nabla^2 \nabla^2 u + r \left(\frac{\partial}{\partial x} u \right)^2 + \left(\frac{\partial}{\partial y} u \right)^2, \tag{4}$$

where $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$, and the real parameter r controls the degree of anisotropy. At a fixed time t^*, $u(x, y, t^*)$ is a patterned surface (periodic in both x and y) defined over the (x, y)-plane. Visibly, the anisotropy appears as a slight tendency for the pattern to be elongated in the vertical or horizontal direction.

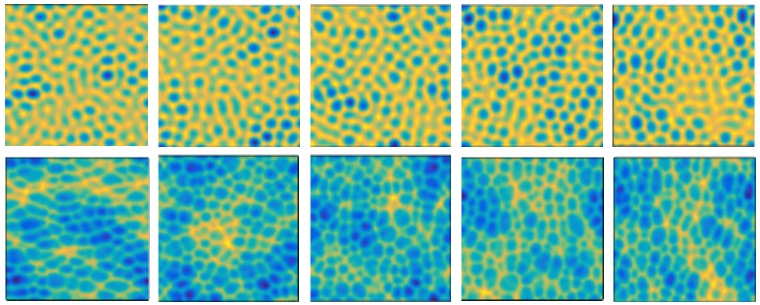


Figure 6: Plots of height-variance-normalized surfaces $u(x, y, \cdot)$ resulting from numerical simulations of the aKS equation (4). Each column represents a different parameter value: (from left) $r = 1$, 1.25, 1.5, 1.75 and 2. Each row represents a different time: $t = 3$ (top) and $t = 5$ (bottom). By $t = 5$ any anisotropic elongation of the surface pattern has visibly stabilized.

Numerical simulations of the aKS equation for a range of parameter values (columns) and simulation times (rows) are shown in Figure 6. For all simulations, the initial conditions were low-amplitude white noise. We employed a Fourier spectral method with periodic boundary conditions on a 512×512 spatial grid, with a fourth-order exponential time differencing Runge-Kutta method for the time stepping. Five values for the parameter r were chosen, namely $r = 1$, 1.25, 1.5, 1.75 and 2, and thirty trials were performed for each parameter value. Figure 7 shows the similarity between surfaces associated to two parameter values $r = 1.75$ and $r = 2$ at an early time.

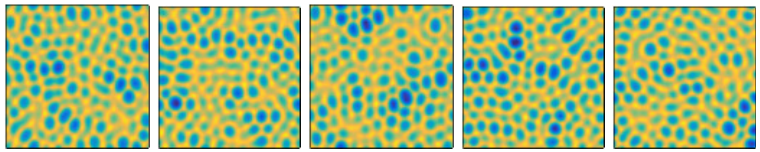


Figure 7: To illustrate the difficulty of our classification task, consider five instances of surfaces $u(x, y, 3)$ for $r = 1.75$ or $r = 2$, plotted on the same color axis. These surfaces are found by numerical integration of Equation (4), starting from random initial conditions. Can you group the images by eye?

Answer: (from left) $r = 1.75, 2, 1.75, 2, 2$.

We aim to identify the anisotropy parameter for each simulation using snapshots of surfaces $u(x, y, \cdot)$ as they evolve in time. Inference of the parameter using the surface alone proves difficult for several reasons. First, Equation (4) exhibits sensitivity to initial conditions: initially nearby solutions diverge quickly. Second, although the surface $u(x, y, t^*)$ at a fixed time is an approximation due to the finite discretization of its domain, the spatial resolution is still very large: in fact, these surfaces may be thought of as points in $\mathbb{R}^{266144}$. We were unable to perform standard classification techniques in this space. It was therefore necessary to perform some sort of dimension reduction. One such method is to simply 'resize' the surface by coarsening the discretization of the spatial domain after computing the simulation at a high resolution by replacing a block of grid elements with their average surface height. The surfaces were resized in this way to a resolution of 10×10 and a subspace discriminant ensemble was used to perform classification. Unsurprisingly, this method performs very poorly at all times (first row of Table 2).

The anisotropy parameter also influences the mean and amplitude of the surface pattern. We eliminate differences in the mean by mean-centering each surface after the simulation. To assess the impact of the variance of surface height on our task, classification was performed using a normal distribution-based classifier built on the variances of the surface heights. In this classifier, a normal distribution was fit to a training set of 2/3 of the variances for each parameter value, and the testing data was classified based on a z-test for each of the different models. That is, a p-value for each new variance was computed for membership to the five normal distributions (corresponding to the five parameter choices of r), and the surface was classified based on the model yielding the highest p-value. After the pattern has more fully emerged (by, say, time $t = 5$) this method of classification yields 75% accuracy,[11] as shown in Table 2. However, early on in the formation of the pattern, this classifier performs very poorly because height variance is not yet a discriminating feature. Figure 8 shows the normal distribution fit to the variance of the surfaces for each parameter value at times $t = 3$ and 5, and illustrates why the variance of surface height is informative only after a surface is allowed to evolve for a sufficiently long time.

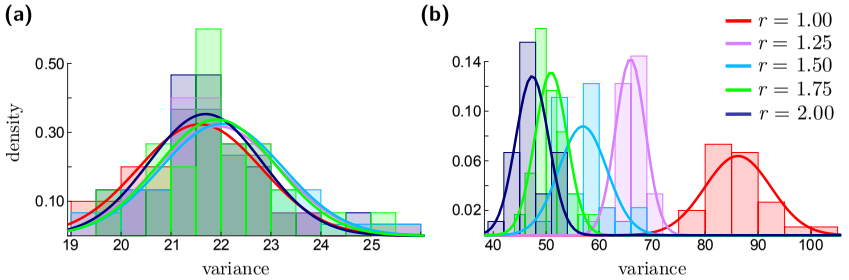


Figure 8: Histograms of the variances of surface heights for each parameter value, and the normal distribution fit to each histogram, for times (a) $t = 3$ and (b) $t = 5$.

11. Accuracy reported is averaged over 100 different training and testing partitions.

Table 2: Classification accuracies at different times of the aKS solution, using different classification approaches. Classification of times $t = 15$ and 20 result in accuracies similar to t =10.

Classification Approach	Time t=3	Time t=5	Time t=10
Subspace Discriminant Ensemble, Resized Surfaces	26.0 %	19.3%	19.3 %
Variance Normal Distribution Classifier	20.74%	75.2%	77.62 %
Subspace Discriminant Ensemble, H_0 PIs	58.3 %	96.0 %	94.7 %
Subspace Discriminant Ensemble, H_1 PIs	67.7 %	87.3 %	93.3%
Subspace Discriminant Ensemble, H_0 and H_1 PIs	72.7 %	95.3 %	97.3 %

Variance of a surface is reflected in its sublevel set filtration (see §A.3 for more details) PD. Yet, the PD and the subsequent PI contain additional topological structure, which may reveal other influences of the anisotropy parameter on the evolution of the surface. Persistence diagrams were computed using the sublevel set filtration, and PIs were generated with resolution 10×10 and a Gaussian with variance $\sigma = 0.01$. We think of our pipeline to a PI as a dimensionality reduction in this case, taking a surface which in actuality is a very high-dimensional point and producing a much lower dimensional one that retains meaningful characteristics of the original surface.

We again use a subspace discriminant ensemble to classify PIs by parameter. Table 2 compares these results to the same technique applied to low dimensional approximations of the raw surfaces and the normal distribution-based classifier built from surface variance alone. At each time in the system evolution, the best classification accuracy results from using PIs, improving accuracies over using either low resolution approximations of the surfaces or variance of surface height alone by at least 20%, including at early times in the evolution of the surface when pattern amplitudes are not visibly differentiated (see Figure 7). We postulate that PIs capture more subtle topological information that is useful for identifying the parameter used to generate each surface.

As we observed in §6.4.1, concatenating H_0 and H_1 PIs can notably improve the classification accuracy over either feature vector individually. We again note that classification accuracy appears insensitive to the PI parameters. For example, when the variance of the Guassians used to generate the PIs was varied from 0.0001 to 0.1, the classification accuracy of the H_0 PIs, changed by less than one percentage point. The classification accuracy for H_1 fluctuated in a range of approximately five points. For a fixed variance, when the resolution of the image was varied from 5 to 20, the H_0 accuracy varied by little more than three points until the accuracy dropped by six points for a resolution of 25.

PIs performed remarkably well in this classification task, allowing one to capitalize on subtle structural differences in the patterns and significantly reduce the dimension of the data for classification. There is more to be explored in the realm of pattern formation and persistence that is outside the scope of this paper.

7. Conclusion

PIs offer a stable representation of the topological characteristics captured by a PD. Through this vectorization, we open the door to a myriad of ML tools. This serves as a vital bridge between the fields of ML and topological data analysis and enables one to capitalize on topological structure (even in multiple homological dimensions) in the classification of data.

We have shown PIs yield improved classification accuracy over PLs and PDs on sampled data of common topological spaces at multiple noise levels using K-medoids. Additionally, computing distances between PIs requires significantly less computation time compared to computing distances between PDs, and comparable computation times with PLs. Through PIs, we have gained access to a wide variety of ML tools, such as SSVM which can be used for feature selection. Features (pixels) selected as discriminatory in a PI are interpretable because they correspond to regions of a PD. We have explored data sets derived from dynamical systems and illustrated that topological information of solutions can be used for inference of parameters since PIs encapsulate this information in a form amenable to ML tools, resulting in high accuracy rates for data that is difficult to classify.

The classification accuracy is robust to the choice of parameters for building PIs, providing evidence that it is not necessary to perform large-scale parameter searches to achieve reasonable classification accuracy. This indicates the utility of PIs even when there is not prior knowledge of the underlying data (i.e., high noise level, expected holes, etc.). The flexibility of PIs allows for customization tailored to a wide variety of real-world data sets.

Acknowledgments

We would like to acknowledge the research group of Paul Bendich at Duke University for allowing us access to a persistent homology package, which greatly reduced computational time and made analysis of large point clouds feasible. This code can be accessed via GitLab after submitting a request to Paul Bendich. This research is partially supported by the National Science Foundation under Grants No. DMS-1228308, DMS-1322508, NSF DMS-1115668, NSF DMS-1412674, NSF DMS-1615909, and DMR-1305449 as well as the DOD-USAF under Award Number FA9550-12-1-0408.

Appendix A. Homology and Data

Homology is an invariant that characterizes the topological properties of a topological space X. In particular, homology measures the number of connected components, loops, trapped volumes, and so on of a topological space, and can be used to distinguish distinct spaces from one another. More explicitly, the k-dimensional holes of a space generate a homology group, $H_k(X)$. The rank of this group is referred to as the *k-th Betti number*, β_k, and counts the number of k-dimensional holes of X. For a comprehensive study of homology, see the textbook of Hatcher (2002).

A.1 Simplicial Complexes and Homology

Simplicial complexes are one way to define topological spaces combinatorially. More precisely, a *simplicial complex* S consists of vertices (0-simplices), edges (1-simplices), triangles (2-simplices), tetrahedra (3-simplices), and higher-dimensional k-simplices (containing $k+1$ vertices), such that

- if σ is a simplex in S then S contains all lower-dimensional simplices of σ, and
- the non-empty intersection of any two simplices in S is a simplex in S.

The following setup is necessary for a rigorous definition of (simplicial) homology. To a simplicial complex, one can associate a chain complex of vector spaces over a field $\mathbb{F}$ (often a finite field $\mathbb{Z}/p\mathbb{Z}$ for p a small prime),

$$\cdots \to C_{k+1} \xrightarrow{\partial_{k+1}} C_k \xrightarrow{\partial_k} C_{k-1} \to \cdots .$$

Here, vector space C_k consists of all $\mathbb{F}$-linear combinations of the k simplices of S, and has as a basis the set of all k-simplices. The linear map $\partial_k : C_k \to C_{k-1}$, known as the *boundary operator*, maps a k-simplex to its boundary, a sum of its $(k-1)$-faces. More formally, the boundary map acts on a k-simplex $[v_0, v_1, \ldots, v_k]$ by

$$\partial_k([v_0, v_1, \ldots, v_k]) = \sum_{i=0}^{k} (-1)^i [v_0, \ldots, \hat{v_i}, \ldots, v_k],$$

where $[v_0, \ldots, \hat{v_i}, \ldots, v_k]$ is the $(k-1)$-simplex obtained from $[v_0, \ldots, v_k]$ by removing vertex v_i. We define two subspaces of C_k: subspace $Z_k = \mathsf{ker}(\partial_k)$ is known as the *k-cycles*, and subspace $B_k = \mathsf{im}(\partial_{k+1}) = \partial_{k+1}(C_{k+1})$ is known as the *k-boundaries*. The boundary operator satisfies the property $\partial_k \circ \partial_{k+1} = 0$, which implies the inclusion $B_k \subseteq Z_k$.

Homology seeks to uncover an equivalence class of cycles that enclose a k-dimensional hole—that is, cycles which are not also boundaries of k-simplices. To this end, the k-th order homology is defined as $H_k(S) = Z_k/B_k$, a quotient of vector spaces. The k-th Betti number $\beta_k = \dim(H_k(S))$ is the dimension of this vector space, and counts the number of independent holes of dimension k. More explicitly, β_0 counts the number of connected components, β_1 the number of loops, β_2 the number of trapped volumes, and so on. Betti numbers are a topological invariant, meaning that topologically equivalent spaces have the same Betti numbers.

A.2 Persistence Diagrams from Point Cloud Data

One way to approximate the topological characteristics of a point cloud data set is to build a simplicial complex on top of it. Though there are a variety of methods to do so, we restrict attention to the *Vietoris–Rips simplicial complex* due to its computational tractability (Ghrist, 2008). Given a data set Y (equipped with a metric) and a scale parameter $\epsilon \geq 0$, the Vietoris–Rips complex S_ϵ has Y as its set of vertices and has a k-simplex for every collection of $k+1$ vertices whose pairwise distance is at most ϵ. However, it is often not apparent how to choose scale ϵ. Selecting ϵ too small results in a topological space with

a large number of connected components, and selecting ϵ too large results in a topological space that is contractible (equivalent to a single point).

The idea of persistent homology is to compute homology at many scales and observe which topological features persist across those scales (Ghrist, 2008; Carlsson, 2009; Edelsbrunner and Harer, 2008). Indeed, if $\epsilon_1 \leq \epsilon_2 \leq \ldots \leq \epsilon_m$ is an increasing sequence of scales, then the corresponding Vietoris–Rips simplicial complexes form a filtered sequence $S_{\epsilon_1} \subseteq S_{\epsilon_2} \subseteq \ldots \subseteq S_{\epsilon_m}$. As ϵ varies, so does the homology of S_ϵ, and for any homological dimension k we get a sequence of linear maps $H_k(S_{\epsilon_1}) \to H_k(S_{\epsilon_2}) \to \ldots \to H_k(S_{\epsilon_m})$. Persistent homology tracks the homological features over a range of values of ϵ. Those features which persist over a larger range are considered to be true topological characteristics, while short-lived features are often considered as noise.

For each choice of homological dimension k, the information measured by persistent homology can be presented as a *persistence diagram* (PD), a multiset of points in the plane. Each point $(x, y) = (\epsilon, \epsilon')$ corresponds to a topological feature that appears (is 'born') at scale parameter ϵ and which no longer remains ('dies') at scale ϵ'. Since all topological features die after they are born, this is an embedding into the upper half plane, above the diagonal line $y = x$. Points near the diagonal are considered to be noise while those further from the diagonal represent more robust topological features.

A.3 Persistence Diagrams from Functions

Let X be a topological space and let $f\colon X \to \mathbb{R}$ be a real-valued function. One way to understand the behavior of map f is to understand the topology of its sublevel sets $f^{-1}((-\infty, \epsilon])$, where $\epsilon \in \mathbb{R}$. Indeed, given $\epsilon_1 \leq \epsilon_2 \leq \ldots \leq \epsilon_m$, one can study map f using the persistent homology of the resulting filtration of topological spaces, known as the sublevel set filtration:

$$f^{-1}((-\infty, \epsilon_1]) \subseteq f^{-1}((-\infty, \epsilon_2]) \subseteq \ldots \subseteq f^{-1}((-\infty, \epsilon_m]).$$

If X is a simplicial complex, then one can produce an increasing sequence of simplicial complexes using a modification of this procedure called the lower star filtration (Edelsbrunner and Harer, 2010). Similarly, if X is a cubical complex (an analogue of a simplicial complex that is instead a union of vertices, edges, squares, cubes, and higher-dimensional cubes), then one can produce an increasing sequence of cubical complexes.

In §6.4.2, we study surfaces $u\colon [0,1]^2 \to \mathbb{R}$ produced from the Kuramoto-Sivashinsky equation. The domain $[0,1]^2$ is discretized into a grid of 512×512 vertices, i.e., a 2-dimensional cubical complex with 512^2 vertices, $511 \cdot 512$ horizontal edges, $511 \cdot 512$ vertical edges, and 511^2 squares. We produce an increasing sequence of cubical complexes as follows:

- A vertex v is included at scale ϵ if $u(v) \leq \epsilon$.
- An edge is included at scale ϵ if both of its vertices are present.
- A square is included at scale ϵ if all four of its vertices are present.

Our PDs are obtained by taking the persistent homology of the sublevel set filtration of this cubical complex.

We remark that PDs from point cloud data in §A.2 can be viewed as a specific case of PDs from functions. Indeed, given a data set X in some metric space (M, d), let $d_X : M \to \mathbb{R}$ be the distance function to set X, defined by $d_X(m) = \inf_{x \in X} d(x, m)$ for all $m \in M$. Note that $d_X^{-1}((-\infty, \epsilon])$ is the union of the metric balls of radius ϵ centered at each point in X. For $\epsilon_1 \leq \epsilon_2 \leq \ldots \leq \epsilon_m$, the persistent homology of

$$d_X^{-1}((-\infty, \epsilon_1]) \subseteq d_X^{-1}((-\infty, \epsilon_2]) \subseteq \ldots \subseteq d_X^{-1}((-\infty, \epsilon_m])$$

is identical to the persistent homology of a simplicial complex filtration called the *Čech complex*. Furthermore, the persistent homology of the Vietoris–Rips complex is an approximation of the persistent homology of the Čech complex (Edelsbrunner and Harer, 2010, Section III.2).

Appendix B. Examples of Persistence Images

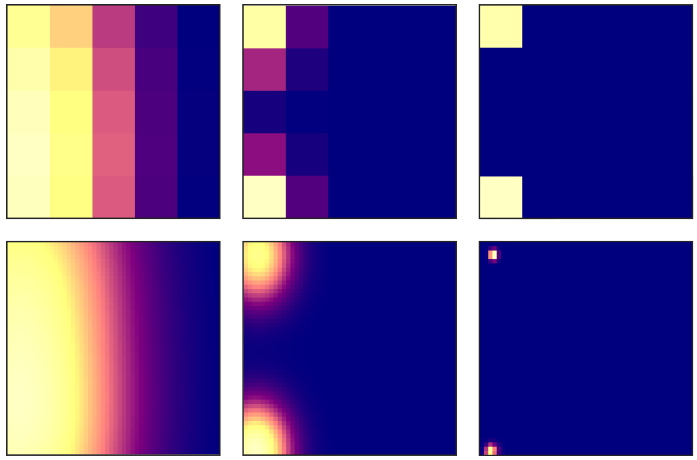


Figure 9: Examples of PIs for homology dimension H_1 arising from a noisy circle with a variety of resolutions and variances. The first row has resolution 5×5 while the second has 50×50. The columns have variance $\sigma = 0.2$, $\sigma = 0.01$, and $\sigma = 0.0001$, respectively.

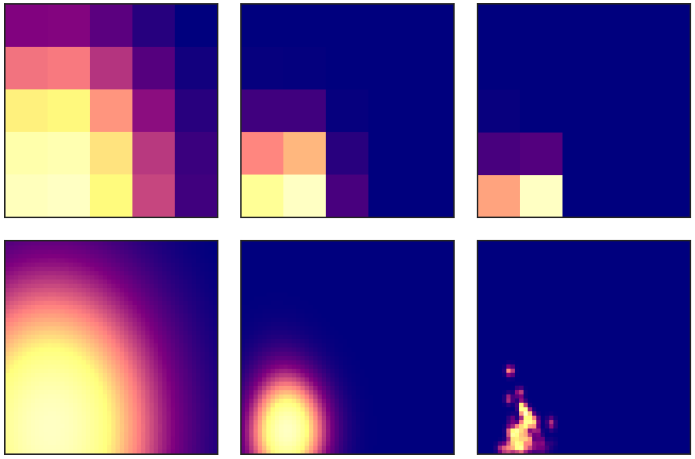


Figure 10: Examples of PIs for homology dimension H_1 arising from a noisy torus with a variety of resolutions and variances. The first row has resolution 5×5 while the second has 50×50. The columns have variance $\sigma = 0.2$, $\sigma = 0.01$, and $\sigma = 0.0001$, respectively.

Appendix C. Proofs of Equation (3) and Lemma 8

Let $u, v \in \mathbb{R}$ and $a, b > 0$. Equation (3) states that $\|ag_u - bg_v\|_1 = F(v-u)$, where $F : \mathbb{R} \to \mathbb{R}$ is defined by

$$F(z) = \begin{cases} |a-b| & \text{if } z = 0 \\ \left| a\text{Erf}\left(\frac{z^2+2\sigma^2\ln(a/b)}{z\sigma 2\sqrt{2}}\right) - b\text{Erf}\left(\frac{-z^2+2\sigma^2\ln(a/b)}{z\sigma 2\sqrt{2}}\right)\right| & \text{otherwise.} \end{cases}$$

Proof If $v = u$ then the statement follows from the fact that g_u and g_v are normalized to have unit area under the curve. Hence we may assume $u \neq v$.

For $u \neq v$ a straightforward calculation shows there is a unique real solution z^* to $ag_u(z) = bg_v(z)$, namely

$$z^* = \frac{v^2 - u^2 + 2\sigma^2 \ln(a/b)}{2(v-u)}.$$

Note

$$\|ag_u - bg_v\|_1 = \int_{-\infty}^{\infty} |ag_u(z) - bg_v(z)|dz = \left| \int_{-\infty}^{z^*} ag_u(z) - bg_v(z)dz + \int_{z^*}^{\infty} bg_v(z) - ag_u(z)dz \right|. \quad (5)$$

There are four integrals to compute, and we do each one in turn. We have

$$\begin{aligned}
\int_{-\infty}^{z^*} a g_u(z)\, dz &= \frac{a}{\sigma\sqrt{2\pi}} \int_{-\infty}^{z^*} e^{-(z-u)^2/2\sigma^2}\, dz \\
&= \frac{a}{\sqrt{\pi}} \int_{-\infty}^{P(v-u)} e^{-t^2}\, dt \qquad \text{by substitution } t = \tfrac{z-u}{\sigma\sqrt{2}} \\
&= \frac{a}{\sqrt{\pi}} \left(\int_{-\infty}^{0} e^{-t^2}\, dt + \int_{0}^{P(v-u)} e^{-t^2}\, dt \right) \\
&= \frac{a}{\sqrt{\pi}} \left(\frac{\sqrt{\pi}}{2} + \frac{\sqrt{\pi}}{2} \mathrm{Erf}(P(v-u)) \right) \\
&= \frac{a}{2} \left(1 + \mathrm{Erf}(P(v-u)) \right),
\end{aligned}$$

where $P(z) = \frac{z^2 + 2\sigma^2 \ln(a/b)}{z\sigma 2\sqrt{2}}$. Nearly identical calculations show

$$\begin{aligned}
\int_{z^*}^{\infty} a g_u(z)\, dz &= \frac{a}{2} \left(1 - \mathrm{Erf}(P(v-u)) \right) \\
\int_{-\infty}^{z^*} b g_v(z)\, dz &= \frac{b}{2} \left(1 + \mathrm{Erf}(Q(v-u)) \right) \\
\int_{z^*}^{\infty} b g_v(z)\, dz &= \frac{b}{2} \left(1 - \mathrm{Erf}(Q(v-u)) \right),
\end{aligned}$$

where $Q(z) = \frac{-z^2 + 2\sigma^2 \ln(a/b)}{z\sigma 2\sqrt{2}}$. Plugging back into (5) gives $\|a g_u - b g_v\|_1 = F(v-u)$. ■

We now give the proof of Lemma 8.

Lemma 8. For $u, v \in \mathbb{R}^2$, let $g_u, g_v \colon \mathbb{R}^2 \to \mathbb{R}$ be normalized 2-dimensional Gaussians. Then

$$\|f(u) g_u - f(v) g_v\|_1 \leq \left(|\nabla f| + \sqrt{\frac{2}{\pi}} \frac{\min\{f(u), f(v)\}}{\sigma} \right) \|u - v\|_2.$$

Proof The result will follow from the observation that we can reduce the two-dimensional case involving Gaussians centered at $u, v \in \mathbb{R}^2$ to one-dimensional Gaussians centered at 0 and $r = \|u - v\|_2$. Let $u = (u_x, u_y)$ and $v = (v_x, v_y)$; we may assume $u_x > v_x$ w.l.o.g. Let (r, θ) be the magnitude and angle of vector $u - v$ when expressed in polar coordinates. The change of variables $(z, w) = R_\theta(x - v_x, y - v_y)$, where R_θ is the clockwise rotation of the

plane by θ, gives

$$\begin{aligned}
&\|f(u) g_u - f(v) g_v\|_1 \\
&= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left| \frac{f(u)}{2\pi\sigma^2} e^{-[(x-u_x)^2 + (y-u_y)^2]/2\sigma^2} - \frac{f(v)}{2\pi\sigma^2} e^{-[(x-v_x)^2 + (y-v_y)^2]/2\sigma^2} \right| dy\, dx \\
&= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left| \frac{f(u)}{2\pi\sigma^2} e^{-[w^2 + (z-r)^2]/2\sigma^2} - \frac{f(v)}{2\pi\sigma^2} e^{-[w^2 + z^2]/2\sigma^2} \right| dz\, dw \\
&= \int_{-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-w^2/2\sigma^2} \left[\int_{-\infty}^{\infty} \left| \frac{f(u)}{\sigma\sqrt{2\pi}} e^{-(z-r)^2/2\sigma^2} - \frac{f(v)}{\sigma\sqrt{2\pi}} e^{-z^2/2\sigma^2} \right| dz \right] dw \\
&= \|f(u) g_r - f(v) g_0\|_1 \int_{-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-w^2/2\sigma^2} dw \quad \text{with } g_0, g_r \text{ 1-dimensional Gaussians} \\
&= \|f(u) g_r - f(v) g_0\|_1 \\
&\leq |f(u) - f(v)| + \sqrt{\frac{2}{\pi}} \frac{\min\{f(u), f(v)\}}{\sigma} \|u - v\|_2 \quad \text{by Lemma 7} \\
&\leq \left(|\nabla f| + \sqrt{\frac{2}{\pi}} \frac{\min\{f(u), f(v)\}}{\sigma} \right) \|u - v\|_2.
\end{aligned}$$

■

Appendix D. SSVM-based Feature Selection

We performed feature selection using one-against-all (OAA) SSVM on the six classes of synthetic data with noise level $\eta = 0.05$. The PIs used in the experiments were generated from the H_1 PDs, with the parameter choices of resolution 20×20 and variance $\sigma = 0.0001$. Note that because of the resolution parameter choice of 20×20, each PI is a vector in $\mathbb{R}^{400}$, and the selected features will be a subset of indices corresponding to pixels within the PI. We trained an OAA SSVM model for PIs of dimension H_1. In the experiment, we used 5-fold cross-validation and obtained 100% overall accuracy. Feature selection was performed by retaining the features with non-zero SSVM weights, determined by magnitude comparison using weight ratios (Chepushtanova et al., 2014). The resulting six sparse models contain subsets of discriminatory features for each class. Note that one can use only these selected features for classification without loss in accuracy. These features correspond to discriminatory pixels in the persistence images.

Figure 11 shows locations of pixels in the vectorized PIs selected by OAA SSVM that discriminate each class from all the others.

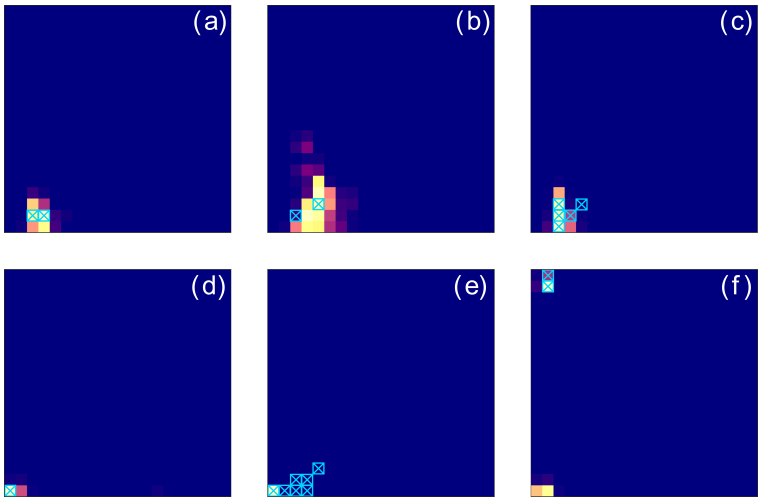


Figure 11: SSVM-based feature (pixel) selection for H_1 PIs from the six classes of the synthetic data. The parameters used are resolution 20×20 and variance 0.0001, for noise level 0.05. Selected pixels are marked by blue crosses. (a) A noisy solid cube with the two selected pixels (indices 59 and 79 out of 400). (b) A noisy torus with the two selected pixels (indices 59 and 98 out of 400). (c) A noisy sphere with the five selected pixels (indices 58, 59, 60, 79, and 98 out of 400). (d) Noisy three clusters with the one selected pixel (index 20 out of 400). (e) Noisy three clusters within three clusters with the seven selected pixels (indices 20, 40, 59, 60, 79, 80, and 98 out of 400). (f) A noisy circle with the two selected pixels (indices 21 and 22 out of 400).

References

Aaron Adcock, Erik Carlsson, and Gunnar Carlsson. The ring of algebraic functions on persistence bar codes. *Homology, Homotopy and Applications*, 18(1):381–402, 2016.

Paul Bendich. *Analyzing Stratified Spaces Using Persistent Versions of Intersection and Local Homology*. PhD thesis, Duke University, 2009.

Paul Bendich, James S. Marron, Ezra Miller, Alex Pieloch, and Sean Skwerer. Persistent homology analysis of brain artery trees. *Ann. Appl. Stat.*, 10(1):198–218, 2016.

Paul S. Bradley and Olvi L. Mangasarian. Feature selection via concave minimization and support vector machines. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 82–90, San Francisco, CA, 1998.

Peter Bubenik. Statistical topological data analysis using persistence landscapes. *The Journal of Machine Learning Research*, 16(1):77–102, 2015.

Peter Bubenik and Pawel Dlotko. A persistence landscapes toolbox for topological statistics. *Journal of Symbolic Computation*, 2016. Accepted.

Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46 (2):255–308, 2009.

Mathieu Carrière, Steve Y. Oudot, and Maks Ovsjanikov. Stable topological signatures for points on 3d shapes. In *Computer Graphics Forum*, volume 34, pages 1–12, 2015.

Frédéric Chazal, Vin de Silva, and Steve Oudot. Persistence stability for geometric complexes. *Geometriae Dedicata*, 173(1):193–214, 2014.

Yen-Chi Chen, Daren Wang, Alessandro Rinaldo, and Larry Wasserman. Statistical analysis of persistence intensity functions. *arXiv preprint arXiv:1510.02502*, 2015.

Sofya Chepushtanova, Christopher Gittins, and Michael Kirby. Band selection in hyperspectral imagery using sparse support vector machines. In *Proceedings SPIE DSS 2014*, volume 9088, pages 90881F–90881F15, 2014.

Moo K. Chung, Peter Bubenik, and Peter T. Kim. Persistence diagrams of cortical surface data. In *Information Processing in Medical Imaging*, pages 386–397. Springer, 2009.

David Cohen-Steiner, Herbert Edelsbrunner, and John Harer. Stability of persistence diagrams. *Discrete & Computational Geometry*, 37(1):103–120, 2007.

David Cohen-Steiner, Herbert Edelsbrunner, John Harer, and Yuriy Mileyko. Lipschitz functions have L_p-stable persistence. *Foundations of computational mathematics*, 10(2): 127–139, 2010.

Rodolfo Cuerno and Albert-Lásló Barabási. Dynamic scaling of ion-sputtered surfaces. *Physical Review Letters*, 74:4746, 1995.

Yu Dabaghian, Facundo Memoli, Loren Frank, and Gunnar Carlsson. A topological paradigm for hippocampal spatial map formation using persistent homology. *PLoS Computational Biology*, 8(8):e1002581, 2012.

Barbara Di Fabio and Massimo Ferri. Comparing persistence diagrams through complex vectors. In *International Conference on Image Analysis and Processing 2015 Part I; Editors V. Murino, E. Puppo, LNCS 9279*, pages 294–305, 2015.

Pietro Donatini, Patrizio Frosini, and Alberto Lovato. Size functions for signature recognition. In *SPIE's International Symposium on Optical Science, Engineering, and Instrumentation*, pages 178–183, 1998.

Herbert Edelsbrunner and John Harer. Persistent homology – a survey. *Contemporary Mathematics*, 453:257–282, 2008.

Herbert Edelsbrunner and John Harer. *Computational topology: An introduction*. American Mathematical Society, 2010.

Herbert Edelsbrunner, Alexandr Ivanov, and Roman Karasev. Current open problems in discrete and computational geometry. *Modelirovanie i Analiz Informats. Sistem*, 19(5): 5–17, 2012.

Tegan Emerson, Michael Kirby, Kelly Bethel, Anand Kolatkar, Madelyn Luttgen, Stephen O'Hara, Paul Newton, and Peter Kuhn. Fourier-ring descriptor to characterize rare circulating cells from images generated using immunofluorescence microscopy. *Computerized Medical Imaging and Graphics*, 40:70–87, 2015.

Massimo Ferri and Claudia Landi. Representing size functions by complex polynomials. *Proc. Math. Met. in Pattern Recognition*, 9:16–19, 1999.

Massimo Ferri, Patrizio Frosini, Alberto Lovato, and Chiara Zambelli. Point selection: A new comparison scheme for size functions (with an application to monogram recognition). In *Computer Vision ACCV'98*, pages 329–337. Springer, 1997.

Glenn M. Fung and Olvi L. Mangasarian. A feature selection newton method for support vector machine classification. *Computational Optimization and Applications*, 28(2):185–202, 2004.

Robert Ghrist. Barcodes: The persistent topology of data. *Bulletin of the American Mathematical Society*, 45(1):61–75, 2008.

Alexander A. Golovin and Stephen H. Davis. Effect of anisotropy on morphological instability in the freezing of a hypercooled melt. *Physica D: Nonlinear Phenomena*, 116:363–391, 1998.

Zhenhua Guo, Lei Zhang, and David Zhang. A completed modeling of local binary pattern operator for texture classification. *IEEE Transactions on Image Processing*, 19(6):1657–1663, 2010.

Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

Kyle Heath, Natasha Gelfand, Maks Ovsjanikov, Mridul Aanjaneya, and Leonidas J Guibas. Image webs: Computing and exploiting connectivity in image collections. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Jan-Martin Hertzsch, Rob Sturman, and Stephen Wiggins. DNA microarrays: Design principles for maximizing ergodic, chaotic mixing. *Small*, 3(2):202–218, 2007.

Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

Leonard Kaufman and Peter Rousseeuw. *Clustering by means of medoids*. North-Holland, 1987.

Michael Kerber, Dmitriy Morozov, and Arnur Nigmetov. Geometry helps to compare persistence diagrams. In *2016 Proceedings of the Eighteenth Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 103–112, 2016.

Yuriy Mileyko, Sayan Mukherjee, and John Harer. Probability measures on the space of persistence diagrams. *Inverse Problems*, 27(12):124007, 2011.

Francis C. Motta, Patrick D. Shipman, and R. Mark Bradley. Highly ordered nanoscale surface ripples produced by ion bombardment of binary compounds. *Journal of Physics D: Applied Physics*, 45(12):122001, 2012.

Deepti Pachauri, Christian Hinrichs, Moo K. Chung, Sterling C. Johnson, and Vikas Singh. Topology-based kernels with application to inference problems in Alzheimer's disease. *IEEE Transactions on Medical Imaging*, 30(10):1760–1770, 2011.

Hae-Sang Park and Chi-Hyuck Jun. A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications*, 36(2):3336–3341, 2009.

Daniel A. Pearson, R. Mark Bradley, Francis C. Motta, and Patrick D. Shipman. Producing nanodot arrays with improved hexagonal order by patterning surfaces before ion sputtering. *Phys. Rev. E*, 92:062401, Dec 2015.

Jose A. Perea and John Harer. Sliding windows and persistence: An application of topological methods to signal analysis. *Foundations of Computational Mathematics*, pages 1–40, 2013.

Jan Reininghaus, Stefan Huber, Ulrich Bauer, and Roland Kwitt. A stable multi-scale kernel for topological machine learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4741–4748, 2015.

Martin Rost and Joachim Krug. Anisotropic Kuramoto-Sivashinsky equation for surface growth and erosion. *Physical Review Letters*, 75:3894, 1995.

David Rouse, Adam Watkins, David Porter, John Harer, Paul Bendich, Nate Strawn, Elizabeth Munch, Jonathan DeSena, Jesse Clarke, Jeffrey Gilbert, Peter Chin, and Andrew Newman. Feature-aided multiple hypothesis tracking using topological and statistical behavior classifiers. In *SPIE Proceedings*, volume 9474, page 94740L, 2015.

Gurjeet Singh, Facundo Memoli, Tigran Ishkhanov, Guillermo Sapiro, Gunnar Carlsson, and Dario L. Ringach. Topological analysis of population activity in visual cortex. *Journal of Vision*, 8(8):11, 2008.

Chad M. Topaz, Lori Ziegelmeier, and Tom Halverson. Topological data analysis of biological aggregation models. *PloS One*, 10(5):e0126383, 2015.

Katharine Turner, Yuriy Mileyko, Sayan Mukherjee, and John Harer. Fréchet means for distributions of persistence diagrams. *Discrete & Computational Geometry*, 52(1):44–70, 2014.

Sara Kališnik Verovšek. Tropical coordinates on the space of persistence barcodes. *arXiv preprint arXiv:1604.00113*, 2016.

J. Villain. Continuum models of crystal growth from atomic beams with and without desorption. *J. Phys. I France*, 1:19–42, 1991.

Dietrich E. Wolf. Kinetic roughening of vicinal surfaces. *Physical Review Letters*, 67:1783, 1991.

Matthias Zeppelzauer, Bartosz Zieliński, Mateusz Juda, and Markus Seidl. Topological descriptors for 3d surface analysis. In *Computational Topology in Image Context: 6th International Workshop Proceedings*, pages 77–87, Marseille, France, 2016.

Li Zhang and Weida Zhou. On the sparseness of 1-norm support vector machines. *Neural Networks*, 23(3):373–385, 2010.

Ji Zhu, Saharon Rosset, Trevor Hastie, and Rob Tibshirani. 1-norm support vector machines. *Advances in neural information processing systems*, 16(1):49–56, 2004.

Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete & Computational Geometry*, 33(2):249–274, 2005.

Journal of Machine Learning Research 18 (2017) 1-57 Submitted 8/14; Revised 7/16; Published 3/17

Spectral Clustering Based on Local PCA

Ery Arias-Castro EARIASCA@UCSD.EDU
Department of Mathematics
University of California, San Diego
La Jolla, CA 92093, USA

Gilad Lerman LERMAN@UMN.EDU
Department of Mathematics
University of Minnesota, Twin Cities
Minneapolis, MN 55455, USA

Teng Zhang TENG.ZHANG@UCF.EDU
Department of Mathematics
University of Central Florida
Orlando, FL 32816, USA

Editor: Mikhail Belkin

Abstract

We propose a spectral clustering method based on local principal components analysis (PCA). After performing local PCA in selected neighborhoods, the algorithm builds a nearest neighbor graph weighted according to a discrepancy between the principal subspaces in the neighborhoods, and then applies spectral clustering. As opposed to standard spectral methods based solely on pairwise distances between points, our algorithm is able to resolve intersections. We establish theoretical guarantees for simpler variants within a prototypical mathematical framework for multi-manifold clustering, and evaluate our algorithm on various simulated data sets.

Keywords: multi-manifold clustering, spectral clustering, local principal component analysis, intersecting clusters

1. Introduction

The task of multi-manifold clustering, where the data are assumed to be located near surfaces embedded in Euclidean space, is relevant in a variety of applications. In cosmology, it arises as the extraction of galaxy clusters in the form of filaments (curves) and walls (surfaces) (Valdarnini, 2001; Martínez and Saar, 2002); in motion segmentation, moving objects tracked along different views form affine or algebraic surfaces (Ma et al., 2008; Fu et al., 2005; Vidal and Ma, 2006; Chen et al., 2009); this is also true in face recognition, in the context of images of faces in fixed pose under varying illumination conditions (Ho et al., 2003; Basri and Jacobs, 2003; Epstein et al., 1995).

We consider a stylized setting where the underlying surfaces are nonparametric in nature, with a particular emphasis on situations where the surfaces intersect. Specifically, we assume the surfaces are smooth, for otherwise the notion of continuation is potentially ill-posed. For example, without smoothness assumptions, an L-shaped cluster is indistinguishable from the union of two line-segments meeting at right angle.

©2017 Ery Arias-Castro, Gilad Lerman, and Teng Zhang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/14-318.html.

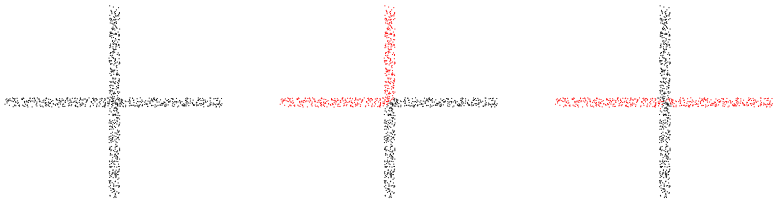


Figure 1: Two rectangular clusters intersecting at right angle. Left: the original data. Center: a typical output of the standard spectral clustering method of Ng et al. (2002), which is generally unable to resolve intersections. Right: a typical output of our method.

Spectral methods (Luxburg, 2007) are particularly suited for nonparametric settings, where the underlying clusters are usually far from convex, making standard methods like K-means irrelevant. However, a drawback of standard spectral approaches such as the well-known variant of Ng, Jordan, and Weiss (2002) is their inability to separate intersecting clusters. Indeed, consider the simplest situation where two straight clusters intersect at right angle, pictured in Figure 1 below. The algorithm of Ng et al. (2002) is based on pairwise affinities that are decreasing in the distances between data points, making it insensitive to smoothness and, therefore, intersections. And indeed, this algorithm typically fails to separate intersecting clusters, even in the easiest setting of Figure 1.

As argued in (Agarwal et al., 2005, 2006; Shashua et al., 2006), a multiway affinity is needed to capture complex structure in data (here, smoothness) beyond proximity attributes. For example, Chen and Lerman (2009a) use a flatness affinity in the context of *hybrid linear modeling*, where the surfaces are assumed to be affine subspaces, and subsequently extended to algebraic surfaces via the 'kernel trick' (Chen, Atev, and Lerman, 2009). Moving beyond parametric models, Arias-Castro, Chen, and Lerman (2011) introduce a localized measure of flatness.

Continuing this line of work, we suggest a spectral clustering method based on the estimation of the local linear structure (tangent bundle) via local principal component analysis (PCA). The idea of using local PCA combined with spectral clustering has precedents in the literature. In particular, our method is inspired by the work of Goldberg, Zhu, Singh, Xu, and Nowak (2009), where the authors develop a spectral clustering method within a semi-supervised learning framework. This approach is in the zeitgeist. While writing this paper, we became aware of two concurrent publications, by Wang, Jiang, Wu, and Zhou (2011) and by Gong, Zhao, and Medioni (2012), both proposing approaches very similar to ours.[1] We also mention the multiscale, spectral-flavored algorithm of Kushnir, Galun, and Brandt (2006), which is also based on local PCA. We comment on these spectral methods

1. The present paper was indeed developed in parallel with these two, first as a short version submitted to ICML in 2011.

in more detail later on. In fact, an early proposal also based on local PCA appears in the literature on subspace clustering (Fan et al., 2006)—although the need for localization is perhaps less intuitive in this setting.

The basic proposition of local PCA combined with spectral clustering has two main stages. The first one forms an affinity between a pair of data points that takes into account both their Euclidean distance and a measure of discrepancy between their tangent spaces. Each tangent space is estimated by PCA in a local neighborhood around each point. The second stage applies standard spectral clustering with this affinity. As a reality check, this relatively simple algorithm succeeds at separating the straight clusters in Figure 1. We tested our algorithm in more elaborate settings, some of them described in Section 4.

Other methods with a spectral component include those of Polito and Perona (2001) and Goh and Vidal (2007), which (roughly speaking) embed the points by a variant of LLE (Saul and Roweis, 2003) and then group the points by K-means clustering. There is also the method of Elhamifar and Vidal (2011), which chooses the neighborhood of each point by computing a sparse linear combination of the remaining points followed by an application of the spectral graph partitioning algorithm of Ng et al. (2002). Note that these methods work under the assumption that the surfaces do not intersect.

Besides spectral-type approaches to multi-manifold clustering, other methods appear in the literature. Among these, Gionis et al. (2005) and Haro et al. (2007) allow for intersecting surfaces but assume that they have different intrinsic dimension or density—and the proposed methodology is entirely based on such assumptions. We also mention the K-manifold method of Souvenir and Pless (2005), which propose an EM-type algorithm; and that of Guo et al. (2007), which propose to minimize an energy functional based on pairwise distances and local curvatures, leading to a combinatorial optimization.

Our contribution is the design and detailed study of a prototypical spectral clustering algorithm based on local PCA, tailored to settings where the underlying clusters come from sampling in the vicinity of smooth surfaces that may intersect. We endeavored to simplify the algorithm as much as possible without sacrificing performance. We provide theoretical results for simpler variants within a standard mathematical framework for multi-manifold clustering. To our knowledge, these are the first mathematically backed successes at the task of resolving intersections in the context of multi-manifold clustering, with the exception of (Arias-Castro et al., 2011), where the corresponding algorithm is shown to succeed at separating intersecting curves. The salient features of our algorithm are illustrated via numerical experiments.

The rest of the paper is organized as follows. In Section 2, we introduce our method in various variants. In Section 3, we analyze the simpler variants in a standard mathematical framework for multi-manifold learning. In Section 4, we perform some numerical experiments illustrating several features of our approach. In Section 5, we discuss possible extensions.

2. The Methodology

We introduce our algorithm and simpler variants that are later analyzed in a mathematical framework. We start with some review of the literature, zooming in on the most closely related publications.

2.1 Some Precedents

Using local PCA within a spectral clustering algorithm was implemented in four other publications we know of (Goldberg et al., 2009; Kushnir et al., 2006; Gong et al., 2012; Wang et al., 2011). As a first stage in their semi-supervised learning method, Goldberg, Zhu, Singh, Xu, and Nowak (2009) design a spectral clustering algorithm. The method starts by subsampling the data points, obtaining 'centers' in the following way. Draw $\boldsymbol{y}_1$ at random from the data and remove its ℓ-nearest neighbors from the data. Then repeat with the remaining data, obtaining centers $\boldsymbol{y}_1, \boldsymbol{y}_2, \dots$. Let $\boldsymbol{C}_i$ denote the sample covariance in the neighborhood of $\boldsymbol{y}_i$ made of its ℓ-nearest neighbors. An m-nearest-neighbor graph is then defined on the centers in terms of the Mahalanobis distances. Explicitly, the centers $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ are connected in the graph if $\boldsymbol{y}_j$ is among the m nearest neighbors of $\boldsymbol{y}_i$ in Mahalanobis distance

$$\|\boldsymbol{C}_i^{-1/2}(\boldsymbol{y}_i - \boldsymbol{y}_j)\|, \tag{1}$$

or vice-versa. The parameters ℓ and m are both chosen of order $\log n$. An existing edge between $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ is then weighted by $\exp(-H_{ij}^2/\eta^2)$, where H_{ij} denotes the Hellinger distance between the probability distributions $\mathcal{N}(\boldsymbol{0}, \boldsymbol{C}_i)$ and $\mathcal{N}(\boldsymbol{0}, \boldsymbol{C}_j)$. The spectral graph partitioning algorithm of Ng, Jordan, and Weiss (2002)—detailed in Algorithm 1—is then applied to the resulting affinity matrix, with some form of constrained K-means. We note that Goldberg et al. (2009) evaluate their method in the context of semi-supervised learning where the clustering routine is only required to return subclusters of actual clusters. In particular, the data points other than the centers are discarded. Note also that their evaluation is empirical.

Algorithm 1 Spectral Graph Partitioning (Ng, Jordan, and Weiss, 2002)

Input:
Affinity matrix $\boldsymbol{W} = (W_{ij})$, size of the partition K

Steps:
1: Compute $\boldsymbol{Z} = (Z_{ij})$ according to $Z_{ij} = W_{ij}/\sqrt{\Delta_i \Delta_j}$, with $\Delta_i := \sum_{j=1}^n W_{ij}$.
2: Extract the top K eigenvectors of $\boldsymbol{Z}$.
3: Renormalize each row of the resulting $n \times K$ matrix.
4: Apply K-means to the row vectors.

The algorithm proposed by Kushnir, Galun, and Brandt (2006) is multiscale and works by coarsening the neighborhood graph and computing sampling density and geometric information inferred along the way such as obtained via PCA in local neighborhoods. This bottom-up flow is then followed by a top-down pass, and the two are iterated a few times. The algorithm is too complex to be described in detail here, and probably too complex to be analyzed mathematically. The clustering methods of Goldberg et al. (2009) and ours can be seen as simpler variants that only go bottom up and coarsen the graph only once.

In the last stages of writing this paper, we learned of the works of Wang, Jiang, Wu, and Zhou (2011) and Gong, Zhao, and Medioni (2012), who propose algorithms very similar to our Algorithm 3 detailed below. Note that these publications do not provide any theoretical guarantees for their methods, which is one of our main contributions here.

2.2 Our Algorithms

We now describe our method and propose several variants. Our setting is standard: we observe data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^D$ that we assume were sampled in the vicinity of K smooth surfaces embedded in $\mathbb{R}^D$. The setting is formalized later in Section 3.1.

2.2.1 Connected Component Extraction: Comparing Local Covariances

We start with our simplest variant, which is also the most natural. The method depends on a neighborhood radius $r > 0$, a spatial scale parameter $\varepsilon > 0$ and a covariance (relative) scale $\eta > 0$. For a vector $\boldsymbol{x}$, $\|\boldsymbol{x}\|$ denotes its Euclidean norm, and for a (square) matrix $\boldsymbol{A}$, $\|\boldsymbol{A}\|$ denotes its spectral norm. For $n \in \mathbb{N}$, we denote by $[n]$ the set $\{1, \ldots, n\}$. Given a data set $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, for any point $\boldsymbol{x} \in \mathbb{R}^D$ and $r > 0$, define the neighborhood

$$N_r(\boldsymbol{x}) = \{\boldsymbol{x}_j : \|\boldsymbol{x} - \boldsymbol{x}_j\| \leq r\}.$$

Algorithm 2 Connected Component Extraction: Comparing Covariances

Input:

Data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$; neighborhood radius $r > 0$; spatial scale $\varepsilon > 0$, covariance scale $\eta > 0$.

Steps:

1: For each $i \in [n]$, compute the sample covariance matrix $\boldsymbol{C}_i$ of $N_r(\boldsymbol{x}_i)$.
2: Remove $\boldsymbol{x}_i$ when there is $\boldsymbol{x}_j$ such that $\|\boldsymbol{x}_j - \boldsymbol{x}_i\| \leq r$ and $\|\boldsymbol{C}_j - \boldsymbol{C}_i\| > \eta r^2$.
3: Compute the following affinities between data points:

$$W_{ij} = \mathbb{1}_{\{\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \leq \varepsilon\}} \cdot \mathbb{1}_{\{\|\boldsymbol{C}_i - \boldsymbol{C}_j\| \leq \eta r^2\}}. \tag{2}$$

4: Extract the connected components of the resulting graph.
5: Each point removed in Step 2 is grouped with the closest point that survived Step 2.

In words, the algorithm first computes local covariances (Step 1). It removes points that are believed to be very near an intersection (Step 2)—we elaborate on this below. With the remaining points, it creates an unweighted graph: the nodes of this graph are the data points and edges are formed between two nodes if both the distance between these nodes and the distance between the local covariance structures at these nodes are sufficiently small (Step 3). The connected components of the resulting graph are extracted (Step 4) and the points that survived Step 2 are labeled accordingly. Each point removed in Step 2 receives the label of its closest labeled point (Step 5).

In principle, the neighborhood size r is chosen just large enough that performing PCA in each neighborhood yields a reliable estimate of the local covariance structure. For this, the number of points inside the neighborhood needs to be large enough, which depends on the sample size n, the sampling density, intrinsic dimension of the surfaces and their surface area (Hausdorff measure), how far the points are from the surfaces (i.e., noise level), and the regularity of the surfaces. The spatial scale parameter ε depends on the sampling density and r. It is meant to be larger than r and needs to be large enough that a point has plenty

of points within distance ε, including some across an intersection, so each cluster is strongly connected. At the same time, ε needs to be small enough that a local linear approximation to the surfaces is a relevant feature of proximity. Its choice is rather similar to the choice of the scale parameter in standard spectral clustering (Ng et al., 2002; Zelnik-Manor and Perona, 2005). The covariance scale η needs to be large enough that centers from the same cluster and within distance ε of each other have local covariance matrices within distance ηr^2, but small enough that points from different clusters near their intersection have local covariance matrices separated by a distance substantially larger than ηr^2. This depends on the curvature of the surfaces and the incidence angle at the intersection of two (or more) surfaces. Note that a typical covariance matrix over a ball of radius r has norm of order r^2, which justifies using our choice of parametrization. In the mathematical framework we introduce later on, these parameters can be chosen automatically as done in (Arias-Castro et al., 2011), at least when the points are sampled exactly on the surfaces. We will not elaborate on that since in practice this does not inform our choice of parameters.

The rationale behind Step 2 is as follows. As we just discussed, the parameters need to be tuned so that points from the same cluster and within distance ε have local covariance matrices within distance ηr^2. Strictly speaking, this is true of points away from the boundary of the underlying surface. Hence, although $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in Step 2 are meant to be from different clusters, they could be from the same surface near its boundary. In the former situation, since they are near each other, in our model this will imply that they are close to an intersection. Therefore, roughly speaking, Step 2 removes points near an intersection, but also points near the boundaries of the underlying surfaces. The reason that we require $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ to be within distance r, as opposed to ε, is because otherwise removing the points in Step 2 would create a "gap" of ε near an intersection which then cannot be bridged in Steps 3-4. (Alternatively, one could replace r with $\xi \in (r, \varepsilon)$ in Step 2, but this would add this additional parameter ξ to the algorithm.) This step is in fact crucial as the local covariance varies smoothly along the intersection of two smooth surfaces.

Although this method works in simple situations like that of two intersecting segments (Figure 1), it is not meant to be practical. Indeed, extracting connected components is known to be sensitive to spurious points and therefore unstable. Furthermore, we found that comparing local covariance matrices as in affinity (2) tends to be less stable than comparing local projections as in affinity (3), which brings us to our next variant.

2.2.2 Connected Component Extraction: Comparing Local Projections

We present another variant also based on extracting the connected components of a neighborhood graph that compares orthogonal projections onto the largest principal directions. See Algorithm 3.

We note that the local intrinsic dimension is determined by thresholding the eigenvalues of the local covariance matrix, keeping the directions with eigenvalues within some range of the largest eigenvalue. The same strategy is used by Kushnir et al. (2006), but with a different threshold. The method is a hard version of what we implemented, which we describe in Algorithm 4.

We note that neither algorithm includes an intersection-removal step as Step 2 in Algorithm 2. The reason is that the algorithms work without such a step. Indeed, we show in

Algorithm 3 Connected Component Extraction: Comparing Projections

Input:

Data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$; neighborhood radius $r > 0$, spatial scale $\varepsilon > 0$, projection scale $\eta > 0$.

Steps:

1: For each $i \in [n]$, compute the sample covariance matrix $\boldsymbol{C}_i$ of $N_r(\boldsymbol{x}_i)$.

2: Compute the projection $\boldsymbol{Q}_i$ onto the eigenvectors of $\boldsymbol{C}_i$ with corresponding eigenvalue exceeding $\sqrt{\eta}\,\|\boldsymbol{C}_i\|$.

3: Compute the following affinities between data points:

$$W_{ij} = \mathbb{1}_{\{\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \le \varepsilon\}} \cdot \mathbb{1}_{\{\|\boldsymbol{Q}_i - \boldsymbol{Q}_j\| \le \eta\}}. \tag{3}$$

4: Extract the connected components of the resulting graph.

Theorem 1 that Algorithm 3 is able to separate the clusters—the only drawback is that it may possibly treat the intersection of two surfaces as a cluster. And we show via numerical experiments in Section 4 that Algorithm 4 performs well in a number of situations.

2.2.3 Covariances or Projections?

In our numerical experiments, we tried working both directly with covariance matrices as in (2) and with projections as in (3). Note that in our experiments we used spectral graph partitioning with soft versions of these affinities, as described in Section 2.2.4. We found working with projections to be more reliable. The problem comes, in part, from boundaries. When a surface has a boundary, local covariances over neighborhoods that overlap with the boundary are quite different from local covariances over nearby neighborhoods that do not touch the boundary. Consider the example of two segments, S_1 and S_2, intersecting at an angle of $\theta \in (0, \pi/2)$ at their middle point, specifically

$$S_1 = [-1, 1] \times \{0\}, \qquad S_2 = \{(x, x\tan\theta) : x \in [-\cos\theta, \cos\theta]\}.$$

Assume there is no noise and that the sampling is uniform. Assume $r \in (0, \frac{1}{2}\sin\theta)$ so that the disc centered at $\boldsymbol{x}_1 := (1/2, 0)$ does not intersect S_2, and the disc centered at $\boldsymbol{x}_2 := (\frac{1}{2}\cos\theta, \frac{1}{2}\sin\theta)$ does not intersect S_1. Let $\boldsymbol{x}_0 = (1, 0)$. For $\boldsymbol{x} \in S_1 \cup S_2$, let $\boldsymbol{C}_{\boldsymbol{x}}$ denote the local covariance at $\boldsymbol{x}$ over a ball of radius $r < \frac{1}{2}\sin\theta$, so that the neighborhoods of $\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2$ are "pure". The situation is described in Figure 2.

Simple calculations yield

$$\boldsymbol{C}_{\boldsymbol{x}_0} = \frac{r^2}{12}\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \boldsymbol{C}_{\boldsymbol{x}_1} = \frac{r^2}{3}\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \boldsymbol{C}_{\boldsymbol{x}_2} = \frac{r^2}{3}\begin{pmatrix} \cos^2\theta & \sin(\theta)\cos(\theta) \\ \sin(\theta)\cos(\theta) & \sin^2\theta \end{pmatrix},$$

so that

$$\|\boldsymbol{C}_{\boldsymbol{x}_0} - \boldsymbol{C}_{\boldsymbol{x}_1}\| = \frac{r^2}{4}, \quad \|\boldsymbol{C}_{\boldsymbol{x}_1} - \boldsymbol{C}_{\boldsymbol{x}_2}\| = \frac{r^2}{3}\sin\theta.$$

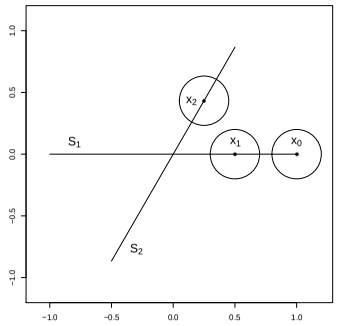


Figure 2: Two segments intersecting. The local covariances (within the disc neighborhoods drawn) at $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are closer than the local covariances at $\boldsymbol{x}_1$ and $\boldsymbol{x}_0$, even though $\boldsymbol{x}_0$ and $\boldsymbol{x}_1$ are on the same segment.

Therefore, when $\sin\theta \le \frac{3}{4}$ (roughly, $\theta \le 48^o$), the local covariances at $\boldsymbol{x}_0, \boldsymbol{x}_1 \in S_1$ are farther (in operator norm) than those at $\boldsymbol{x}_1 \in S_1$ and $\boldsymbol{x}_2 \in S_2$. As for projections, however,

$$\boldsymbol{Q}_{\boldsymbol{x}_0} = \boldsymbol{Q}_{\boldsymbol{x}_1} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad \boldsymbol{Q}_{\boldsymbol{x}_2} = \begin{pmatrix} \cos^2\theta & \sin(\theta)\cos(\theta) \\ \sin(\theta)\cos(\theta) & \sin^2\theta \end{pmatrix},$$

so that

$$\|\boldsymbol{Q}_{\boldsymbol{x}_0} - \boldsymbol{Q}_{\boldsymbol{x}_1}\| = 0, \quad \|\boldsymbol{Q}_{\boldsymbol{x}_1} - \boldsymbol{Q}_{\boldsymbol{x}_2}\| = \sqrt{2}\sin\theta.$$

While in theory points within distance r from the boundary account for a small portion of the sample, in practice this is not the case, at least not with the sample sizes that we are able to rapidly process. In fact, we find that spectral graph partitioning is challenged by having points near the boundary that are far (in affinity) from nearby points from the same cluster. This may explain why the (soft version of) affinity (3) yields better results than the (soft version of) affinity (2) in our experiments.

2.2.4 Spectral Clustering Based on Local PCA

The following variant is more robust in practice and is the algorithm we actually implemented. The method assumes that the surfaces are of same dimension d and that there are K of them, with both parameters K and d known.

We note that $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{n_0}$ forms an r-packing of the data. The underlying rationale for this coarsening is justified in (Goldberg et al., 2009) by the fact that the covariance matrices, and also the top principal directions, change smoothly with the location of the neighborhood, so that without subsampling these characteristics would not help detect the abrupt event of an intersection. The affinity (4) is of course a soft version of (3).

Algorithm 4 Spectral Clustering Based on Local PCA

Input:

Data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$; neighborhood radius $r > 0$; spatial scale $\varepsilon > 0$, projection scale $\eta > 0$; intrinsic dimension d; number of clusters K.

Steps:

0: Pick one point $\boldsymbol{y}_1$ at random from the data. Pick another point $\boldsymbol{y}_2$ among the data points not included in $N_r(\boldsymbol{y}_1)$, and repeat the process, selecting centers $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{n_0}$.

1: For each $i = 1, \ldots, n_0$, compute the sample covariance matrix $\boldsymbol{C}_i$ of $N_r(\boldsymbol{y}_i)$. Let $\boldsymbol{Q}_i$ denote the orthogonal projection onto the space spanned by the top d eigenvectors of $\boldsymbol{C}_i$.

2: Compute the following affinities between center pairs:

$$W_{ij} = \exp\left(-\frac{\|\boldsymbol{y}_i - \boldsymbol{y}_j\|^2}{\varepsilon^2}\right) \cdot \exp\left(-\frac{\|\boldsymbol{Q}_i - \boldsymbol{Q}_j\|^2}{\eta^2}\right). \tag{4}$$

3: Apply spectral graph partitioning (Algorithm 1) to $\boldsymbol{W}$.

4: The data points are clustered according to the closest center in Euclidean distance.

2.2.5 Comparison with Closely Related Methods

We highlight some differences with the other proposals in the literature. We first compare our approach to that of Goldberg et al. (2009), which was our main inspiration.

- *Neighborhoods.* Comparing with Goldberg et al. (2009), we define neighborhoods over r-balls instead of ℓ-nearest neighbors, and connect points over ε-balls instead of m-nearest neighbors. This choice is for convenience, as these ways are in fact essentially equivalent when the sampling density is fairly uniform. This is elaborated at length in (Maier et al., 2009; Brito et al., 1997; Arias-Castro, 2011).

- *Mahalanobis distances.* Goldberg et al. (2009) use Mahalanobis distances (1) between centers. In our version, we could for example replace the Euclidean distance $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|$ in the affinity (2) with the average Mahalanobis distance

$$\|\boldsymbol{C}_i^{-1/2}(\boldsymbol{x}_i - \boldsymbol{x}_j)\| + \|\boldsymbol{C}_j^{-1/2}(\boldsymbol{x}_j - \boldsymbol{x}_i)\|.$$

  We actually tried this and found that the algorithm was less stable, particularly under low noise. Introducing a regularization in this distance—which requires the introduction of another parameter—solves this problem, at least partially.

  That said, using Mahalanobis distances makes the procedure less sensitive to the choice of ε, in that neighborhoods may include points from different clusters. Think of two parallel line segments separated by a distance of δ, and assume there is no noise, so the points are sampled exactly from these segments. Assuming an infinite sample size, the local covariance is the same everywhere so that points within distance ε are connected by the affinity (2). Hence, Algorithm 2 requires that $\varepsilon < \delta$. In terms of Mahalanobis distances, points on different segments are infinitely separated, so a version based on these distances would work with any $\varepsilon > 0$. In the case of curved surfaces and/or noise, the situation is similar, though not as evident. Even then, the gain in performance is not obvious since we only require that ε be slightly larger in order of magnitude than r.

- *Hellinger distances.* As we mentioned earlier, Goldberg et al. (2009) use Hellinger distances of the probability distributions $\mathcal{N}(\boldsymbol{0}, \boldsymbol{C}_i)$ and $\mathcal{N}(\boldsymbol{0}, \boldsymbol{C}_j)$ to compare covariance matrices, specifically

$$\left(1 - 2^{D/2}\frac{\det(\boldsymbol{C}_i\boldsymbol{C}_j)^{1/4}}{\det(\boldsymbol{C}_i + \boldsymbol{C}_j)^{1/2}}\right)^{1/2}, \tag{5}$$

  if $\boldsymbol{C}_i$ and $\boldsymbol{C}_j$ are full-rank. While using these distances or the Frobenius distances makes little difference in practice, we find it easier to work with the latter when it comes to proving theoretical guarantees. Moreover, it seems more natural to assume a uniform sampling distribution in each neighborhood rather than a normal distribution, so that using the more sophisticated similarity (5) does not seem justified.

- *K-means.* We use K-means++ for a good initialization. We found that the more sophisticated size-constrained K-means (Bradley et al., 2000) used in (Goldberg et al., 2009) did not improve the clustering results.

As we mentioned above, our work was developed in parallel to that of Wang et al. (2011) and Gong et al. (2012). We highlight some differences. First, there is no subsampling, but rather, the local tangent space is estimated at each data point $\boldsymbol{x}_i$. Wang et al. (2011) fit a mixture of d-dimensional affine subspaces to the data using MPPCA (Tipping and Bishop, 1999), which is then used to estimate the tangent subspaces at each data point. Gong et al. (2012) develop some sort of robust local PCA. While Wang et al. (2011) assume all surfaces are of same dimension known to the user, Gong et al. (2012) estimate that locally by looking at the largest gap in the spectrum of estimated local covariance matrix. This is similar in spirit to what is done in Step 2 of Algorithm 3, but we did not include this step in Algorithm 4 because we did not find it reliable in practice. We also tried estimating the local dimensionality using the method of Little et al. (2009), but this failed in the most complex cases.

Wang et al. (2011) use a nearest-neighbor graph and their affinity is defined as

$$W_{ij} = \Delta_{ij} \cdot \left(\prod_{s=1}^{d} \cos\theta_s(i,j)\right)^{\alpha},$$

where $\Delta_{ij} = 1$ if $\boldsymbol{x}_i$ is among the ℓ-nearest neighbors of $\boldsymbol{x}_j$, or vice versa, while $\Delta_{ij} = 0$ otherwise; $\theta_1(i,j) \geq \cdots \geq \theta_d(i,j)$ are the principal (a.k.a., canonical) angles (Stewart and Sun, 1990) between the estimated tangent subspaces at $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. ℓ and α are parameters of the method. Gong et al. (2012) define an affinity that incorporates the self-tuning method of Zelnik-Manor and Perona (2005); in our notation, their affinity is

$$\exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{\varepsilon_i\varepsilon_j}\right) \cdot \exp\left(-\frac{(\sin^{-1}(\|\boldsymbol{Q}_i - \boldsymbol{Q}_j\|))^2}{\eta^2\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/(\varepsilon_i\varepsilon_j)}\right).$$

where ε_i is the distance from $\boldsymbol{x}_i$ to its ℓ-nearest neighbor. ℓ is a parameter.

Although we do not analyze their respective ways of estimating the tangent subspaces, our analysis provides essential insights into their methods, and for that matter, any other method built on spectral clustering based on tangent subspace comparisons.

3. Mathematical Analysis

While the analysis of Algorithm 4 seems within reach, there are some complications due to the fact that points near the intersection may form a cluster of their own—we were not able to discard this possibility. Instead, we study the simpler variants described in Algorithm 2 and Algorithm 3. Even then, the arguments are rather complex and interestingly involved. The theoretical guarantees that we obtain for these variants are stated in Theorem 1 and proved in Section 6. We comment on the analysis of Algorithm 4 right after that. We note that there are very few theoretical results on resolving intersecting manifolds—in fact, we are only aware of (Arias-Castro et al., 2011) (under severe restrictions on the dimension of the intersection). Some such results have been established for a number of methods for subspace clustering (affine surfaces), for example, in (Chen and Lerman, 2009b; Soltanolkotabi and Candès, 2012; Soltanolkotabi et al., 2014; Wang and Xu, 2013; Heckel and Bölcskei, 2013; Tsakiris and Vidal, 2015; Ma et al., 2008).

The generative model we assume is a natural mathematical framework for multi-manifold learning where points are sampled in the vicinity of smooth surfaces embedded in Euclidean space. For concreteness and ease of exposition, we focus on the situation where two surfaces (i.e., $K = 2$) of same dimension $1 \leq d \leq D$ intersect. This special situation already contains all the geometric intricacies of separating intersecting clusters. On the one hand, clusters of different intrinsic dimension may be separated with an accurate estimation of the local intrinsic dimension without further geometry involved (Haro et al., 2007). On the other hand, more complex intersections (3-way and higher) complicate the situation without offering truly new challenges. For simplicity of exposition, we assume that the surfaces are submanifolds without boundary, though it will be clear from the analysis (and the experiments) that the method can handle surfaces with (smooth) boundaries that may self-intersect. We discuss other possible extensions in Section 5.

Within that framework, we show that Algorithm 2 and Algorithm 3 are able to identify the clusters accurately except for points near the intersection. Specifically, with high probability with respect to the sampling distribution, Algorithm 2 divides the data points into two groups such that, except for points within distance $C\varepsilon$ of the intersection, all points from the first cluster are in one group and all points from the second cluster are in the other group. The constant C depends on the surfaces, including their curvatures, separation between them and intersection angle. The situation for Algorithm 3 is more complex, as it may return more than two clusters, but most of the two clusters (again, away from the intersection) are in separate connected components.

3.1 Generative Model

Each surface we consider is a connected, C^2 and compact submanifold without boundary and of dimension d embedded in $\mathbb{R}^D$. Any such surface has a positive reach, which is what we use to quantify smoothness. The notion of reach was introduced by Federer (1959).

Intuitively, a surface has reach exceeding r if and only if one can roll a ball of radius r on the surface without obstruction (Walther, 1997; Cuevas et al., 2012). Formally, for $\boldsymbol{x} \in \mathbb{R}^D$ and $S \subset \mathbb{R}^D$, let

$$\operatorname{dist}(\boldsymbol{x}, S) = \inf_{\boldsymbol{s} \in S} \|\boldsymbol{x} - \boldsymbol{s}\|,$$

and

$$B(S, r) = \{\boldsymbol{x} : \operatorname{dist}(\boldsymbol{x}, S) < r\},$$

which is often called the r-tubular neighborhood (or r-neighborhood) of S. The reach of S is the supremum over $r > 0$ such that, for each $\boldsymbol{x} \in B(S, r)$, there is a unique point in S nearest $\boldsymbol{x}$. It is well-known that, for C^2 submanifolds, the reach bounds the radius of curvature from below (Federer, 1959, Lem. 4.17). For submanifolds without boundaries, the reach coincides with the condition number introduced in (Niyogi et al., 2008).

When two surfaces S_1 and S_2 intersect, meaning $S_1 \cap S_2 \neq \emptyset$, we define their incidence angle as

$$\theta(S_1, S_2) := \inf \left\{ \theta_{\min}(T_{S_1}(\boldsymbol{s}), T_{S_2}(\boldsymbol{s})) : \boldsymbol{s} \in S_1 \cap S_2 \right\}, \tag{6}$$

where $T_S(\boldsymbol{s})$ denote the tangent subspace of submanifold S at point $\boldsymbol{s} \in S$, and $\theta_{\min}(T_1, T_2)$ is the smallest *nonzero* principal (a.k.a., canonical) angle between subspaces T_1 and T_2 (Stewart and Sun, 1990).

The clusters are generated as follows. Each data point $\boldsymbol{x}_i$ is drawn according to

$$\boldsymbol{x}_i = \boldsymbol{s}_i + \boldsymbol{z}_i, \tag{7}$$

where $\boldsymbol{s}_i$ is drawn from the uniform distribution over $S_1 \cup S_2$ and $\boldsymbol{z}_i$ is an additive noise term satisfying $\|\boldsymbol{z}_i\| \leq \tau$—thus τ represents the noise or jitter level. When $\tau = 0$ the points are sampled exactly on the surfaces. We assume the points are sampled independently of each other. We let

$$I_k = \{i : \boldsymbol{s}_i \in S_k\},$$

and the goal is to recover the groups I_1 and I_2, up to some errors.

3.2 Performance Guarantees

We state some performance guarantees for Algorithm 2 and Algorithm 3.

Theorem 1. *Consider two connected, compact, twice continuously differentiable submanifolds without boundary, of same dimension d, intersecting at a strictly positive angle, with the intersection set having strictly positive reach. Assume the parameters are set so that*

$$\tau \leq r\eta/C, \quad r \leq \varepsilon/C, \quad \varepsilon \leq \eta/C, \quad \eta \leq 1/C, \tag{8}$$

for a large-enough constant $C \geq 1$ that depends on the configuration. Then with probability at least $1 - Cn \exp\left[-nr^d\eta^2/C\right]$:

- *Algorithm 2 returns exactly two groups such that two points from different clusters are not grouped together unless one of them is within distance Cr from the intersection.*
- *Algorithm 3 returns at least two groups, and such that two points from different clusters are not grouped together unless one of them is within distance Cr from the intersection.*

Thus, as long as, (8) is satisfied, the algorithms have the above properties with high probability when $r^d\eta^2 \geq C' \log n/n$, where $C' > C$ is fixed. In particular, we may choose $\eta \asymp 1$ and $\varepsilon \asymp r \asymp \tau \vee (\log(n)/n)^{1/d}$, which lightens the computational burden.

We note that, while the constant $C > 0$ does not depend on the sample size n, it depends in somewhat complicated ways on characteristics of the surfaces and their position relative to each other, such as their reach and intersection angle, but also aspects that are harder to quantify, like their separation away from their intersection. We note, however, that it behaves as expected: C is indeed decreasing in the reach and the intersection angle, and increasing in the intrinsic dimensions of the surfaces, for example.

The algorithms may make clustering mistakes within distance Cr of the intersection, where $Cr \asymp \tau \vee (\log(n)/n)^{1/d}$ with the choice of parameters just described. Whether this is optimal in the nonparametric setting that we consider—for example, in a minimax sense—we do not know.

We now comment on the challenge of proving a similar result for Algorithm 4. This algorithm relies on knowledge of the intrinsic dimension of the surfaces d and the number of clusters (here $K = 2$), but these may be estimated as in (Arias-Castro et al., 2011), at least in theory, so we assume these parameters are known. The subsampling done in Step 0 does not pose any problem whatsoever, since the centers are well-spread when the points themselves are. The difficulty resides in the application of the spectral graph partitioning, Algorithm 1. If we were to include the intersection-removal step (Step 2 of Algorithm 2) before applying spectral graph partitioning, then a simple adaptation of arguments in (Arias-Castro, 2011) would suffice. The real difficulty, and potential pitfall of the method in this framework (without the intersection-removal step), is that the points near the intersection may form their own cluster. For example, in the simplest case of two affine surfaces intersecting at a positive angle and no sampling noise, the projection matrix at a point near the intersection—meaning a point whose r-ball contains a substantial piece of both surfaces—would be the projection matrix onto $S_1 + S_2$ seen as a linear subspace. We were not able to discard this possibility, although we do not observe this happening in practice. A possible remedy is to constrain the K-means part to only return large-enough clusters. However, a proper analysis of this would require a substantial amount of additional work and we did not engage seriously in this pursuit.

4. Numerical Experiments

4.1 Some Illustrative Examples

We started by applying our method[2] on a few artificial examples to illustrate the theory. As we argued earlier, the methods of Wang et al. (2011) and Gong et al. (2012) are quite similar to ours, and we encourage the reader to also look at the numerical experiments they performed. Our numerical experiments should be regarded as a proof of concept, only here to show that our method can be implemented and works on some stylized examples.

In all experiments, the number of clusters K and the dimension of the manifolds d are assumed known. We choose the spatial scale ε and the projection scale η automatically as

2. The code is available online at `https://math.cos.ucf.edu/tengz`.

follows: we let

$$\varepsilon = \max_{1 \leq i \leq n_0} \min_{j \neq i} \|\boldsymbol{y}_i - \boldsymbol{y}_j\|, \tag{9}$$

and

$$\eta = \operatorname*{median}_{(i,j):\|\boldsymbol{y}_i - \boldsymbol{y}_j\| < \varepsilon} \|\boldsymbol{Q}_i - \boldsymbol{Q}_j\|.$$

Here, we implicitly assume that the union of all the underlying surfaces forms a connected set. In that case, the idea behind choosing ε as in (9) is that we want the ε-graph on the centers $\boldsymbol{y}_1, \dots, \boldsymbol{y}_n$ to be connected. Then η is chosen so that a center $\boldsymbol{y}_i$ remains connected in the (ε, η)-graph to most of its neighbors in the ε-graph.

The neighborhood radius r is chosen by hand for each situation. Although we do not know how to choose r automatically, there are some general ad hoc guidelines. When r is too large, the local linear approximation to the underlying surfaces may not hold in neighborhoods of radius r, resulting in local PCA becoming inappropriate. When r is too small, there might not be enough points in a neighborhood of radius r to accurately estimate the local tangent subspace to a given surface at that location, resulting in local PCA becoming inaccurate. From a computational point of view, the smaller r, the larger the number of neighborhoods and the heavier the computations, particularly at the level of spectral graph partitioning. In our numerical experiments, we find that our algorithm is more sensitive to the choice of r when the clustering problem is more difficult. We note that automatic choice of tuning parameters remains a challenge in clustering, and machine learning at large, especially when no labels are available whatsoever. See (Zelnik-Manor and Perona, 2005; Zhang et al., 2012; Little et al., 2009; Kaslovsky and Meyer, 2011).

Since the algorithm is randomized (see Step 0 in Algorithm 4) we repeat each simulation 100 times and report the median misclustering rate and number of times where the misclustering rate is smaller than 5%, 10%, and 15%.

We first run Algorithm 4 on several artificial data sets, which are demonstrated in the LHS of Figures 3 and 4. Table 1 reports the local radius r used for each data set (R is the global radius of each data set), and the statistics for misclustering rates. Typical clustering results are demonstrated in the RHS of Figures 3 and 4. It is evident that Algorithm 4 performs well in these simulations.

data set	r	median misclustering rate	5%	10%	15%
Three curves	0.02 ($0.034R$)	4.16%	76	89	89
Self-intersecting curves	0.1 ($0.017R$)	1.16%	85	85	86
Two spheres	0.2 ($0.059R$)	3.98%	100	100	100
Möbius strips	0.1 ($0.028R$)	2.22%	85	86	88
Monkey saddle	0.1 ($0.069R$)	9.73%	0	67	97
Paraboloids	0.07 ($0.048R$)	10.42%	0	12	91

Table 1: Choices for r and misclustering statistics for the artificial data sets demonstrated in Figures 3 and 4. The statistics are based on 100 repeats and include the median misclustering rate and number of repeats where the misclustering rate is smaller than 5%, 10% and 15%.

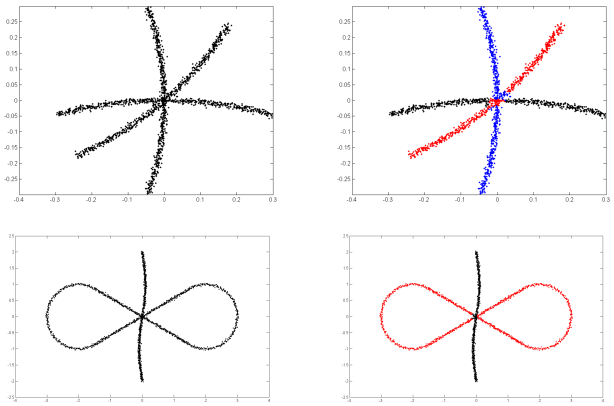


Figure 3: Performance of Algorithm 4 on data sets "Three curves" and "Self-intersecting curves". Left column is the input data sets, and right column demonstrates the typical clustering.

In another simulation, we show the dependence of the success of our algorithm on the intersecting angle between curves in Table 2 and Figure 5. Here, we fix two curves intersecting at a point, and gradually decrease the intersection angle by rotating one of them while holding the other one fixed. The angles are $\pi/2$, $\pi/4$, $\pi/6$ and $\pi/8$. From the table we can see that our algorithm performs well when the angle is $\pi/4$, but the performance deteriorates as the angle becomes smaller, and the algorithm almost always fails when the angle is $\pi/8$.

Intersecting angle	r	median misclustering rate	5%	10%	15%
$\pi/2$	0.02 (0.034R)	2.08%	98	98	98
$\pi/4$	0.02 (0.034R)	3.33%	92	94	94
$\pi/6$	0.02 (0.034R)	5.53%	32	59	59
$\pi/8$	0.02 (0.033R)	27.87%	0	2	2

Table 2: Choices for r and misclustering statistics for the instances of two intersecting curves demonstrated in Figure 5. The statistics are based on 100 repeats and include the median misclustering rate and number of repeats where the misclustering rate is smaller than 5%, 10% and 15%.

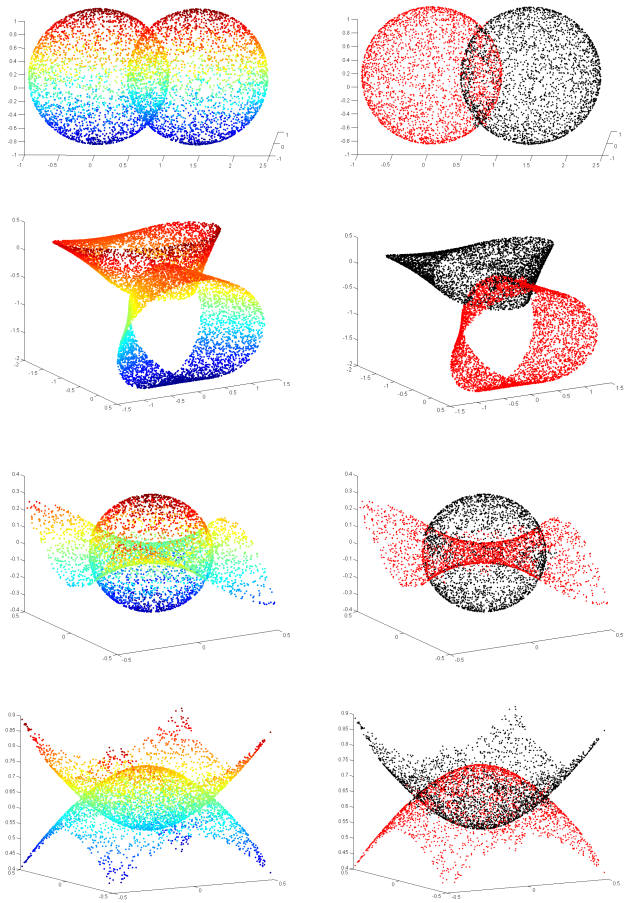


Figure 4: Performance of Algorithm 4 on data sets "Two spheres", "Möbius strips", "Monkey saddle" and "Paraboloids". Left column is the input data sets, and right column demonstrates the typical clustering.

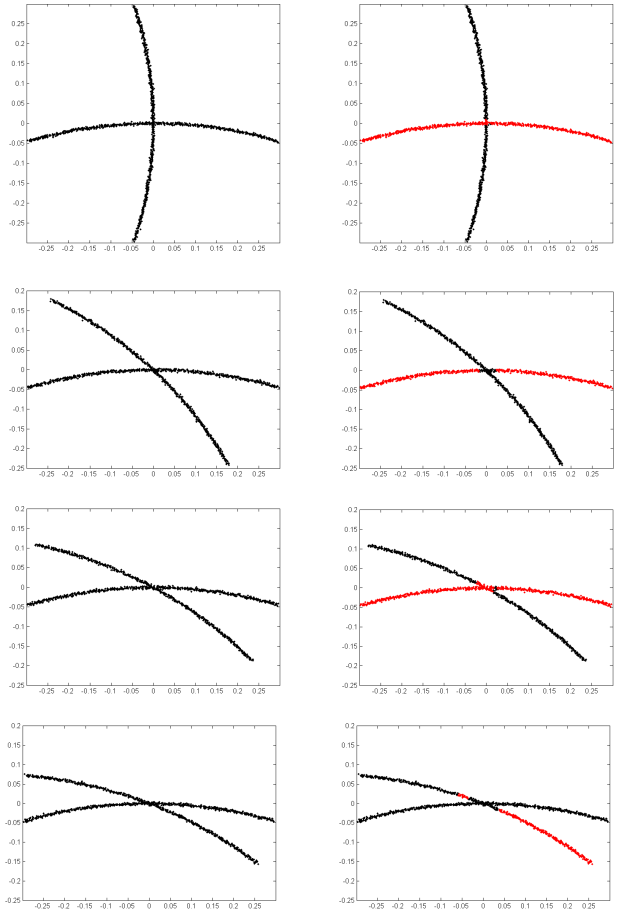


Figure 5: Performance of Algorithm 4 on two curves intersecting at angles $\frac{\pi}{2}$, $\frac{\pi}{4}$, $\frac{\pi}{6}$, $\frac{\pi}{8}$.

4.2 Comparison with Other Multi-manifold Clustering Algorithms

In this section, we compare our algorithm with several existing algorithms on multi-manifold clustering. While many algorithms have been proposed, we focus on the methods based on spectral clustering, including Sparse Manifold Clustering and Embedding (SMCE) (Elhamifar and Vidal, 2011) and Local Linear Embedding of (Polito and Perona, 2001; Goh and Vidal, 2007). Compared to these methods, a major difference of Algorithm 4 is the size of the affinity matrix $\boldsymbol{W}$: SMCE and LLE each creates an $n \times n$ affinity matrix while our method creates a smaller affinity matrix of size $n_0 \times n_0$, based on the centers chosen in step 0 of Algorithm 4. This difference enables our algorithm to handle large data sets such as $n > 10^4$, while these other two methods are computationally expensive due to eigenvalue decomposition of the $n \times n$ affinity matrix. In order to make a fair comparison, we will run simulations and experiments on small data sets. We modified our algorithm to make it more competitive in such a setting: we modify Steps 0 and 1 in Algorithm 4 slightly and use all data points $\{\boldsymbol{x}_i\}_{i=1}^n$ as centers, that is, $\boldsymbol{y}_i = \boldsymbol{x}_i$ for all $1 \le i \le n$. The motivation is that, when there is no computational constraint on the eigenvalue decomposition (due to small n), we may improve Algorithm 4 by constructing a larger affinity matrix by including all points as centers. The resulting algorithm is summarized in Algorithm 5.

Algorithm 5 Spectral Clustering Based on Local PCA (for small data sets)

Input:

Data points $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$; neighborhood size $N > 0$; spatial scale $\varepsilon > 0$, projection scale $\eta > 0$; intrinsic dimension d; number of clusters K.

Steps:

1: For each $i = 1, \dots, n$, compute the sample covariance matrix $\boldsymbol{C}_i$ of from the N nearest neighbors of $\boldsymbol{x}_i$. Let $\boldsymbol{Q}_i$ denote the orthogonal projection onto the space spanned by the top d eigenvectors of $\boldsymbol{C}_i$.

2: Compute the following affinities between n data points:

$$W_{ij} = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{\varepsilon^2}\right) \cdot \exp\left(-\frac{\|\boldsymbol{Q}_i - \boldsymbol{Q}_j\|^2}{\eta^2}\right).$$

3: Apply spectral graph partitioning (Algorithm 1) to $\boldsymbol{W}$.

First we test the algorithms on a simulated data set of two curves, which is subsampled from the first data set in Figure 5 with 300 data points. We plot the clustering result from Algorithm 5, SMCE, LLE in Figure 6. For Algorithm 5, K is set to be 10. For SMCE[3], $\lambda = 10$ and $L = 60$, and we remark that similar results are obtained for a wide range of parameters. For LLE, we follow the implementation in (Polito and Perona, 2001), use 10-nearest neighbors to embed the data set into $\mathbb{R}^2$ and run K-means on the embedded data set. It is clear from the figure that Algorithm 5 resolves the problem of intersection well

3. In (Elhamifar and Vidal, 2011), λ is the ℓ_1-penalty parameter and at each point the optimization is restricted to the L nearest neighbors.

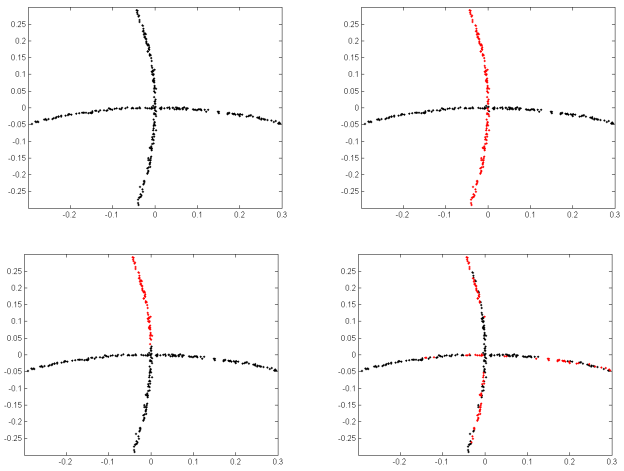


Figure 6: Performance of Algorithm 5 on two intersecting curves. Left top: the input data set. Right top: the clustering result by Algorithm 5. Left bottom: the clustering result by SMCE. Right bottom: the clustering result by LLE.

by using the affinity from estimated local subspaces, while SMCE and LLE tend to give a larger affinity between nearby data points and have difficulties in handling intersection.

Next, we run experiments on the Extended Yale Face Database B (Lee et al., 2005), with the goal of clustering face images of two different subjects. This data set contains face images from 39 subjects, and each subject has 64 images of 192 pixels under varying lightening conditions. In our experiments, we found that the images of a person in this database lie roughly in a 4-dimensional subspace. We preprocess the data set by applying PCA and reducing the dimension to 8. We also "normalize" the covariance of the data set when performing dimension reduction, such that the projected data set has a unit covariance. We record the misclustering rates of Algorithm 5, SMCE and LLE in Table 3. For SMCE, we follow (Elhamifar and Vidal, 2011) by setting $\lambda = 10$ and we let $L = 30$. For Algorithm 5, we let the neighborhood size be 40. From the table, we can see that the two methods perform similarly.

subjects	[1,2]	[1,3]	[1,4]	[1,5]	[1,6]	[1,7]	[1,8]	[1,9]
Local PCA	8.59%	11.72%	10.94%	4.69%	8.59%	7.81%	5.47%	7.03%
SMCE	8.59%	11.72%	8.59%	0.00%	4.69%	8.59%	9.38%	4.69%

Table 3: Some misclustering rates for the Extended Yale Face Database B.

5. Discussion

We distilled the ideas of Goldberg et al. (2009) and of Kushnir et al. (2006) to cluster points sampled near smooth surfaces. The key ingredient is the use of local PCA to learn about the local spread and orientation of the data, so as to use that information in an affinity when building a neighborhood graph.

In a typical stylized setting for multi-manifold clustering, we established performance bounds for the simple variants described in Algorithm 2 and Algorithm 3, which essentially consist of connecting points that are close in space and orientation, and then extracting the connected components of the resulting graph. Both are shown to resolve general intersections as long as the incidence angle is strictly positive and the parameters are carefully chosen. As is commonly the case in such analyses, our setting can be generalized to other sampling schemes, to multiple intersections, to some features of the surfaces changing with the sample size, and so on, in the spirit of (Arias-Castro et al., 2011; Arias-Castro, 2011; Chen and Lerman, 2009b). We chose to simplify the setup as much as possible while retaining the essential features that make resolving intersecting clusters challenging. The resulting arguments are nevertheless rich enough to satisfy the mathematically thirsty reader. Whether the conditions required in Theorem 1 are optimal in some sense is an interesting and challenging open question for future research. Note that very few optimality results exist for manifold clustering; see (Arias-Castro, 2011) for an example.

We implemented a spectral version of Algorithm 3, described in Algorithm 4, that assumes the intrinsic dimensionality and the number of clusters are known. The resulting approach is very similar to what is offered by Wang et al. (2011) and Gong et al. (2012), although it was developed independently of these works. Algorithm 4 is shown to perform well in some simulated experiments, although it is somewhat sensitive to the choice of parameters. This is the case of all other methods for multi-manifold clustering we know of and choosing the parameters automatically remains an open challenge in the field.

6. Proofs

We start with some additional notation. The ambient space is $\mathbb{R}^D$ unless noted otherwise. For a vector $\boldsymbol{v} \in \mathbb{R}^D$, $\|\boldsymbol{v}\|$ denotes its Euclidean norm and for a real matrix $\boldsymbol{M} \in \mathbb{R}^{D\times D}$, $\|\boldsymbol{M}\|$ denotes the corresponding operator norm. For a point $\boldsymbol{x} \in \mathbb{R}^D$ and $r > 0$, $B(\boldsymbol{x}, r)$ denotes the open ball of center $\boldsymbol{x}$ and radius r, i.e., $B(\boldsymbol{x}, r) = \{\boldsymbol{y} \in \mathbb{R}^D : \|\boldsymbol{y} - \boldsymbol{x}\| < r\}$. For a set S and a point $\boldsymbol{x}$, define $\operatorname{dist}(\boldsymbol{x}, S) = \inf\{\|\boldsymbol{x} - \boldsymbol{y}\| : \boldsymbol{y} \in S\}$. For two points $\boldsymbol{a}, \boldsymbol{b}$ in the same Euclidean space, $\boldsymbol{b} - \boldsymbol{a}$ denotes the vector moving $\boldsymbol{a}$ to $\boldsymbol{b}$. For a point $\boldsymbol{a}$ and a vector $\boldsymbol{v}$ in the same Euclidean space, $\boldsymbol{a} + \boldsymbol{v}$ denotes the translate of $\boldsymbol{a}$ by $\boldsymbol{v}$. We identify an affine subspace T with its corresponding linear subspace, for example, when saying that a vector belongs to T.

For two subspaces T and T', of possibly different dimensions, let $0 \le \theta_{\max}(T, T') \le \pi/2$ denote the largest and by $\theta_{\min}(T, T')$ the smallest nonzero principal angle between T and T' (Stewart and Sun, 1990). When $\boldsymbol{v}$ is a vector and T is a subspace, $\angle(\boldsymbol{v}, T) := \theta_{\max}(\mathbb{R}\boldsymbol{v}, T)$ this is the usual definition of the angle between $\boldsymbol{v}$ and T.

For a subset $A \subset \mathbb{R}^D$ and positive integer d, $\mathrm{vol}_d(A)$ denotes the d-dimensional Hausdorff measure of A, and $\mathrm{vol}(A)$ is defined as $\mathrm{vol}_{\dim(A)}(A)$, where $\dim(A)$ is the Hausdorff dimension of A. For a Borel set A, let λ_A denote the uniform distribution on A.

For a set $S \subset \mathbb{R}^D$ with reach at least $1/\kappa$, and $\boldsymbol{x}$ with $\mathrm{dist}(\boldsymbol{x}, S) < 1/\kappa$, let $P_S(\boldsymbol{x})$ denote the metric projection of $\boldsymbol{x}$ onto S, that is, the point on S closest to $\boldsymbol{x}$. Note that, if T is an affine subspace, then P_T is the usual orthogonal projection onto T, and we let $\boldsymbol{P}_T$ denote the orthogonal projection onto the linear subspace of same dimension and parallel to T. Let $\mathcal{S}_d(\kappa)$ denote the class of connected, C^2 and compact d-dimensional submanifolds without boundary embedded in $\mathbb{R}^D$, with reach at least $1/\kappa$. For a submanifold $S \in \mathbb{R}^D$, let $T_S(\boldsymbol{x})$ denote the tangent space of S at $\boldsymbol{x} \in S$.

We will often identify a linear map with its matrix in the canonical basis. For a symmetric (real) matrix $\boldsymbol{M}$, let $\beta_1(\boldsymbol{M}) \geq \beta_2(\boldsymbol{M}) \geq \cdots$ denote its eigenvalues in decreasing order.

We say that $f : \Omega \subset \mathbb{R}^D \to \mathbb{R}^D$ is C-Lipschitz if $\|f(\boldsymbol{x}) - f(\boldsymbol{y})\| \leq C\|\boldsymbol{x} - \boldsymbol{y}\|, \forall \boldsymbol{x}, \boldsymbol{y} \in \Omega$.

For two reals a and b, $a \vee b = \max(a, b)$ and $a \wedge b = \min(a, b)$. Additional notation will be introduced as needed.

6.1 Preliminaries

This section gathers a number of general results from geometry and probability. We took time to package them into standalone lemmas that could be of potential independent interest, particularly to researchers working in machine learning and computational geometry. When needed, we use C to denote a constant that may change with each appearance.

6.1.1 Smooth Surfaces and Their Tangent Subspaces

The following comes directly from (Federer, 1959, Th. 4.18(12)). It gives us a simple criterion for identifying the metric projection of a point on a surface with given reach.

Lemma 1. *Consider $S \in \mathcal{S}_d(\kappa)$ and $\boldsymbol{x} \in \mathbb{R}^D$ such that $\mathrm{dist}(\boldsymbol{x}, S) < 1/\kappa$. Then $\boldsymbol{s} = P_S(\boldsymbol{x})$ if and only if $\|\boldsymbol{x} - \boldsymbol{s}\| < 1/\kappa$ and $\boldsymbol{x} - \boldsymbol{s} \perp T_S(\boldsymbol{s})$.*

The following result is on approximating a smooth surface near a point by the tangent subspace at that point. It is based on (Federer, 1959, Th. 4.18(2)).

Lemma 2. *For $S \in \mathcal{S}_d(\kappa)$, and any two points $\boldsymbol{s}, \boldsymbol{s}' \in S$,*

$$\mathrm{dist}(\boldsymbol{s}', T_S(\boldsymbol{s})) \leq \frac{\kappa}{2}\|\boldsymbol{s}' - \boldsymbol{s}\|^2, \tag{10}$$

and when $\mathrm{dist}(\boldsymbol{s}', T_S(\boldsymbol{s})) \leq 1/\kappa$,

$$\mathrm{dist}(\boldsymbol{s}', T_S(\boldsymbol{s})) \leq \kappa\|P_{T_S(\boldsymbol{s})}(\boldsymbol{s}') - \boldsymbol{s}\|^2. \tag{11}$$

Moreover, for $\boldsymbol{t} \in T_S(\boldsymbol{s})$ such that $\|\boldsymbol{s} - \boldsymbol{t}\| \leq \frac{1}{3\kappa}$,

$$\mathrm{dist}(\boldsymbol{t}, S) \leq \kappa\|\boldsymbol{t} - \boldsymbol{s}\|^2. \tag{12}$$

Proof. Let T be short for $T_S(\boldsymbol{s})$. (Federer, 1959, Th. 4.18(2)) says that

$$\mathrm{dist}(\boldsymbol{s}' - \boldsymbol{s}, T) \leq \frac{\kappa}{2}\|\boldsymbol{s}' - \boldsymbol{s}\|^2. \tag{13}$$

Immediately, we have

$$\mathrm{dist}(\boldsymbol{s}' - \boldsymbol{s}, T) = \|\boldsymbol{s}' - P_T(\boldsymbol{s}')\| = \mathrm{dist}(\boldsymbol{s}', T),$$

and (10) comes from that. Based on that and Pythagoras theorem, we have

$$\mathrm{dist}(\boldsymbol{s}', T) = \|P_T(\boldsymbol{s}') - \boldsymbol{s}'\| \leq \frac{\kappa}{2}\|\boldsymbol{s}' - \boldsymbol{s}\|^2 = \frac{\kappa}{2}\big(\|P_T(\boldsymbol{s}') - \boldsymbol{s}'\|^2 + \|P_T(\boldsymbol{s}') - \boldsymbol{s}\|^2\big),$$

so that

$$\mathrm{dist}(\boldsymbol{s}', T)\big(1 - \frac{\kappa}{2}\,\mathrm{dist}(\boldsymbol{s}', T)\big) \leq \frac{\kappa}{2}\|P_T(\boldsymbol{s}') - \boldsymbol{s}\|^2,$$

and (11) follows easily from that. For (12), let $r = 1/(3\kappa)$ and $\boldsymbol{s}' = P_T^{-1}(\boldsymbol{t})$, the latter being well-defined by Lemma 5 below and belongs to $B(\boldsymbol{s}, r(1 + \kappa r)) \subset B(\boldsymbol{s}, 4/(9\kappa))$. By (10), $\|\boldsymbol{s}' - \boldsymbol{s}\| \leq 8/(81\kappa) < 1/\kappa$, and by (11),

$$\mathrm{dist}(\boldsymbol{t}, S) \leq \|\boldsymbol{t} - \boldsymbol{s}'\| = \mathrm{dist}(\boldsymbol{s}', T) \leq \kappa\|\boldsymbol{t} - \boldsymbol{s}\|^2.$$

This concludes the proof of (12). □

We will need a bound on the angle between tangent subspaces on a smooth surface as a function of the distance between the corresponding points of contact.

Lemma 3 (Boissonnat et al. (2013))**.** *For $S \in \mathcal{S}_d(\kappa)$, and any $\boldsymbol{s}, \boldsymbol{s}' \in S$,*

$$\sin\theta_{\max}(T_S(\boldsymbol{s}), T_S(\boldsymbol{s}')) \leq 6\kappa\|\boldsymbol{s}' - \boldsymbol{s}\|. \tag{14}$$

The following bounds the difference between the metric projection onto a surface and the orthogonal projection onto one of its tangents.

Lemma 4. *Consider $S \in \mathcal{S}_d(\kappa)$ and $\boldsymbol{s} \in S$. Then for any $\boldsymbol{x} \in B(S, r)$, we have*

$$\|P_S(\boldsymbol{x}) - P_{T_S(\boldsymbol{s})}(\boldsymbol{x})\| \leq C_4\kappa\|\boldsymbol{x} - \boldsymbol{s}\|^2,$$

for a numeric constant $C_4 > 0$.

Proof. Let $T = T_S(\boldsymbol{s})$ and define $\boldsymbol{t} = P_T(\boldsymbol{x})$, $\boldsymbol{s}' = P_S(\boldsymbol{x})$ and $\tilde{\boldsymbol{t}} = P_T(\boldsymbol{s}')$, and also $T' = T_S(\boldsymbol{s}')$ and $\theta = \theta_{\max}(T, T')$. We have

$$\begin{aligned} P_S(\boldsymbol{x}) &= P_{T'}(\boldsymbol{x}) = \boldsymbol{P}_{T'}(\boldsymbol{x} - \boldsymbol{s}') + \boldsymbol{s}' \\ P_T(\boldsymbol{x}) &= \boldsymbol{P}_T(\boldsymbol{x} - \tilde{\boldsymbol{t}}) + \tilde{\boldsymbol{t}} = \boldsymbol{P}_T(\boldsymbol{x} - \boldsymbol{s}') + \boldsymbol{P}_T(\boldsymbol{s}' - \tilde{\boldsymbol{t}}) + \tilde{\boldsymbol{t}}. \end{aligned}$$

Hence,

$$\|P_S(\boldsymbol{x}) - P_T(\boldsymbol{x})\| \leq \|\boldsymbol{P}_{T'}(\boldsymbol{x} - \boldsymbol{s}') - \boldsymbol{P}_T(\boldsymbol{x} - \boldsymbol{s}')\| + \|\boldsymbol{P}_T(\boldsymbol{s}' - \tilde{\boldsymbol{t}})\| + \|\boldsymbol{s}' - \tilde{\boldsymbol{t}}\|.$$

On the one hand,

$$\|\boldsymbol{P}_T(\boldsymbol{s}'-\tilde{\boldsymbol{t}})\| \le \|\boldsymbol{s}'-\tilde{\boldsymbol{t}}\| = \mathrm{dist}(\boldsymbol{s}',T) \le \frac{\kappa}{2}\|\boldsymbol{s}'-\boldsymbol{s}\|^2 \le 2\kappa\|\boldsymbol{x}-\boldsymbol{s}\|^2,$$

by (10) and the fact that

$$\|\boldsymbol{s}'-\boldsymbol{s}\| \le \|\boldsymbol{s}'-\boldsymbol{x}\| + \|\boldsymbol{x}-\boldsymbol{s}\| = \mathrm{dist}(\boldsymbol{x},S) + \|\boldsymbol{x}-\boldsymbol{s}\| \le 2\|\boldsymbol{x}-\boldsymbol{s}\|.$$

On the other hand, applying Lemma 18 (see further down)

$$\|\boldsymbol{P}_{T'}(\boldsymbol{x}-\boldsymbol{s}') - \boldsymbol{P}_T(\boldsymbol{x}-\boldsymbol{s}')\| \le \|\boldsymbol{P}_{T'}-\boldsymbol{P}_T\|\|\boldsymbol{x}-\boldsymbol{s}'\| = (\sin\theta)\|\boldsymbol{x}-\boldsymbol{s}'\|,$$

with $\|\boldsymbol{x}-\boldsymbol{s}'\| \le \|\boldsymbol{x}-\boldsymbol{s}\|$ and, applying Lemma 3,

$$\sin\theta \le 6\kappa\|\boldsymbol{s}'-\boldsymbol{s}\| \le 12\kappa\|\boldsymbol{x}-\boldsymbol{s}\|.$$

All together, we conclude. □

Below we state some properties of a projection onto a tangent subspace. A result similar to the first part was proved in (Arias-Castro et al., 2011, Lem. 2) based on results in (Niyogi et al., 2008), but the arguments are simpler here and the constants are sharper.

Lemma 5. *There is a numeric constant $C_5 \ge 1$ such that the following holds. Take $S \in \mathcal{S}_d(\kappa)$, $\boldsymbol{s} \in S$ and $r \le 1/C_5\kappa$, and let T be short for $T_S(\boldsymbol{s})$. P_T is injective on $B(\boldsymbol{s},r)\cap S$ and its image contains $B(\boldsymbol{s},r')\cap T$, where $r' := (1-C_5(\kappa r)^2)r$. Moreover, P_T^{-1} has Lipschitz constant bounded by $1+C_5(\kappa r)^2 \le 1+\kappa r$ over $B(\boldsymbol{s},r)\cap T$.*

Proof. Take $\boldsymbol{s}', \boldsymbol{s}'' \in S$ distinct such that $P_T(\boldsymbol{s}') = P_T(\boldsymbol{s}'')$. Equivalently, $\boldsymbol{s}''-\boldsymbol{s}'$ is perpendicular to T. Let T' be short for $T_S(\boldsymbol{s}')$. By (13) and the fact that $\mathrm{dist}(\boldsymbol{v},T) = \|\boldsymbol{v}\|\sin\angle(\boldsymbol{v},T)$ for any vector $\boldsymbol{v}$ and any linear subspace T, we have

$$\sin\angle(\boldsymbol{s}''-\boldsymbol{s}',T') \le \frac{\kappa}{2}\|\boldsymbol{s}''-\boldsymbol{s}'\|,$$

and by (14),

$$\sin\theta_{\max}(T,T') \le 6\kappa\|\boldsymbol{s}-\boldsymbol{s}'\|.$$

Now, by the triangle inequality,

$$\frac{\pi}{2} = \angle(\boldsymbol{s}''-\boldsymbol{s}',T) \le \angle(\boldsymbol{s}''-\boldsymbol{s}',T') + \theta_{\max}(T,T'),$$

so that

$$\frac{\kappa}{2}\|\boldsymbol{s}''-\boldsymbol{s}'\| \wedge 1 \ge \frac{\pi}{2} - \sin^{-1}\left(6\kappa\|\boldsymbol{s}'-\boldsymbol{s}\| \wedge 1\right).$$

When $\|\boldsymbol{s}'-\boldsymbol{s}\| \le 1/12\kappa$, the RHS is bounded from below by $\pi/2 - \sin^{-1}(1/2)$, which then implies that $\frac{\kappa}{2}\|\boldsymbol{s}''-\boldsymbol{s}'\| \ge \sin(\pi/2-\sin^{-1}(1/2)) = \sqrt{3}/2$, that is, $\|\boldsymbol{s}''-\boldsymbol{s}'\| \ge \sqrt{3}/\kappa$. This precludes the situation where $\boldsymbol{s}', \boldsymbol{s}'' \in B(\boldsymbol{s}, 1/12\kappa)$, so that P_T is injective on $B(\boldsymbol{s},r)$ when $r \le 1/12\kappa$.

The same arguments imply that P_T is an open map on $R := B(\boldsymbol{s},r)\cap S$. In particular, $P_T(R)$ contains an open ball in T centered at $\boldsymbol{s}$ and $P_T(\partial R) = \partial P_T(R)$, with $\partial R = S \cap \partial B(\boldsymbol{s},r)$ since $\partial S = \emptyset$. Now take any ray out of $\boldsymbol{s}$ within T, which is necessarily of the form $\boldsymbol{s}+\mathbb{R}_+\boldsymbol{v}$, where $\boldsymbol{v}$ is a unit vector in T. Let $\boldsymbol{t}_a = \boldsymbol{s}+a\boldsymbol{v} \in T$ for $a \in [0,\infty)$. Let a_* be the infimum over all $a > 0$ such that $\boldsymbol{t}_a \in P_T(R)$. Note that $a_* > 0$ and $\boldsymbol{t}_{a_*} \in P_T(\partial R)$, so that there is $\boldsymbol{s}_* \in \partial R$ such that $P_T(\boldsymbol{s}_*) = \boldsymbol{t}_{a_*}$. Let $\boldsymbol{s}_a = P_T^{-1}(\boldsymbol{t}_a)$, which is well-defined on $[0,a_*]$ by definition of a_* and the fact that P_T is injective on R. Let $J_{\boldsymbol{t}}$ denote the differential of P_T^{-1} at $\boldsymbol{t}$. We have that $\dot{\boldsymbol{s}}_a = J_{\boldsymbol{t}_a}\boldsymbol{v}$ is the unique vector in $T_a := T_S(\boldsymbol{s}_a)$ such that $P_T(\dot{\boldsymbol{s}}_a) = \boldsymbol{v}$. Elementary geometry shows that

$$\|P_T(\dot{\boldsymbol{s}}_a)\| = \|\dot{\boldsymbol{s}}_a\|\cos\angle(\dot{\boldsymbol{s}}_a,T) \ge \|\dot{\boldsymbol{s}}_a\|\cos\theta_{\max}(T_a,T),$$

with

$$\cos\theta_{\max}(T_a,T) \ge \cos\left[\sin^{-1}\left(6\kappa\|\boldsymbol{s}_a-\boldsymbol{s}\| \wedge 1\right)\right] \ge \zeta := 1-(6\kappa r)^2,$$

by (14) and fact that $\|\boldsymbol{s}_a-\boldsymbol{s}\| \le r$ (and assuming $6\kappa r \le 1$). Since $\|P_T(\dot{\boldsymbol{s}}_a)\| = \|\boldsymbol{v}\| = 1$, we have $\|\dot{\boldsymbol{s}}_a\| \le 1/\zeta$, and this holds for all $a < a_*$. So we can extend $\boldsymbol{s}_a$ to $[0,a_*]$ into a Lipschitz function with constant $1/\zeta$. Together with the fact that $\boldsymbol{s}_* \in \partial B(\boldsymbol{s},r)$, this implies that

$$r = \|\boldsymbol{s}_*-\boldsymbol{s}\| = \|\boldsymbol{s}_{a_*}-\boldsymbol{s}_0\| \le a_*/\zeta.$$

Hence, $a_* \ge \zeta r$ and therefore $P_T(R)$ contains $B(\boldsymbol{s},\zeta r)\cap T$ as stated.

For the last part, assume $r \le 1/C\kappa$, with C large enough that there is a unique $h \le 1/12\kappa$ such that $\zeta h = r$, where ζ is redefined as $\zeta := 1-(6\kappa h)^2$. Take $\boldsymbol{t}' \in B(\boldsymbol{s},r)\cap T$ and let $\boldsymbol{s}' = P_T^{-1}(\boldsymbol{t}')$ and $T' = T_S(\boldsymbol{s}')$. We saw that P_T^{-1} is Lipschitz with constant $1/\zeta$ on any ray emanating from $\boldsymbol{s}$ of length $\zeta h = r$, so that $\|\boldsymbol{s}'-\boldsymbol{s}\| \le (1/\zeta)\|\boldsymbol{t}'-\boldsymbol{s}\| \le r/\zeta = h$. The differential of P_T at $\boldsymbol{s}'$ is P_T itself, seen as a linear map between T' and T. Then for any vector $\boldsymbol{u} \in T'$, we have

$$\|P_T(\boldsymbol{u})\| = \|\boldsymbol{u}\|\cos\angle(\boldsymbol{u},T) \ge \|\boldsymbol{u}\|\cos\theta_{\max}(T',T),$$

with

$$\cos\theta_{\max}(T',T) \ge \cos\left[\sin^{-1}\left(6\kappa\|\boldsymbol{s}'-\boldsymbol{s}\|\right)\right] \ge 1-(6\kappa h)^2 = \zeta,$$

as before. Hence, $\|J_{\boldsymbol{t}'}\| \le 1/\zeta$, and we proved this for all $\boldsymbol{t}' \in B(\boldsymbol{s},r)\cap T$. This last set being convex, we can apply Taylor's theorem and get that P_T^{-1} is Lipschitz on that set with constant $1/\zeta$. We then note that $\zeta = 1+O(\kappa r)^2$. □

6.1.2 Volumes and Uniform Distributions

Below is a result that quantifies how much the volume of a set changes when applying a Lipschitz map. This is well-known in measure theory and we only provide a proof for completeness.

Lemma 6. *Suppose Ω is a measurable subset of $\mathbb{R}^D$ and $f : \Omega \subset \mathbb{R}^D \to \mathbb{R}^D$ is C-Lipschitz. Then for any measurable set $A \subset \Omega$ and real $d > 0$, $\mathrm{vol}_d(f(A)) \le C^d\,\mathrm{vol}_d(A)$.*

Proof. By definition,

$$\mathrm{vol}_d(A) = \lim_{t\to 0} V_d^t(A), \qquad V_d^t(A) := \inf_{(R_i)\in\mathcal{R}^t(A)} \sum_{i\in\mathbb{N}} \mathrm{diam}(R_i)^d,$$

where $\mathcal{R}^t(A)$ is the class of countable sequences $(R_i : i \in \mathbb{N})$ of subsets of $\mathbb{R}^D$ such that $A \subset \bigcup_i R_i$ and $\operatorname{diam}(R_i) < t$ for all i. Since f is C-Lipschitz, $\operatorname{diam}(f(R)) \leq C \operatorname{diam}(R)$ for any $R \subset \Omega$. Hence, for any $(R_i) \in \mathcal{R}^t(A)$, $(f(R_i)) \in \mathcal{R}^{Ct}(f(A))$. This implies that

$$V_d^{Ct}(f(A)) \leq \sum_{i\in\mathbb{N}} \operatorname{diam}(f(R_i))^d \leq C^d \sum_{i\in\mathbb{N}} \operatorname{diam}(R_i)^d.$$

Taking the infimum over $(R_i) \in \mathcal{R}^t(A)$, we get $V_d^{Ct}(f(A)) \leq C^d V_d^t(A)$, and we conclude by taking the limit as $t \to 0$, noticing that $V_d^{Ct}(f(A)) \to \operatorname{vol}_d(f(A))$ while $V_d^t(A) \to \operatorname{vol}_d(A)$. □

We compare below two uniform distributions. For two Borel probability measures P and Q on $\mathbb{R}^D$, $\operatorname{TV}(P, Q)$ denotes their total variation distance, meaning,

$$\operatorname{TV}(P, Q) = \sup\{|P(A) - Q(A)| : A \text{ Borel}\}.$$

Remember that for a Borel set A, λ_A denotes the uniform distribution on A.

Lemma 7. *Suppose A and B are two Borel subsets of $\mathbb{R}^D$. Then*

$$\operatorname{TV}(\lambda_A, \lambda_B) \leq 4\, \frac{\operatorname{vol}(A \triangle B)}{\operatorname{vol}(A \cup B)}.$$

Proof. If A and B are not of same dimension, say $\dim(A) > \dim(B)$, then $\operatorname{TV}(\lambda_A, \lambda_B) = 1$ since $\lambda_A(B) = 0$ while $\lambda_B(B) = 1$. And we also have

$$\operatorname{vol}(A \triangle B) = \operatorname{vol}_{\dim(A)}(A \triangle B) = \operatorname{vol}_{\dim(A)}(A) = \operatorname{vol}(A),$$

and

$$\operatorname{vol}(A \cup B) = \operatorname{vol}_{\dim(A)}(A \cup B) = \operatorname{vol}_{\dim(A)}(A) = \operatorname{vol}(A),$$

in both cases because $\operatorname{vol}_{\dim(A)}(B) = 0$. So the result works in that case.

Therefore assume that A and B are of same dimension. Assume WLOG that $\operatorname{vol}(A) \geq \operatorname{vol}(B)$. For any Borel set U,

$$\lambda_A(U) - \lambda_B(U) = \frac{\operatorname{vol}(A \cap U)}{\operatorname{vol}(A)} - \frac{\operatorname{vol}(B \cap U)}{\operatorname{vol}(B)},$$

so that

$$\begin{aligned} |\lambda_A(U) - \lambda_B(U)| &= \left| \frac{\operatorname{vol}(A \cap U)}{\operatorname{vol}(A)} - \frac{\operatorname{vol}(B \cap U)}{\operatorname{vol}(A)} + \frac{\operatorname{vol}(B \cap U)}{\operatorname{vol}(A)} - \frac{\operatorname{vol}(B \cap U)}{\operatorname{vol}(B)} \right| \\ &\leq \frac{|\operatorname{vol}(A \cap U) - \operatorname{vol}(B \cap U)|}{\operatorname{vol}(A)} + \frac{\operatorname{vol}(B \cap U)}{\operatorname{vol}(B)} \frac{|\operatorname{vol}(A) - \operatorname{vol}(B)|}{\operatorname{vol}(A)} \\ &\leq \frac{2 \operatorname{vol}(A \triangle B)}{\operatorname{vol}(A)}, \end{aligned}$$

and we conclude with the fact that $\operatorname{vol}(A \cup B) \leq \operatorname{vol}(A) + \operatorname{vol}(B) \leq 2 \operatorname{vol}(A)$. □

We now look at the projection of the uniform distribution on a neighborhood of a surface onto a tangent subspace. For a Borel probability measure P and measurable function $f : \mathbb{R}^D \to \mathbb{R}^D$, P^f denotes the push-forward (Borel) measure defined by $P^f(A) = P(f^{-1}(A))$.

Lemma 8. *Suppose $A \subset \mathbb{R}^D$ is Borel and $f : A \to \mathbb{R}^D$ is invertible on $f(A)$, and that both f and f^{-1} are C-Lipschitz. Then*

$$\operatorname{TV}(\lambda_A^f, \lambda_{f(A)}) \leq 2(C^{\dim(A)} - 1).$$

Proof. First, note that A and $f(A)$ are both of same dimension, and that $C \geq 1$ necessarily. Let d be short for $\dim(A)$. Take $U \subset f(A)$ Borel and let $V = f^{-1}(U)$. Then

$$\lambda_A^f(U) = \frac{\operatorname{vol}(A \cap V)}{\operatorname{vol}(A)}, \qquad \lambda_{f(A)}(U) = \frac{\operatorname{vol}(f(A) \cap U)}{\operatorname{vol}(f(A))},$$

and as in (16),

$$|\lambda_A^f(U) - \lambda_{f(A)}(U)| \leq \frac{|\operatorname{vol}(A \cap V) - \operatorname{vol}(f(A) \cap U)|}{\operatorname{vol}(A)} + \frac{|\operatorname{vol}(A) - \operatorname{vol}(f(A))|}{\operatorname{vol}(A)}.$$

f being invertible, we have $f(A \cap V) = f(A) \cap U$ and $f^{-1}(f(A) \cap U) = A \cap V$. Therefore, applying Lemma 6, we get

$$C^{-d} \leq \frac{\operatorname{vol}(f(A) \cap U)}{\operatorname{vol}(A \cap V)} \leq C^d,$$

so that

$$|\operatorname{vol}(A \cap V) - \operatorname{vol}(f(A) \cap U)| \leq (C^d - 1) \operatorname{vol}(A \cap V) \leq (C^d - 1) \operatorname{vol}(A).$$

And taking $V = A$, we also get

$$|\operatorname{vol}(A) - \operatorname{vol}(f(A))| \leq (C^d - 1) \operatorname{vol}(A).$$

From this we conclude. □

Now comes a technical result on the intersection of a smooth surface and a ball.

Lemma 9. *There is a constant $C_9 \geq 3$ depending only on d such that the following is true. Take $S \in \mathcal{S}_d(\kappa)$, $r < \frac{1}{C_9 \kappa}$ and $\boldsymbol{x} \in \mathbb{R}^D$ such that $\operatorname{dist}(\boldsymbol{x}, S) < \kappa$. Let $\boldsymbol{s} = P_S(\boldsymbol{x})$ and $T = T_S(\boldsymbol{s})$. Then*

$$\operatorname{vol}\left(P_T(S \cap B(\boldsymbol{x}, r)) \triangle (T \cap B(\boldsymbol{x}, r))\right) \leq C_9 \kappa (\|\boldsymbol{x} - \boldsymbol{s}\| + \kappa r^2) \operatorname{vol}(T \cap B(\boldsymbol{x}, r)).$$

Proof. If $\operatorname{dist}(\boldsymbol{x}, S) > r$, then $S \cap B(\boldsymbol{x}, r) = T \cap B(\boldsymbol{x}, r) = \emptyset$, and if $\operatorname{dist}(\boldsymbol{x}, S) = r$, then $S \cap B(\boldsymbol{x}, r) = T \cap B(\boldsymbol{x}, r) = \{\boldsymbol{s}\}$, and in both cases the inequality holds trivially. So it suffices to consider the case where $\operatorname{dist}(\boldsymbol{x}, S) < r$.

Let $A_r = B(\boldsymbol{s}, r)$, $B_r = B(\boldsymbol{x}, r)$ and $g = P_T$ for short. Note that $T \cap B_r = T \cap A_{r_0}$ where $r_0 := (r^2 - \delta^2)^{1/2}$ and $\delta := \|\boldsymbol{x} - \boldsymbol{s}\|$. Take $\boldsymbol{s}_1 \in S \cap B_r$ such that $g(\boldsymbol{s}_1)$ is farthest from $\boldsymbol{s}$, so that $g(S \cap B_r) \subset A_{r_1}$ where $r_1 := \|\boldsymbol{s} - g(\boldsymbol{s}_1)\|$—note that $r_1 \leq r$. Let $\ell_1 = \|\boldsymbol{s}_1 - g(\boldsymbol{s}_1)\|$ and $\boldsymbol{y}_1$ be the orthogonal projection of $\boldsymbol{s}_1$ onto the line $(\boldsymbol{x}, \boldsymbol{s})$. By Pythagoras theorem,

we have $\|\boldsymbol{x}-\boldsymbol{s}_1\|^2 = \|\boldsymbol{x}-\boldsymbol{y}_1\|^2 + \|\boldsymbol{y}_1-\boldsymbol{s}_1\|^2$. We have $\|\boldsymbol{x}-\boldsymbol{s}_1\| \le r$ and $\|\boldsymbol{y}_1-\boldsymbol{s}_1\| = \|\boldsymbol{s}-g(\boldsymbol{s}_1)\| = r_1$. And because $\ell_1 \le \kappa r_1^2 < r$ by (11), either $\boldsymbol{y}_1$ is between $\boldsymbol{x}$ and $\boldsymbol{s}$, in which case $\|\boldsymbol{x}-\boldsymbol{y}_1\| = \delta - \ell_1$, or $\boldsymbol{s}$ is between $\boldsymbol{x}$ and $\boldsymbol{y}_1$, in which case $\|\boldsymbol{x}-\boldsymbol{y}_1\| = \delta + \ell_1$. In any case, $r^2 \ge r_1^2 + (\delta-\ell_1)^2$, which together with $\ell_1 \le \kappa r_1^2$ implies $r_1^2 \le r^2 - \delta^2 + 2\delta\ell_1 \le r_0^2 + 2\kappa r_1^2\delta$, leading to $r_1 \le (1-2\kappa\delta)^{-1/2} r_0 \le (1+4\kappa\delta) r_0$ after noticing that $\delta \le r < 1/(3\kappa)$. From $g(S\cap B_r) \subset T \cap A_{r_1}$, we get

$$\begin{aligned}\mathrm{vol}\left(g(S\cap B_r)\setminus(T\cap B_r)\right) &\le \mathrm{vol}(T\cap A_{r_1}) - \mathrm{vol}(T\cap A_{r_0}) \\ &= ((r_1/r_0)^d - 1)\,\mathrm{vol}(T\cap A_{r_0}).\end{aligned}$$

We follow similar arguments to get a sort of reverse relationship. Take $\boldsymbol{s}_2 \in S\cap B_r$ such that $g(S\cap B_r) \supset T\cap A_{r_2}$, where $r_2 := \|\boldsymbol{s}-g(\boldsymbol{s}_2)\|$ is largest. Assuming r is small enough, by Lemma 5, g^{-1} is well-defined on $T\cap A_r$, so that necessarily $\boldsymbol{s}_2 \in \partial B_r$. Let $\ell_2 = \|\boldsymbol{s}_2 - g(\boldsymbol{s}_2)\|$ and $\boldsymbol{y}_2$ be the orthogonal projection of $\boldsymbol{s}_2$ onto the line $(\boldsymbol{x},\boldsymbol{s})$. By Pythagoras theorem, we have $\|\boldsymbol{x}-\boldsymbol{s}_2\|^2 = \|\boldsymbol{x}-\boldsymbol{y}_2\|^2 + \|\boldsymbol{y}_2-\boldsymbol{s}_2\|^2$. We have $\|\boldsymbol{x}-\boldsymbol{s}_2\| = r$ and $\|\boldsymbol{y}_2-\boldsymbol{s}_2\| = \|\boldsymbol{s}-g(\boldsymbol{s}_2)\| = r_2$. And by the triangle inequality, $\|\boldsymbol{x}-\boldsymbol{y}_2\| \le \|\boldsymbol{x}-\boldsymbol{s}\| + \|\boldsymbol{y}_2-\boldsymbol{s}\| = \delta + \ell_2$. Hence, $r^2 \le r_2^2 + (\delta+\ell_2)^2$, which together with $\ell_2 \le \kappa r_2^2$ by (11), implies $r_2^2 \ge r^2 - \delta^2 - 2\delta\ell_2 - \ell_2^2 \ge r_0^2 - (2\delta + \kappa r^2)\kappa r_2^2$, leading to $r_2 \ge (1 + 2\kappa\delta + \kappa^2 r^2)^{-1/2} r_0 \ge (1 - 2\kappa\delta - \kappa^2 r^2) r_0$. From $g(S\cap B_r) \supset T\cap A_{r_2}$, we get

$$\begin{aligned}\mathrm{vol}\left((T\cap B_r)\setminus g(S\cap B_r)\right) &\le \mathrm{vol}(T\cap A_{r_0}) - \mathrm{vol}(T\cap A_{r_2}) \\ &= (1-(r_2/r_0)^d)\,\mathrm{vol}(T\cap A_{r_0}).\end{aligned}$$

All together, we have

$$\begin{aligned}\mathrm{vol}\left(g(S\cap B_r)\,\triangle\,(T\cap B_r)\right) &\le \left((r_1/r_0)^d - (r_2/r_0)^d\right)\ \mathrm{vol}(T\cap A_{r_0}) \\ &\le \left((1+4\kappa\delta)^d - (1-2\kappa\delta-\kappa^2r^2)^d\right)\mathrm{vol}((T\cap B_r)) \\ &\le Cd\kappa(\delta+\kappa r^2)\,\mathrm{vol}((T\cap B_r)),\end{aligned}$$

when $\delta \le r \le 1/(C\kappa)$ and C is a large-enough numerical constant. □

We bound below the d-volume of a the intersection of a ball with a smooth surface. Although it could be obtained as a special case of Lemma 9, we provide a direct proof because this result is at the cornerstone of many results in the literature on sampling points uniformly on a smooth surface.

Lemma 10. *Suppose $S \in \mathcal{S}_d(\kappa)$. Then for any $\boldsymbol{s} \in S$ and $r < \frac{1}{(d\vee C_5)\kappa}$, we have*

$$1 - 2d\kappa r \le \frac{\mathrm{vol}(S\cap B(\boldsymbol{s},r))}{\mathrm{vol}(T\cap B(\boldsymbol{s},r))} \le 1 + 2d\kappa r,$$

where $T := T_S(\boldsymbol{s})$ is the tangent subspace of S at $\boldsymbol{s}$.

Proof. Let $T = T_S(\boldsymbol{s})$, $B_r = B(\boldsymbol{s},r)$ and $g = P_T$ for short. By Lemma 5, g is 1-Lipschitz and g^{-1} is $(1+\kappa r)$-Lipschitz on $T\cap B_r$, so by Lemma 6 we have

$$(1+\kappa r)^{-d} \le \frac{\mathrm{vol}(g(S\cap B_r))}{\mathrm{vol}(S\cap B_r)} \le 1.$$

That g^{-1} is Lipschitz with constant $1+\kappa r$ on $g(S\cap B_r)$ also implies that $g(S\cap B_r)$ contains $T\cap B_{r'}$ where $r' := r/(1+\kappa r)$. From this, and the fact that $g(S\cap B_r) \subset T\cap B_r$, we get

$$1 \le \frac{\mathrm{vol}(T\cap B_r)}{\mathrm{vol}(g(S\cap B_r))} \le \frac{\mathrm{vol}(T\cap B_r)}{\mathrm{vol}(T\cap B_{r'})} = \frac{r^d}{r'^d} = (1+\kappa r)^d.$$

We therefore have

$$\mathrm{vol}(S\cap B_r) \ge \mathrm{vol}(g(S\cap B_r)) \ge (1+\kappa r)^{-d}\,\mathrm{vol}(T\cap B_r),$$

and

$$\mathrm{vol}(S\cap B_r) \le (1+\kappa r)^d\,\mathrm{vol}(g(S\cap B_r)) \le (1+\kappa r)^d\,\mathrm{vol}(T\cap B_r).$$

And we conclude with the inequality $(1+x)^d \le 1 + 2dx$ valid for any $x \in [0, 1/d]$ and any $d \ge 1$. □

We now look at the density of a sample from the uniform on a smooth, compact surface.

Lemma 11. *Consider $S \in \mathcal{S}_d(\kappa)$. There is a constant $C_{11} > 0$ depending on S such that the following is true. Sample n points $\boldsymbol{s}_1, \dots, \boldsymbol{s}_n$ independently and uniformly at random from S. Take $0 < r < 1/(C_{11}\kappa)$. Then with probability at least $1 - C_{11} r^{-d} \exp(-nr^d/C_{11})$, any ball of radius r with center on S has between nr^d/C_{11} and $C_{11} nr^d$ sample points.*

Proof. For a set R, let $N(R)$ denote the number of sample points in R. For any R measurable, $N(R) \sim \mathrm{Bin}(n, p_R)$, where $p_R := \mathrm{vol}(R\cap S)/\mathrm{vol}(S)$. Let $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m$ be a maximal $(r/2)$-packing of S, and let $B_i = B(\boldsymbol{x}_i, r/4) \cap S$. For any $\boldsymbol{s} \in S$, there is i such that $\|\boldsymbol{s} - \boldsymbol{x}_i\| \le r/2$, which implies $B_i \subset B(\boldsymbol{s}, r)$ by the triangle inequality. Hence, $\min_{\boldsymbol{s}\in S} N(B(\boldsymbol{s},r)) \ge \min_i N(B_i)$.

By the fact that $B_i \cap B_j = \emptyset$ for $i \ne j$,

$$\mathrm{vol}(S) \ge \sum_{i=1}^m \mathrm{vol}(B_i) \ge m \min_i \mathrm{vol}(B_i),$$

and assuming that κr is small enough, we have

$$\min_i \mathrm{vol}(B_i) \ge \frac{\omega_d}{2}(r/4)^d,$$

by Lemma 10, where ω_d is the volume of the d-dimensional unit ball. This leads to $m \le Cr^{-d}$ and $p := \min_i p_{B_i} \ge r^d/C$, when $\kappa r \le 1/C$, where $C > 0$ depends only on S.

Now, applying Bernstein's inequality to the binomial distribution, we get

$$\mathbb{P}\left(N(B_i) \le np/2\right) \le \mathbb{P}\left(N(B_i) \le np_{B_i}/2\right) \le e^{-(3/32)np_{B_i}} \le e^{-(3/32)np}.$$

We follow this with the union bound, to get

$$\mathbb{P}\left(\min_{\boldsymbol{s}\in S} N(B(\boldsymbol{s},r)) \le nr^d/(2C)\right) \le me^{-(3/32)np} \le\ Cr^{-d} e^{-\frac{3}{32C}nr^d}.$$

From this the lower bound follows. The proof of the upper bound is similar. □

Next, we bound the volume of the symmetric difference between two balls.

Lemma 12. *Take $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$ and $0 < \delta \leq 1$. Then*

$$\frac{\mathrm{vol}(B(\boldsymbol{x}, \delta) \triangle B(\boldsymbol{y}, 1))}{2\,\mathrm{vol}(B(0,1))} \leq 1 - (1 - \|\boldsymbol{x} - \boldsymbol{y}\|)_+^d \wedge \delta^d.$$

Proof. It suffices to prove the result when $\|\boldsymbol{x} - \boldsymbol{y}\| < 1$. In that case, with $\gamma := (1 - \|\boldsymbol{x} - \boldsymbol{y}\|) \wedge \delta$, we have $B(\boldsymbol{x}, \gamma) \subset B(\boldsymbol{x}, \delta) \cap B(\boldsymbol{y}, 1)$, so that

$$\begin{aligned} \mathrm{vol}(B(\boldsymbol{x}, \delta) \triangle B(\boldsymbol{y}, 1)) &= \mathrm{vol}(B(\boldsymbol{x}, \delta)) + \mathrm{vol}(B(\boldsymbol{y}, 1)) - 2\,\mathrm{vol}(B(\boldsymbol{x}, \delta) \cap B(\boldsymbol{y}, 1)) \\ &\leq 2\,\mathrm{vol}(B(\boldsymbol{y}, 1)) - 2\,\mathrm{vol}(B(\boldsymbol{x}, \gamma)) \\ &= 2\,\mathrm{vol}(B(0,1))(1 - \gamma^d). \end{aligned}$$

This concludes the proof. □

6.1.3 Covariances

The result below describes explicitly the covariance matrix of the uniform distribution over the unit ball of a subspace.

Lemma 13. *Let T be a subspace of dimension d. Then the covariance matrix of the uniform distribution on $T \cap B(0, a)$ (seen as a linear map) is equal to $a^2 c \boldsymbol{P}_T$, where $c := \frac{1}{d+2}$.*

Proof. Assume WLOG that $T = \mathbb{R}^d \times \{0\}$. Let X be distributed according to the uniform distribution on $T \cap B(0,1)$ and let $R = \|X\|$. Note that

$$\mathbb{P}(R \leq r) = \frac{\mathrm{vol}(T \cap B(0,r))}{\mathrm{vol}(T \cap B(0,1))} = r^d, \quad \forall r \in [0,1].$$

By symmetry, $\mathbb{E}(X_i X_j) = 0$ if $i \neq j$, while

$$\mathbb{E}(X_1^2) = \frac{1}{d}\,\mathbb{E}(X_1^2 + \cdots + X_d^2) = \frac{1}{d}\,\mathbb{E}(R^2) = \frac{1}{d}\int_0^t r^2 (d r^{d-1}) \mathrm{d}r = \frac{1}{d+2}.$$

Now the covariance matrix of the uniform distribution on $T \cap B(0,a)$ equals $\mathbb{E}[(aX)(aX)^\top] = a^2\,\mathbb{E}[X X^\top]$, which is exactly the representation of $a^2 c \boldsymbol{P}_T$ in the canonical basis of $\mathbb{R}^D$. □

We now show that a bound on the total variation distance between two compactly supported distributions implies a bound on the difference between their covariance matrices. For a measure P on $\mathbb{R}^D$ and an integrable function f, let $P(f)$ denote the integral of f with respect to P, that is,

$$P(f) = \int f(x) P(dx),$$

and let $\mathbb{E}(P) = P(\boldsymbol{x})$ and $\mathrm{Cov}(P) = P(\boldsymbol{x}\boldsymbol{x}^\top) - P(\boldsymbol{x})P(\boldsymbol{x})^\top$ denote the mean and covariance matrix of P, respectively.

Lemma 14. *Suppose λ and ν are two Borel probability measures on $\mathbb{R}^d$ supported on $B(0,1)$. Then*

$$\|\,\mathbb{E}(\lambda) - \mathbb{E}(\nu)\| \leq \sqrt{d}\,\mathrm{TV}(\lambda, \nu), \qquad \|\,\mathrm{Cov}(\lambda) - \mathrm{Cov}(\nu)\| \leq 3d\,\mathrm{TV}(\lambda, \nu).$$

Proof. Let $f_k(\boldsymbol{t}) = t_k$ when $\boldsymbol{t} = (t_1, \ldots, t_d)$, and note that $|f_k(\boldsymbol{t})| \leq 1$ for all k and all $\boldsymbol{t} \in B(0,1)$. By the fact that

$$\mathrm{TV}(\lambda, \nu) = \sup\{\lambda(f) - \nu(f) : f : \mathbb{R}^d \to \mathbb{R} \text{ measurable with } |f| \leq 1\},$$

we have

$$|\lambda(f_k) - \nu(f_k)| \leq \mathrm{TV}(\lambda, \nu), \quad \forall k = 1, \ldots, d.$$

Therefore,

$$\|\,\mathbb{E}(\lambda) - \mathbb{E}(\nu)\|^2 = \sum_{k=1}^d (\lambda(f_k) - \nu(f_k))^2 \leq d\,\mathrm{TV}(\lambda, \nu)^2,$$

which proves the first part.

Similarly, let $f_{k\ell}(\boldsymbol{t}) = t_k t_\ell$. Since $|f_{k\ell}(\boldsymbol{t})| \leq 1$ for all k, ℓ and all $\boldsymbol{t} \in B(0,1)$, we have

$$|\lambda(f_{k\ell}) - \nu(f_{k\ell})| \leq \mathrm{TV}(\lambda, \nu), \quad \forall k, \ell = 1, \ldots, d.$$

Since for any probability measure μ on $\mathbb{R}^d$,

$$\mathrm{Cov}(\mu) = \big(\mu(f_{k\ell}) - \mu(f_k)\mu(f_\ell) : k, \ell = 1, \ldots, d\big),$$

we have

$$\begin{aligned} \|\,\mathrm{Cov}(\lambda) - \mathrm{Cov}(\nu)\| &\leq d \max_{k,\ell} \big(|\lambda(f_{k\ell}) - \nu(f_{k\ell})| + |\lambda(f_k)\lambda(f_\ell) - \nu(f_k)\nu(f_\ell)|\big) \\ &\leq d \max_{k,\ell} \big(|\lambda(f_{k\ell}) - \nu(f_{k\ell})| + |\lambda(f_k)||\lambda(f_\ell) - \nu(f_\ell)| \\ &\qquad + |\nu(f_\ell)||\lambda(f_k) - \nu(f_k)|\big) \\ &\leq 3d\,\mathrm{TV}(\lambda, \nu), \end{aligned}$$

using the fact that $|\lambda(f_k)| \leq 1$ and $|\nu(f_k)| \leq 1$ for all k. □

The following compares the mean and covariance matrix of a distribution before and after transformation by a function.

Lemma 15. *Let λ be a Borel distribution with compact support $A \subset \mathbb{R}^D$ and consider a measurable function $g : A \mapsto \mathbb{R}^D$ such that $\|g(x) - x\| \leq \eta$. Then*

$$\|\,\mathbb{E}(\lambda^g) - \mathbb{E}(\lambda)\| \leq \eta, \quad \|\,\mathrm{Cov}(\lambda^g) - \mathrm{Cov}(\lambda)\| \leq \eta(\mathrm{diam}(A) + \eta).$$

Proof. Take $X \sim \lambda$ and let $Y = g(X) \sim \lambda^g$. For the means, by Jensen's inequality,

$$\|\,\mathbb{E}\,X - \mathbb{E}\,Y\| \leq \mathbb{E}\,\|X - Y\| = \mathbb{E}\,\|X - g(X)\| \leq \eta.$$

For the covariances, we first note that

$$\mathrm{Cov}(X) - \mathrm{Cov}(Y) = \frac{1}{2}\big(\mathrm{Cov}(X - Y, X + Y) + \mathrm{Cov}(X + Y, X - Y)\big),$$

where $\mathrm{Cov}(U,V) := \mathbb{E}((U-\mathbb{E}\,U)(V-\mathbb{E}\,V)^T)$ is the cross-covariance of random vectors U and V. By Jensen's inequality, the fact $\|\boldsymbol{u}\boldsymbol{v}^T\| = \|\boldsymbol{u}\|\|\boldsymbol{v}\|$ for any pair of vectors $\boldsymbol{u},\boldsymbol{v}$, and then the Cauchy-Schwarz inequality,

$$\|\mathrm{Cov}(U,V)\| \le \mathbb{E}(\|U-\mathbb{E}\,U\|\cdot\|V-\mathbb{E}\,V\|) \le \mathbb{E}(\|U-\mathbb{E}\,U\|^2)^{1/2}\cdot\mathbb{E}(\|V-\mathbb{E}\,V\|^2)^{1/2}.$$

Hence,

$$\begin{aligned}\|\mathrm{Cov}(X)-\mathrm{Cov}(Y)\| &\le \|\mathrm{Cov}(X-Y,X+Y)\| \\ &\le \mathbb{E}\left[\|X-Y-\mathbb{E}\,X+\mathbb{E}\,Y\|^2\right]^{1/2}\mathbb{E}\left[\|X+Y-\mathbb{E}\,X-\mathbb{E}\,Y\|^2\right]^{1/2}.\end{aligned}$$

By the fact that the mean minimizes the mean-squared error,

$$\mathbb{E}\left[\|X-Y-\mathbb{E}\,X+\mathbb{E}\,Y\|^2\right]^{1/2} \le \mathbb{E}\left[\|X-Y\|^2\right]^{1/2} = \mathbb{E}\left[\|X-g(X)\|^2\right]^{1/2} \le \eta.$$

In the same vein, letting $\boldsymbol{z}\in\mathbb{R}^D$ be such that $\|\boldsymbol{x}-\boldsymbol{z}\| \le \frac{1}{2}\,\mathrm{diam}(A)$ for all $\boldsymbol{x}\in A$, we get

$$\begin{aligned}\mathbb{E}\left[\|X+Y-\mathbb{E}\,X-\mathbb{E}\,Y\|^2\right]^{1/2} &\le \mathbb{E}\left[\|X-\boldsymbol{z}+Y-\boldsymbol{z}\|^2\right]^{1/2} \\ &\le \mathbb{E}\left[\|X-\boldsymbol{z}\|^2\right]^{1/2} + \mathbb{E}\left[\|Y-\boldsymbol{z}\|^2\right]^{1/2} \\ &\le 2\,\mathbb{E}\left[\|X-\boldsymbol{z}\|^2\right]^{1/2} + \mathbb{E}\left[\|X-g(X)\|^2\right]^{1/2} \\ &\le \mathrm{diam}(A)+\eta.\end{aligned}$$

Using the triangle inequality twice. From this, we conclude. □

Next we compare the covariance matrix of the uniform distribution on a small piece of smooth surface with that of the uniform distribution on the projection of that piece onto a nearby tangent subspace.

Lemma 16. *There is a constant $C_{16}>0$ depending only on d such that the following is true. Take $S\in\mathcal{S}_d(\kappa)$, $r<\frac{1}{C_{16}\kappa}$ and $\boldsymbol{x}\in\mathbb{R}^D$ such that $\mathrm{dist}(\boldsymbol{x},S)\le r$. Let $\boldsymbol{s}=P_S(\boldsymbol{x})$ and $T=T_S(\boldsymbol{s})$. If $\boldsymbol{\zeta}$ and $\boldsymbol{\xi}$ are the means, and $\boldsymbol{M}$ and $\boldsymbol{N}$ are the covariance matrices, of the uniform distributions on $S\cap B(\boldsymbol{x},r)$ and $T\cap B(\boldsymbol{x},r)$, respectively, then*

$$\|\boldsymbol{\zeta}-\boldsymbol{\xi}\| \le C_{16}\kappa r^2, \qquad \|\boldsymbol{M}-\boldsymbol{N}\| \le C_{16}\kappa r^3.$$

Proof. We focus on proving the bound on the covariances, and leave the bound on the means—whose proof is both similar and simpler—as an exercise to the reader. Let $T=T_S(\boldsymbol{s})$, $B_r=B(\boldsymbol{x},r)$ and $g=P_T$ for short. Let $A=S\cap B_r$ and $A'=T\cap B_r$.

First, applying Lemma 15, assuming κr is sufficiently small, we get that

$$\|\mathrm{Cov}(\lambda_A)-\mathrm{Cov}(\lambda_A^g)\| \le \frac{\kappa}{2}r^2\Big(2r+\frac{\kappa}{2}r^2\Big) \le 2\kappa r^3, \tag{38}$$

because $\mathrm{diam}(A)\le 2r$ and $\|g(\boldsymbol{x})-\boldsymbol{x}\| \le \frac{\kappa}{2}\|\boldsymbol{x}-\boldsymbol{s}\|^2 \le \frac{\kappa}{2}r^2$ for all $\boldsymbol{x}\in A$ by (10).

Let $\lambda_{g(A)}$ denote the uniform distribution on $g(A)$. λ_A^g and $\lambda_{g(A)}$ are both supported on B_r, so that applying Lemma 14 with proper scaling, we get

$$\|\mathrm{Cov}(\lambda_A^g)-\mathrm{Cov}(\lambda_{g(A)})\| \le 3dr^2\,\mathrm{TV}(\lambda_A^g,\lambda_{g(A)}).$$

When κr is small enough, we know that g is 1-Lipschitz, and by Lemma 5, g^{-1} is well-defined and is $(1+\kappa r)$-Lipschitz on A'. Hence, by Lemma 8 and the fact that $\dim(A)=d$, we have

$$\mathrm{TV}(\lambda_A^g,\lambda_{g(A)}) \le 2((1+\kappa r)^d-1) \le 4d\kappa r,$$

using the inequality $(1+x)^d\le 1+2dx$, valid for any $x\in[0,1/d]$ and any $d\ge 1$.

Noting that $\lambda_{A'}$ is also supported on B_r, applying Lemma 14 with proper scaling, we get

$$\|\mathrm{Cov}(\lambda_{g(A)})-\mathrm{Cov}(\lambda_{A'})\| \le 3dr^2\,\mathrm{TV}(\lambda_{g(A)},\lambda_{A'}),$$

with

$$\mathrm{TV}(\lambda_{g(A)},\lambda_{A'}) \le 4\frac{\mathrm{vol}(g(A)\triangle A')}{\mathrm{vol}(A')} \le 4C_9\kappa(r+\kappa r^2) \le 8C_9\kappa r,$$

by Lemma 7 and Lemma 9, and assuming that κr is small enough.

By the triangle inequality,

$$\begin{aligned}\|\boldsymbol{M}-\boldsymbol{N}\| &= \|\mathrm{Cov}(\lambda_A)-\mathrm{Cov}(\lambda_{A'})\| \\ &\le \|\mathrm{Cov}(\lambda_A)-\mathrm{Cov}(\lambda_A^g)\| + \|\mathrm{Cov}(\lambda_A^g)-\mathrm{Cov}(\lambda_{g(A)})\| \\ &\qquad + \|\mathrm{Cov}(\lambda_{g(A)})-\mathrm{Cov}(\lambda_{A'})\| \\ &\le 2\kappa r^3 + 12d^2\kappa r^3 + 24dC_9\kappa r^3.\end{aligned}$$

From this, we conclude. □

Next is a lemma on the estimation of a covariance matrix. The result is a simple consequence of the matrix Hoeffding inequality of Tropp (2012). Note that simply bounding the operator norm by the Frobenius norm, and then applying the classical Hoeffding inequality (Hoeffding, 1963) would yield a bound sufficient for our purposes, but this is a good opportunity to use a more recent and sophisticated result.

Lemma 17. *Let $\boldsymbol{C}_m$ denote the empirical covariance matrix based on an i.i.d. sample of size m from a distribution with support in $B(0,r)\subset\mathbb{R}^d$ with covariance $\boldsymbol{\Sigma}$. Then*

$$\mathbb{P}\left(\|\boldsymbol{C}_m-\boldsymbol{\Sigma}\| > r^2t\right) \le 4d\exp\left(-\frac{mt}{16}\min\left(\frac{t}{32},\frac{m}{d}\right)\right), \quad \forall t\ge 0. \tag{43}$$

Proof. Without loss of generality, we assume that the distribution has zero mean and is now supported on $B(0,2r)$. In fact, by a simple rescaling, we may also assume that $r=1$. Let $\boldsymbol{x}_1,\dots,\boldsymbol{x}_m$ denote the sample, with $\boldsymbol{x}_i=(x_{i,1},\dots,x_{i,d})$. We have

$$\boldsymbol{C}_m = \boldsymbol{C}_m^\star - \frac{1}{m}\bar{\boldsymbol{x}}\bar{\boldsymbol{x}}^T,$$

where

$$\boldsymbol{C}_m^\star := \frac{1}{m}\sum_{i=1}^m \boldsymbol{x}_i\boldsymbol{x}_i^T, \quad \bar{\boldsymbol{x}} := \frac{1}{m}\sum_{i=1}^m \boldsymbol{x}_i.$$

Note that

$$\|\boldsymbol{C}_m-\boldsymbol{\Sigma}\| \le \|\boldsymbol{C}_m^\star-\boldsymbol{\Sigma}\| + \frac{1}{m}\|\bar{\boldsymbol{x}}\|^2.$$

Applying the union bound and then Hoeffding's inequality to each coordinate—which is in $[-2,2]$—we get

$$\mathbb{P}(\|\bar{\boldsymbol{x}}\| > t) \leq \sum_{j=1}^{d} \mathbb{P}(|\bar{x}_j| > t/\sqrt{d}) \leq 2d \exp\left(-\frac{mt^2}{8d}\right).$$

Noting that $\frac{1}{m}(\boldsymbol{x}_i\boldsymbol{x}_i^T - \boldsymbol{\Sigma}), i = 1,\dots,m$, are independent, zero-mean, self-adjoint matrices with spectral norm bounded by $4/m$, we may apply the matrix Hoeffding inequality (Tropp, 2012, Th. 1.3), to get

$$\mathbb{P}\left(\|\boldsymbol{C}_m^\star - \boldsymbol{\Sigma}\| > t\right) \leq 2d \exp\left(-\frac{t^2}{8\sigma^2}\right), \quad \sigma^2 := m(4/m)^2 = 16/m.$$

Applying the union bound and using the previous inequalities, we arrive at

$$\begin{aligned}
\mathbb{P}\left(\|\boldsymbol{C}_m - \boldsymbol{\Sigma}\| > t\right) &\leq \mathbb{P}\left(\|\boldsymbol{C}_m^\star - \boldsymbol{\Sigma}\| > t/2\right) + \mathbb{P}\left(\|\bar{\boldsymbol{x}}\| > \sqrt{mt/2}\right) \\
&\leq 2d \exp\left(-\frac{mt^2}{512}\right) + 2d \exp\left(-\frac{m^2 t}{16d}\right) \\
&\leq 4d \exp\left(-\frac{mt}{16} \min\left(\frac{t}{32}, \frac{m}{d}\right)\right),
\end{aligned}$$

which concludes the proof. □

6.1.4 Projections

We relate below the difference of two orthogonal projections with the largest principal angle between the corresponding subspaces.

Lemma 18. *For two affine non-null subspaces T, T',*

$$\|\boldsymbol{P}_T - \boldsymbol{P}_{T'}\| = \begin{cases} \sin\theta_{\max}(T,T'), & \text{if } \dim(T) = \dim(T'), \\ 1, & \text{otherwise.} \end{cases}$$

Proof. For two affine subspaces $T, T' \subset \mathbb{R}^D$ of same dimension, let $\frac{\pi}{2} \geq \theta_1 \geq \cdots \geq \theta_D \geq 0$, denote the principal angles between them. By (Stewart and Sun, 1990, Th. I.5.5), the singular values of $\boldsymbol{P}_T - \boldsymbol{P}_{T'}$ are $\{\sin\theta_j : j = 1,\dots,q\}$, so that $\|\boldsymbol{P}_T - \boldsymbol{P}_{T'}\| = \max_j \sin\theta_j = \sin\theta_1 = \sin\theta_{\max}(T,T')$. Suppose now that T and T' are of different dimension, say $\dim(T) > \dim(T')$. We have $\|\boldsymbol{P}_T - \boldsymbol{P}_{T'}\| \leq \|\boldsymbol{P}_T\| \vee \|\boldsymbol{P}_{T'}\| = 1$, since $\boldsymbol{P}_T$ and $\boldsymbol{P}_{T'}$ are orthogonal projections and therefore positive semidefinite with operator norm equal to 1. Let $L = \boldsymbol{P}_T(T')$. Since $\dim(L) \leq \dim(T') < \dim(T)$, there is $\boldsymbol{u} \in T \cap L^\perp$ with $\boldsymbol{u} \neq 0$. Then $\boldsymbol{v}^\top \boldsymbol{u} = \boldsymbol{P}_T(\boldsymbol{v})^\top \boldsymbol{u} = 0$ for all $\boldsymbol{v} \in T'$, implying that $\boldsymbol{P}_{T'}(\boldsymbol{u}) = 0$ and consequently $(\boldsymbol{P}_T - \boldsymbol{P}_{T'})\boldsymbol{u} = \boldsymbol{u}$, so that $\|\boldsymbol{P}_T - \boldsymbol{P}_{T'}\| \geq 1$. □

The lemma below is a perturbation result for eigenspaces and widely known as the $\sin\Theta$ Theorem of Davis and Kahan (1970). See also (Luxburg, 2007, Th. 7) or (Stewart and Sun, 1990, Th. V.3.6).

Lemma 19. *Let $\boldsymbol{M}$ be positive semi-definite with eigenvalues $\beta_1 \geq \beta_2 \geq \cdots$. Suppose that $\beta_d - \beta_{d+1} > 0$. Then for any other positive semi-definite matrix $\boldsymbol{N}$,*

$$\|\boldsymbol{P}_{\boldsymbol{N}}^{(d)} - \boldsymbol{P}_{\boldsymbol{M}}^{(d)}\| \leq \frac{\sqrt{2}\|\boldsymbol{N} - \boldsymbol{M}\|}{\beta_d - \beta_{d+1}},$$

where $\boldsymbol{P}_{\boldsymbol{M}}^{(d)}$ and $\boldsymbol{P}_{\boldsymbol{M}}^{(d)}$ denote the orthogonal projections onto the top d eigenvectors of $\boldsymbol{M}$ and $\boldsymbol{N}$, respectively.

6.1.5 Intersections

We start with an elementary result on points near the intersection of two affine subspaces.

Lemma 20. *Take any two linear subspaces $T_1, T_2 \subset \mathbb{R}^D$. For any point $\boldsymbol{t}_1 \in T_1 \setminus T_2$, we have*

$$\operatorname{dist}(\boldsymbol{t}_1, T_2) \geq \operatorname{dist}(\boldsymbol{t}_1, T_1 \cap T_2) \sin\theta_{\min}(T_1, T_2).$$

Proof. We may reduce the problem to the case where $T_1 \cap T_2 = \{0\}$. Indeed, let $\tilde{T}_1 = T_1 \cap T_2^\perp$, $\tilde{T}_2 = T_1^\perp \cap T_2$ and $\tilde{\boldsymbol{t}}_1 = \boldsymbol{t}_1 - P_{T_1 \cap T_2}(\boldsymbol{t}_1)$. Then

$$\|\boldsymbol{t}_1 - P_{T_2}(\boldsymbol{t}_1)\| = \|\tilde{\boldsymbol{t}}_1 - P_{\tilde{T}_2}(\tilde{\boldsymbol{t}}_1)\|, \quad \|\boldsymbol{t}_1 - P_{T_1 \cap T_2}(\boldsymbol{t}_1)\| = \|\tilde{\boldsymbol{t}}_1\|,$$

and

$$\sin\theta_{\min}(T_1, T_2) = \sin\theta_{\min}(\tilde{T}_1, \tilde{T}_2).$$

So assume that $T_1 \cap T_2 = \{0\}$. By (Afriat, 1957, Th. 10.1), the angle formed by $\boldsymbol{t}_1$ and $P_{T_2}(\boldsymbol{t}_1)$ is at least as large as the smallest principal angle between T_1 and T_2, which is $\theta_{\min}(T_1, T_2)$ since $T_1 \cap T_2 = \{0\}$. From this the result follows immediately. □

The following result says that a point cannot be close to two compact and smooth surfaces intersecting at a positive angle without being close to their intersection. Note that the constant there cannot be solely characterized by κ, as it also depends on the separation between the surfaces away from their intersection.

Lemma 21. *Suppose $S_1, S_2 \in \mathcal{S}_d(\kappa)$ intersect at a strictly positive angle and that $\operatorname{reach}(S_1 \cap S_2) \geq 1/\kappa$. Then there is a constant C_{21} such that*

$$\operatorname{dist}(\boldsymbol{x}, S_1 \cap S_2) \leq C_{21} \max\left\{\operatorname{dist}(\boldsymbol{x}, S_1), \operatorname{dist}(\boldsymbol{x}, S_2)\right\}, \quad \forall \boldsymbol{x} \in \mathbb{R}^D.$$

Proof. Assume the result is not true, so there is a sequence $(\boldsymbol{x}_n) \subset \mathbb{R}^D$ such that $\operatorname{dist}(\boldsymbol{x}_n, S_1 \cap S_2) > n \max_k \operatorname{dist}(\boldsymbol{x}_n, S_k)$. Because the surfaces are bounded, we may assume WLOG that the sequence is bounded. Then $\operatorname{dist}(\boldsymbol{x}_n, S_1 \cap S_2)$ is bounded, which implies that $\max_k \operatorname{dist}(\boldsymbol{x}_n, S_k) = O(1/n)$. This also forces $\operatorname{dist}(\boldsymbol{x}_n, S_1 \cap S_2) \to 0$. Indeed, otherwise there is a constant $C > 0$ and a subsequence $(\boldsymbol{x}_{n'}) \subset (\boldsymbol{x}_n)$ such that $\operatorname{dist}(\boldsymbol{x}_{n'}, S_1 \cap S_2) \geq C$. Since $(\boldsymbol{x}_{n'})$ is bounded, there is a subsequence $(\boldsymbol{x}_{n''}) \subset (\boldsymbol{x}_{n'})$ that converges, and by the fact that $\max_k \operatorname{dist}(\boldsymbol{x}_{n''}, S_k) = o(1)$, and by compactness of S_k, the limit is necessarily in $S_1 \cap S_2$, which is incompatible with the fact that $\operatorname{dist}(\boldsymbol{x}_{n''}, S_1 \cap S_2) \geq C$. Since $\operatorname{dist}(\boldsymbol{x}_n, S_1 \cap S_2) \to 1$, we have

$$n \max_k \operatorname{dist}(\boldsymbol{x}_n, S_k) < \operatorname{dist}(\boldsymbol{x}_n, S_1 \cap S_2) = o(1). \tag{47}$$

Assume n is large enough that $\text{dist}(\boldsymbol{x}_n, S_1 \cap S_2) < 1/\kappa$ and let $\boldsymbol{s}_n^k$ be the projection of $\boldsymbol{x}_n$ onto S_k, and $\boldsymbol{s}_n^\ddagger$ the projection of $\boldsymbol{x}_n$ onto $S_1 \cap S_2$. Let $T_k = T_{S_k}(\boldsymbol{s}_n^\ddagger)$ and note that $\theta_{\min}(T_1, T_2) \geq \theta := \theta(S_1, S_2)$—see definition (6). Let $\boldsymbol{t}_n^k$ be the projection of $\boldsymbol{s}_n^k$ onto T_k. Assume WLOG that $\|\boldsymbol{t}_n^1 - \boldsymbol{s}_n^1\| \geq \|\boldsymbol{t}_n^2 - \boldsymbol{s}_n^2\|$. Let $\boldsymbol{t}_n$ denote the projection of $\boldsymbol{t}_n^1$ onto $T_1 \cap T_2$, and then let $\boldsymbol{s}_n = P_{S_1 \cap S_2}(\boldsymbol{t}_n)$.

By (47), we have

$$n \max_k \|\boldsymbol{x}_n - \boldsymbol{s}_n^k\| \leq \|\boldsymbol{x}_n - \boldsymbol{s}_n^\ddagger\| = o(1). \tag{48}$$

We start with the RHS:

$$\|\boldsymbol{x}_n - \boldsymbol{s}_n^\ddagger\| = \min_{\boldsymbol{s} \in S_1 \cap S_2} \|\boldsymbol{x}_n - \boldsymbol{s}\| \leq \|\boldsymbol{x}_n - \boldsymbol{s}_n\|, \tag{49}$$

and first show that $\|\boldsymbol{x}_n - \boldsymbol{s}_n\| = o(1)$ too. We use the triangle inequality multiple times in what follows. We have

$$\|\boldsymbol{x}_n - \boldsymbol{s}_n\| \leq \|\boldsymbol{x}_n - \boldsymbol{s}_n^1\| + \|\boldsymbol{s}_n^1 - \boldsymbol{t}_n^1\| + \|\boldsymbol{t}_n^1 - \boldsymbol{t}_n\| + \|\boldsymbol{t}_n - \boldsymbol{s}_n\|. \tag{50}$$

From (48), $\|\boldsymbol{x}_n - \boldsymbol{s}_n^1\| \leq \|\boldsymbol{x}_n - \boldsymbol{s}_n^\ddagger\| = o(1)$, so that $\|\boldsymbol{s}_n^1 - \boldsymbol{s}_n^\ddagger\| = o(1)$, and by (10),

$$\|\boldsymbol{s}_n^1 - \boldsymbol{t}_n^1\| \leq \frac{\kappa}{2}\|\boldsymbol{s}_n^1 - \boldsymbol{s}_n^\ddagger\|^2 \leq \kappa(\|\boldsymbol{s}_n^1 - \boldsymbol{x}_n\|^2 + \|\boldsymbol{x}_n - \boldsymbol{s}_n^\ddagger\|^2) = o(1). \tag{51}$$

We also have

$$\|\boldsymbol{t}_n^1 - \boldsymbol{t}_n\| = \min_{\boldsymbol{t} \in T_1 \cap T_2} \|\boldsymbol{t}_n^1 - \boldsymbol{t}\| \leq \|\boldsymbol{t}_n^1 - \boldsymbol{s}_n^\ddagger\| \leq \|\boldsymbol{t}_n^1 - \boldsymbol{s}_n^1\| + \|\boldsymbol{s}_n^1 - \boldsymbol{x}_n\| + \|\boldsymbol{x}_n - \boldsymbol{s}_n^\ddagger\| = o(1). \tag{52}$$

Finally,

$$\|\boldsymbol{t}_n - \boldsymbol{s}_n\| = \min_{\boldsymbol{s} \in S_1 \cap S_2} \|\boldsymbol{t}_n - \boldsymbol{s}\| \leq \|\boldsymbol{t}_n - \boldsymbol{s}_n^\ddagger\| \leq \|\boldsymbol{t}_n - \boldsymbol{t}_n^1\| + \|\boldsymbol{t}_n^1 - \boldsymbol{s}_n^\ddagger\| = o(1).$$

We now proceed. The last upper bound is rather crude. Indeed, using (12) for $S = S_1 \cap S_2$ and $\boldsymbol{s} = \boldsymbol{s}_n^\ddagger$, and noting that $T_{S_1 \cap S_2}(\boldsymbol{s}_n^\ddagger) = T_1 \cap T_2$, and using the fact that $\|\boldsymbol{t}_n - \boldsymbol{s}_n^\ddagger\| = o(1)$, we get

$$\|\boldsymbol{t}_n - \boldsymbol{s}_n\| \leq \kappa\|\boldsymbol{t}_n - \boldsymbol{s}_n^\ddagger\|^2 \leq \kappa(\|\boldsymbol{t}_n - \boldsymbol{s}_n\| + \|\boldsymbol{s}_n - \boldsymbol{x}_n\| + \|\boldsymbol{x}_n - \boldsymbol{s}_n^\ddagger\|)^2.$$

Using (49),

$$\|\boldsymbol{t}_n - \boldsymbol{s}_n\| \leq \kappa(\|\boldsymbol{t}_n - \boldsymbol{s}_n\| + 2\|\boldsymbol{s}_n - \boldsymbol{x}_n\|)^2 \leq 5\kappa\|\boldsymbol{x}_n - \boldsymbol{s}_n\|^2, \tag{53}$$

eventually, since $\|\boldsymbol{t}_n - \boldsymbol{s}_n\| = o(1)$.

Combining (49), (50), (51) and (53), we get

$$\|\boldsymbol{x}_n - \boldsymbol{s}_n\| \leq \|\boldsymbol{x}_n - \boldsymbol{s}_n^1\| + O(\|\boldsymbol{x}_n - \boldsymbol{s}_n^1\|^2 + \|\boldsymbol{x}_n - \boldsymbol{s}_n\|^2) + \|\boldsymbol{t}_n^1 - \boldsymbol{t}_n\| + O(\|\boldsymbol{x}_n - \boldsymbol{s}_n\|^2),$$

which leads to

$$\|\boldsymbol{x}_n - \boldsymbol{s}_n\| \leq 2\|\boldsymbol{x}_n - \boldsymbol{s}_n^1\| + 2\|\boldsymbol{t}_n^1 - \boldsymbol{t}_n\|, \tag{54}$$

when n is large enough. Using this bound in (48) combined with (49), we get

$$\|\boldsymbol{t}_n^1 - \boldsymbol{t}_n\| \geq \frac{n-2}{2} \max_k \|\boldsymbol{x}_n - \boldsymbol{s}_n^k\|.$$

We then have

$$\begin{aligned}
\max_k \|\boldsymbol{x}_n - \boldsymbol{s}_n^k\| &\geq \frac{1}{2}\|\boldsymbol{s}_n^1 - \boldsymbol{s}_n^2\| \\
&\geq \frac{1}{2}(\|\boldsymbol{t}_n^1 - \boldsymbol{t}_n^2\| - \|\boldsymbol{s}_n^1 - \boldsymbol{t}_n^1\| - \|\boldsymbol{s}_n^2 - \boldsymbol{t}_n^2\|) \\
&\geq \frac{1}{2}\,\text{dist}(\boldsymbol{t}_n^1, T_2) - \|\boldsymbol{s}_n^1 - \boldsymbol{t}_n^1\|,
\end{aligned}$$

with

$$\|\boldsymbol{s}_n^1 - \boldsymbol{t}_n^1\| = O(\|\boldsymbol{x}_n - \boldsymbol{s}_n^1\|^2 + \|\boldsymbol{x}_n - \boldsymbol{s}_n^\ddagger\|^2) = O(\|\boldsymbol{x}_n - \boldsymbol{s}_n\|^2) = O(\|\boldsymbol{t}_n^1 - \boldsymbol{t}_n\|^2),$$

due (in the same order) to (51), (48)-(49), and (54). Recalling that $\|\boldsymbol{t}_n^1 - \boldsymbol{t}_n\| = \text{dist}(\boldsymbol{t}_n^1, T_1 \cap T_2)$, we conclude that

$$\text{dist}(\boldsymbol{t}_n^1, T_2) = O(1/n)\,\text{dist}(\boldsymbol{t}_n^1, T_1 \cap T_2) + O(1)\,\text{dist}(\boldsymbol{t}_n^1, T_1 \cap T_2)^2.$$

However, by Lemma 20, we have $\text{dist}(\boldsymbol{t}_n^1, T_2) \geq (\sin\theta)\,\text{dist}(\boldsymbol{t}_n^1, T_1 \cap T_2)$, so that dividing by $\text{dist}(\boldsymbol{t}_n^1, T_1 \cap T_2)$ above leads to

$$1 = O(1/n) + O(1)\,\text{dist}(\boldsymbol{t}_n^1, T_1 \cap T_2),$$

which is in contradiction with the fact that $\text{dist}(\boldsymbol{t}_n^1, T_2) \leq \|\boldsymbol{t}_n^1 - \boldsymbol{t}_n\| = o(1)$, established in (52). □

6.1.6 Covariances near an Intersection

The following compares a covariance matrix of a distribution supported on the union of two smooth surfaces with that of the projection of that distribution on tangent subspaces.

Lemma 22. *Consider $S_1, S_2 \in \mathcal{S}_d(\kappa)$ intersecting at a positive angle, with $\text{reach}(S_1 \cap S_2) \geq 1/\kappa$. Fix $\boldsymbol{s} \in S_1$ and let $A = B(\boldsymbol{s}, r) \cap (S_1 \cup S_2)$, where $r < 1/\kappa$. Let $\boldsymbol{C}$ denote the empirical covariance of an i.i.d. sample from a distribution ν supported on A of size m and define $\boldsymbol{\Sigma} = \text{Cov}(\nu)$. Then*

$$\mathbb{P}\left(\|\boldsymbol{C}_m - \boldsymbol{\Sigma}\| > r^2 t + 3\kappa r^3\right) \leq C_{22} \exp\left(-\frac{mt}{C_{22}} \min(t, m)\right), \quad \forall t \geq 0,$$

where C_{22} depends only on d.

Proof. Let $T_1 = T_{S_1}(\boldsymbol{s})$, and if $\text{dist}(\boldsymbol{s}, S_2) \leq r$, let $\boldsymbol{s}_2 = P_{S_2}(\boldsymbol{s})$ and $T_2 = T_{S_2}(\boldsymbol{s}_2)$. Define $g : A \mapsto T_1 \cup T_2$ where $g(\boldsymbol{x}) = P_{T_1}(\boldsymbol{x})$ if $\boldsymbol{x} \in S_1$, and $g(\boldsymbol{x}) = P_{T_2}(\boldsymbol{x})$ otherwise. By (10), $\|g(\boldsymbol{x}) - \boldsymbol{x}\| \leq \frac{\kappa}{2} r^2$ for all $\boldsymbol{x} \in A$. Let $\boldsymbol{\Sigma}^g = \text{Cov}(\nu^g)$. Also, let ν_m denote the sample distribution and note that $\boldsymbol{C}_m = \text{Cov}(\nu_m)$. Let $\boldsymbol{C}_m^g = \text{Cov}(\nu_m^g)$. Note that ν_m^g is the empirical distribution of an i.i.d. sample of size m from ν^g, so the notation is congruent. If $\text{dist}(\boldsymbol{s}, S_2) > r$, ν^g is supported on $B(\boldsymbol{s}, r) \cap T_1$, which is a d-dimensional ball of radius r, so that (43) applies to $\boldsymbol{C}_m^g$ and $\boldsymbol{\Sigma}^g$. If $\text{dist}(\boldsymbol{s}, S_2) \leq r$, ν^g is supported on

$$(B(\boldsymbol{s}, r) \cap T_1) \cup (B(\boldsymbol{s}_2, r) \cap T_2) \subset B(\boldsymbol{s}, 2r) \cap (T_1 \cup T_2) \subset B(\boldsymbol{s}, 2r) \cap (T_1 + T_2),$$

where the latter is a ball in dimension $d' := \dim(T_1 + T_2) \le 2d$ of radius $2r$, so that (43)—with $2r$ in place of r and $2d$ in place of d—applies to $\boldsymbol{C}_m^g$ and $\boldsymbol{\Sigma}^g$. Then, by the triangle inequality and Lemma 15,

$$\begin{aligned}\|\boldsymbol{C}_m - \boldsymbol{\Sigma}\| &\le \|\boldsymbol{C}_m - \boldsymbol{C}_m^g\| + \|\boldsymbol{C}_m^g - \boldsymbol{\Sigma}^g\| + \|\boldsymbol{\Sigma}^g - \boldsymbol{\Sigma}\| \\ &\le 2\kappa r^3 + \kappa^2 r^4/2 + \|\boldsymbol{C}_m^g - \boldsymbol{\Sigma}^g\|,\end{aligned}$$

as in (38). □

Here is a continuity result.

Lemma 23. *Let T_1 and T_2 be two linear subspaces of same dimension d. For $\boldsymbol{x} \in T_1$, denote by $\boldsymbol{\Sigma}_T(\boldsymbol{x})$ the covariance matrix of the uniform distribution over $B(\boldsymbol{x}, r) \cap (T_1 \cup T_2)$. Then, for all $\boldsymbol{x}, \boldsymbol{y} \in T_1$,*

$$\|\boldsymbol{\Sigma}_T(\boldsymbol{x}) - \boldsymbol{\Sigma}_T(\boldsymbol{y})\| \le \begin{cases} 5dr\,\|\boldsymbol{x} - \boldsymbol{y}\|, & \text{if } d \ge 2, \\ r\,\|\boldsymbol{x} - \boldsymbol{y}\| + 2\sqrt{r^3\,\|\boldsymbol{x} - \boldsymbol{y}\|}, & \text{if } d = 1. \end{cases}$$

We note that, when $d = 1$, we also have the bound

$$\|\boldsymbol{\Sigma}_T(\boldsymbol{x}) - \boldsymbol{\Sigma}_T(\boldsymbol{y})\| \le 4\gamma r\,\|\boldsymbol{x} - \boldsymbol{y}\|, \quad \text{if } \min(\operatorname{dist}(\boldsymbol{x}, T_2), \operatorname{dist}(\boldsymbol{y}, T_2)) \le r\sqrt{1 - 1/\gamma^2}.$$

Proof. Assume without loss of generality that $r = 1$. Let $A_{\boldsymbol{x}}^j = B(\boldsymbol{x}, 1) \cap T_j$ for any $\boldsymbol{x}$ and $j = 1, 2$. By Lemma 14 and then Lemma 7, we have

$$\begin{aligned}\|\boldsymbol{\Sigma}_T(\boldsymbol{x}) - \boldsymbol{\Sigma}_T(\boldsymbol{y})\| &= \|\operatorname{Cov}(\lambda_{A_{\boldsymbol{x}}^1 \cup A_{\boldsymbol{x}}^2}) - \operatorname{Cov}(\lambda_{A_{\boldsymbol{y}}^1 \cup A_{\boldsymbol{y}}^2})\| \\ &\le 3d\,\mathrm{TV}(\lambda_{A_{\boldsymbol{x}}^1 \cup A_{\boldsymbol{x}}^2}, \lambda_{A_{\boldsymbol{y}}^1 \cup A_{\boldsymbol{y}}^2}) \\ &\le 12d\,\frac{\operatorname{vol}\left((A_{\boldsymbol{x}}^1 \cup A_{\boldsymbol{x}}^2) \triangle (A_{\boldsymbol{y}}^1 \cup A_{\boldsymbol{y}}^2)\right)}{\operatorname{vol}\left((A_{\boldsymbol{x}}^1 \cup A_{\boldsymbol{x}}^2) \cup (A_{\boldsymbol{y}}^1 \cup A_{\boldsymbol{y}}^2)\right)} \\ &\le 12d\,\frac{\operatorname{vol}(A_{\boldsymbol{x}}^1 \triangle A_{\boldsymbol{y}}^1) + \operatorname{vol}(A_{\boldsymbol{x}}^2 \triangle A_{\boldsymbol{y}}^2)}{\operatorname{vol}(A_{\boldsymbol{x}}^1)}.\end{aligned}$$

Note that $A_{\boldsymbol{x}}^2 = B(\boldsymbol{x}_2, \eta) \cap T_2$ where $\boldsymbol{x}_2 := P_{T_2}(\boldsymbol{x})$ and $\eta := \sqrt{1 - \|\boldsymbol{x} - \boldsymbol{x}_2\|^2 \wedge 1}$. Similarly, $A_{\boldsymbol{y}}^2 = B(\boldsymbol{y}_2, \delta) \cap T_2$ where $\boldsymbol{y}_2 := P_{T_2}(\boldsymbol{y})$ and $\delta := \sqrt{1 - \|\boldsymbol{y} - \boldsymbol{y}_2\|^2 \wedge 1}$. Therefore, applying Lemma 12, we get

$$\frac{\operatorname{vol}(A_{\boldsymbol{x}}^1 \triangle A_{\boldsymbol{y}}^1)}{2\operatorname{vol}(A_{\boldsymbol{x}}^1)} \le 1 - (1 - t)_+^d, \tag{64}$$

and assuming WLOG that $\delta \le \eta$ (i.e., $\|\boldsymbol{y} - \boldsymbol{y}_2\| \ge \|\boldsymbol{x} - \boldsymbol{x}_2\|$) and after proper scaling, we get

$$\frac{\operatorname{vol}(A_{\boldsymbol{x}}^2 \triangle A_{\boldsymbol{y}}^2)}{2\operatorname{vol}(A_{\boldsymbol{x}}^1)} \le \zeta := \eta^d - (\eta - t_2)_+^d \wedge \delta^d, \tag{65}$$

where $t := \|\boldsymbol{x} - \boldsymbol{y}\|$ and $t_2 := \|\boldsymbol{x}_2 - \boldsymbol{y}_2\|$. Note that $t_2 \le t$ by the fact that P_{T_2} is 1-Lipschitz.

For (64), we have $1 - (1 - t)_+^d \le dt$. This is obvious when $t \ge 1$, while when $t \le 1$ it is obtained using the fact that, for any $0 \le a < b \le 1$,

$$b^d - a^d = (b - a)(b^{d-1} + ab^{d-2} + \cdots + a^{d-2}b + a^{d-1}) \le db^{d-1}(b - a) \le d(b - a). \tag{66}$$

For (65), we consider several cases.

- When $\eta \le t_2$, then $\zeta = \eta^d \le \eta \le t_2 \le t$.
- When $t_2 < \eta \le t_2 + \delta$, then $\zeta = \eta^d - (\eta - t_2)^d \le dt_2 \le dt$, by (66).
- When $\eta \ge t_2 + \delta$ and $d \ge 2$, we have

$$\begin{aligned}\zeta &= \eta^d - \delta^d \le d\eta^{d-1}(\eta - \delta) \le d\eta(\eta - \delta) \le d(\eta^2 - \delta^2) \\ &= d(\|\boldsymbol{y} - \boldsymbol{y}_2\|^2 \wedge 1 - \|\boldsymbol{x} - \boldsymbol{x}_2\|^2 \wedge 1) \\ &= d(\|\boldsymbol{y} - \boldsymbol{y}_2\| \wedge 1 + \|\boldsymbol{x} - \boldsymbol{x}_2\| \wedge 1)(\|\boldsymbol{y} - \boldsymbol{y}_2\| \wedge 1 - \|\boldsymbol{x} - \boldsymbol{x}_2\| \wedge 1) \\ &\le 2d(t + t_2) \le 4dt,\end{aligned}$$

  where (66) was applied in the first line and the triangle inequality was applied in the last line, in the form of

$$\|\boldsymbol{y} - \boldsymbol{y}_2\| \le \|\boldsymbol{y} - \boldsymbol{x}\| + \|\boldsymbol{x} - \boldsymbol{x}_2\| + \|\boldsymbol{x}_2 - \boldsymbol{y}_2\| = \|\boldsymbol{x} - \boldsymbol{x}_2\| + t + t_2.$$

- When $\eta \ge t_2 + \delta$ and $d = 1$, we have

$$\zeta = \eta - \delta \le \sqrt{\|\boldsymbol{y} - \boldsymbol{y}_2\|^2 \wedge 1 - \|\boldsymbol{x} - \boldsymbol{x}_2\|^2 \wedge 1} \le 2\sqrt{t},$$

  using (70), preceded by the fact that, for any $0 \le a < b \le 1$,

$$0 \le \sqrt{1 - a} - \sqrt{1 - b} = \frac{b - a}{\sqrt{1 - a} + \sqrt{1 - b}} \le \frac{b - a}{\sqrt{1 - b + b - a}} \le \frac{b - a}{\sqrt{b - a}} = \sqrt{b - a}.$$

In summary, when $d \ge 2$, we can therefore bound $\|\boldsymbol{\Sigma}_T(\boldsymbol{x}) - \boldsymbol{\Sigma}_T(\boldsymbol{y})\|$ by $dt + 4dt = 5dt$; and when $d = 1$, we bound that by $t + 2\sqrt{t}$. □

The following is in some sense a converse to Lemma 23, in that we lower-bound the distance between covariance matrices near an intersection of linear subspaces. Note that the covariance matrix does not change when moving parallel to the intersection; however, it does when moving perpendicular to the intersection.

Lemma 24. *Let T_1 and T_2 be two linear subspaces of same dimension d with $\theta_{\min}(T_1, T_2) \ge \theta_0 > 0$. Fix a unit norm vector $\boldsymbol{v} \in T_1 \cap (T_1 \cap T_2)^\perp$. With $\boldsymbol{\Sigma}_T(h\boldsymbol{v})$ denoting the covariance of the uniform distribution over $B(h\boldsymbol{v}, r) \cap (T_1 \cup T_2)$, we have*

$$\inf_h \sup_\ell \|\boldsymbol{\Sigma}_T(h\boldsymbol{v}) - \boldsymbol{\Sigma}_T(\ell\boldsymbol{v})\| \ge r^2/C_{24},$$

where the infimum is over $0 \le h \le 1/\sin\theta_0$ and the supremum over $\max(0, h - 1/2) \le \ell \le \min(1/\sin\theta_0, h + 1/2)$, and $C_{24} \ge 1$ depends only on d and θ_0.

Proof. Assume without loss of generality that $r = 1$. If the statement of the lemma is not true, there are subspaces T_1 and T_2 of same dimension d, a unit length vector $\boldsymbol{v} \in T_1 \cap (T_1 \cap T_2)^\perp$ and $0 \le h \le 1/\sin\theta_0$, such that

$$\boldsymbol{\Sigma}_T(\ell\boldsymbol{v}) = \boldsymbol{\Sigma}_T(h\boldsymbol{v}) \text{ for all } \max(0, h - 1/2) \le \ell \le \min(1/\sin\theta_0, h + 1/2). \tag{71}$$

This is because $\ell \mapsto \mathbf{\Sigma}_T(\ell \boldsymbol{v})$ is continuous once T_1, T_2 and $\boldsymbol{v}$ are chosen. By projecting onto $(T_1 \cap T_2)^\perp$, we may assume that $T_1 \cap T_2 = \{0\}$ without loss of generality. Let $\theta = \angle(\boldsymbol{v}, T_2)$ and note that $\theta \geq \theta_0$ since $T_1 \cap T_2 = \{0\}$. Define $\boldsymbol{u} = (\boldsymbol{v} - P_{T_2}\boldsymbol{v})/\sin\theta$ and also $\boldsymbol{w} = P_{T_2}\boldsymbol{v}/\cos\theta$ when $\theta < \pi/2$, and $\boldsymbol{w} \in T_2$ is any vector perpendicular to $\boldsymbol{v}$ when $\theta = \pi/2$. $B(h\boldsymbol{v}, 1) \cap T_1$ is the d-dimensional ball of T_1 of radius 1 and center $h\boldsymbol{v}$, while—using Pythagoras theorem—$B(h\boldsymbol{v}, 1) \cap T_2$ is the d-dimensional ball of T_2 of radius $t := (1 - (h\sin\theta)^2)^{1/2}$ and center $(h\cos\theta)\boldsymbol{w}$. Let λ denote the uniform distribution over $B(h\boldsymbol{v}, 1) \cap (T_1 \cup T_2)$ and λ_k the uniform distribution over $B(h\boldsymbol{v}, 1) \cap T_k$. Note that $\lambda = \alpha\lambda_1 + (1-\alpha)\lambda_2$, where

$$\alpha := \frac{\mathrm{vol}(B(h\boldsymbol{v}, 1) \cap T_1)}{\mathrm{vol}(B(h\boldsymbol{v}, 1) \cap (T_1 \cup T_2))} = \frac{\mathrm{vol}(B(h\boldsymbol{v}, 1) \cap T_1)}{\mathrm{vol}(B(h\boldsymbol{v}, 1) \cap T_1) + \mathrm{vol}(B((h\cos\theta)\boldsymbol{w}, t) \cap T_2)} = \frac{1}{1+t^d}.$$

By the law of total covariance,

$$\mathrm{Cov}(\lambda) = \alpha\,\mathrm{Cov}(\lambda_1) + (1-\alpha)\,\mathrm{Cov}(\lambda_2) + \alpha(1-\alpha)(\mathbb{E}(\lambda_1) - \mathbb{E}(\lambda_2))(\mathbb{E}(\lambda_1) - \mathbb{E}(\lambda_2))^\top. \quad (72)$$

with $\mathbb{E}(\lambda_1) = h\boldsymbol{v}$ and $\mathbb{E}(\lambda_2) = h\cos(\theta)\boldsymbol{w}$, and $\mathrm{Cov}(\lambda_1) = c\boldsymbol{P}_{T_1}$ and $\mathrm{Cov}(\lambda_2) = t^2 c\boldsymbol{P}_{T_2}$, by Lemma 13. Hence,

$$\mathbf{\Sigma}_T(h\boldsymbol{v}) = \alpha c\boldsymbol{P}_{T_1} + (1-\alpha)t^2 c\boldsymbol{P}_{T_2} + \alpha(1-\alpha)(1-t^2)\boldsymbol{u}\boldsymbol{u}^\top,$$

using the fact that $\boldsymbol{v} - (\cos\theta)\boldsymbol{w} = (\sin\theta)\boldsymbol{u}$ and the definition of t. Let $\theta_1 = \theta_{\max}(T_1, T_2)$ and let $\boldsymbol{v}_1 \in T_1$ be of unit length and such that $\angle(\boldsymbol{v}_1, T_2) = \theta_1$. Then for any $0 \leq h, \ell \leq 1/\sin\theta_0$, we have

$$\|\mathbf{\Sigma}_T(h\boldsymbol{v}) - \mathbf{\Sigma}_T(\ell\boldsymbol{v})\| \geq |\boldsymbol{v}_1^\top \mathbf{\Sigma}_T(h\boldsymbol{v})\boldsymbol{v}_1 - \boldsymbol{v}_1^\top \mathbf{\Sigma}_T(\ell\boldsymbol{v})\boldsymbol{v}_1| = |f(t_h) - f(t_\ell)|, \quad (73)$$

where $t_h := (1 - (h\sin\theta)^2)^{1/2}$ and

$$f(t) \;=\; \frac{c}{1+t^d} + \frac{ct^{d+2}(\cos\theta_1)^2}{1+t^d} + \frac{t^d(1-t^2)(\boldsymbol{u}^\top\boldsymbol{v}_1)^2}{(1+t^d)^2}.$$

It is easy to see that the interval

$$I_h = \{t_\ell : (h - 1/2)_+ \leq \ell \leq (1/\sin\theta_0) \wedge (h + 1/2)\}$$

is non empty. Because of (71) and (73), $f(t)$ is constant over $t \in I_h$, but this is not possible since f is a rational function not equal to a constant and therefore cannot be constant over an interval of positive length. □

We now look at the eigenvalues of the covariance matrix.

Lemma 25. *Let T_1 and T_2 be two linear subspaces of same dimension d. For $\boldsymbol{x} \in T_1$, denote by $\mathbf{\Sigma}_T(\boldsymbol{x})$ the covariance matrix of the uniform distribution over $B(\boldsymbol{x}, 1) \cap (T_1 \cup T_2)$. Then, for all $\boldsymbol{x} \in T_1$,*

$$c\big(1 - (1-\delta^2(\boldsymbol{x}))_+^{d/2}\big) \leq \beta_d(\mathbf{\Sigma}_T(\boldsymbol{x})), \qquad \beta_1(\mathbf{\Sigma}_T(\boldsymbol{x})) \leq c + \delta^2(\boldsymbol{x})(1-\delta^2(\boldsymbol{x}))_+^{d/2}, \quad (75)$$

$$\frac{c}{8}(1 - \cos\theta_{\max}(T_1, T_2))^2(1-\delta^2(\boldsymbol{x}))_+^{d/2+1} \leq \beta_{d+1}(\mathbf{\Sigma}_T(\boldsymbol{x})) \leq (c + \delta^2(\boldsymbol{x}))(1-\delta^2(\boldsymbol{x}))_+^{d/2}, \quad (76)$$

where $c := 1/(d+2)$ and $\delta(\boldsymbol{x}) := \mathrm{dist}(\boldsymbol{x}, T_2)$.

Proof. Applying (72), we have

$$\mathbf{\Sigma}_T(x) = \alpha c\boldsymbol{P}_{T_1} + (1-\alpha)ct^2\boldsymbol{P}_{T_2} + \alpha(1-\alpha)(\boldsymbol{x} - \boldsymbol{x}_2)(\boldsymbol{x} - \boldsymbol{x}_2)^\top, \quad (77)$$

where $\boldsymbol{x}_2 := P_{T_2}(\boldsymbol{x})$ and $\alpha := (1+t^d)^{-1}$ with $t := (1 - \delta^2(\boldsymbol{x}))_+^{1/2}$. Because all the matrices in this display are positive semidefinite, we have

$$\beta_d(\mathbf{\Sigma}_T(\boldsymbol{x})) \geq \alpha c\|\boldsymbol{P}_{T_1}\| = \alpha c,$$

with $\alpha \geq 1 - t^d$. And because of the triangle inequality, we have

$$\beta_1(\mathbf{\Sigma}_T(\boldsymbol{x})) \leq \alpha c\|\boldsymbol{P}_{T_1}\| + (1-\alpha)ct^2\|\boldsymbol{P}_{T_2}\| + \alpha(1-\alpha)\|\boldsymbol{x} - \boldsymbol{x}_2\|^2 \leq c + \alpha(1-\alpha)\delta^2(\boldsymbol{x}),$$

with $\alpha(1-\alpha) \leq t^d$. Hence, (75) is proved.

For the upper bound in (76), by Weyl's inequality (Stewart and Sun, 1990, Cor. IV.4.9) and the fact that $\beta_{d+1}(\boldsymbol{P}_{T_1}) = 0$, and then the triangle inequality, we get

$$\begin{aligned}\beta_{d+1}(\mathbf{\Sigma}_T(\boldsymbol{x})) &\leq \|\mathbf{\Sigma}_T(\boldsymbol{x}) - \alpha c\boldsymbol{P}_{T_1}\| \\ &\leq c(1-\alpha)t^2\|\boldsymbol{P}_{T_2}\| + \alpha(1-\alpha)\delta^2(\boldsymbol{x}) \\ &\leq (1-\alpha)(c + \delta^2(\boldsymbol{x})),\end{aligned}$$

and we then use the fact that $1 - \alpha \leq t^d$.

For the lower bound in (76), let $\theta_1 \geq \theta_2 \geq \cdots \geq \theta_d$ denote the principal angles between T_1 and T_2. By the definition of principal angles, there are orthonormal bases for T_1 and T_2, denoted $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d$ and $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_d$, such that $\boldsymbol{v}_j^\top \boldsymbol{w}_k = \mathbb{1}_{j=k} \cdot \cos\theta_j$. Take $\boldsymbol{u} \in \mathrm{span}(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d, \boldsymbol{w}_1)$, that is, of the form $\boldsymbol{u} = a\boldsymbol{v}_1 + \boldsymbol{v} + b\boldsymbol{w}_1$, with $\boldsymbol{v} \in \mathrm{span}(\boldsymbol{v}_2, \ldots, \boldsymbol{v}_d)$. Since $\boldsymbol{P}_{T_1} = \boldsymbol{v}_1\boldsymbol{v}_1^\top + \cdots + \boldsymbol{v}_d\boldsymbol{v}_d^\top$ and $\boldsymbol{P}_{T_2} = \boldsymbol{w}_1\boldsymbol{w}_1^\top + \cdots + \boldsymbol{w}_d\boldsymbol{w}_d^\top$, when $\|\boldsymbol{u}\| = 1$, we have

$$\frac{1}{c}\boldsymbol{u}^\top \mathbf{\Sigma}_T(\boldsymbol{x})\boldsymbol{u} \geq \alpha\boldsymbol{u}^\top \boldsymbol{P}_{T_1}\boldsymbol{u} + (1-\alpha)t^2\boldsymbol{u}^\top \boldsymbol{P}_{T_2}\boldsymbol{u}$$

with

$$\boldsymbol{u}^\top \boldsymbol{P}_{T_1}\boldsymbol{u} = a^2 + \|\boldsymbol{v}\|^2 + 2ab\cos\theta_1 + b^2\cos^2\theta_1 = 1 - b^2\sin^2\theta_1 \geq 0,$$

and

$$\boldsymbol{u}^\top \boldsymbol{P}_{T_2}\boldsymbol{u} = b^2 + 2ab\cos\theta_1 + a^2\cos^2\theta_1 = (a\cos\theta_1 + b)^2.$$

If $|b| \leq 1/2$, then $\boldsymbol{u}^\top \boldsymbol{P}_{T_1}\boldsymbol{u} \geq 3/4$, implying that $\frac{1}{c}\boldsymbol{u}^\top \mathbf{\Sigma}_T(\boldsymbol{x})\boldsymbol{u} \geq 3\alpha/4 \geq 3/8$. Otherwise,

$$\begin{aligned}\boldsymbol{u}^\top \boldsymbol{P}_{T_1}\boldsymbol{u} + \boldsymbol{u}^\top \boldsymbol{P}_{T_2}\boldsymbol{u} &\geq (a + b\cos\theta_1)^2 + (a\cos\theta_1 + b)^2 \\ &\geq (1 - \cos\theta_1)^2(a^2 + b^2) \geq (1 - \cos\theta_1)^2/4,\end{aligned}$$

and using the fact that $\alpha \geq 1 - \alpha \geq (1-\alpha)t^2$, we get

$$\frac{1}{c}\boldsymbol{u}^\top \mathbf{\Sigma}_T(\boldsymbol{x})\boldsymbol{u} \geq (1-\alpha)t^2(1 - \cos\theta_1)^2/4.$$

This last bound is always worst. Hence, by the Courant-Fischer theorem (Stewart and Sun, 1990, Cor. IV.4.7), we have

$$\beta_{d+1}(\mathbf{\Sigma}_T(\boldsymbol{x})) \geq \frac{c}{4}(1-\alpha)t^2(1 - \cos\theta_1)^2,$$

and we conclude with $1 - \alpha \geq t^d/2$. □

Below are two technical results on the covariance matrix of the uniform distribution on the intersection of a ball and the union of two smooth surfaces, near where the surfaces intersect. The first one is in fact a stepping stone to the second. The latter will enable us to apply Lemma 23.

Lemma 26. *There is a constant $C_{26} \geq 3$ such that the following holds. Consider $S_1, S_2 \in \mathcal{S}_d(\kappa)$. Fix $r \leq \frac{1}{C_{26}\kappa}$, and for $\boldsymbol{s} \in S_1$ with $\mathrm{dist}(\boldsymbol{s}, S_2) < 1/\kappa$, let $\boldsymbol{\Sigma}(\boldsymbol{s})$ and $\boldsymbol{\Sigma}_T(\boldsymbol{s})$ denote the covariance matrices of the uniform distributions over $B(\boldsymbol{s}, r) \cap (S_1 \cup S_2)$ and $B(\boldsymbol{s}, r) \cap (T_1 \cup T_2)$, respectively, where $T_1 := T_{S_1}(\boldsymbol{s})$ and $T_2 := T_{S_2}(P_{S_2}(\boldsymbol{s}))$. Then*

$$\|\boldsymbol{\Sigma}(\boldsymbol{s}) - \boldsymbol{\Sigma}_T(\boldsymbol{s})\| \leq C_{26}\kappa r^3. \tag{83}$$

Proof. We note that it is enough to prove the result when r is small enough. Let $\delta = \mathrm{dist}(\boldsymbol{s}, S_2)$ and let $\boldsymbol{s}_2 = P_{S_2}(\boldsymbol{s})$, so that $\|\boldsymbol{s} - \boldsymbol{s}_2\| = \delta$. We first treat the case where $\delta \leq r$. Let B_r be short for $B(\boldsymbol{s}, r)$ and define $A_k = B_r \cap S_k$, $\boldsymbol{\mu}_k = \mathbb{E}(\lambda_{A_k})$ and $\boldsymbol{D}_k = \mathrm{Cov}(\lambda_{A_k})$, for $k = 1, 2$. Applying (72), we have

$$\boldsymbol{\Sigma}(\boldsymbol{s}) = \alpha \boldsymbol{D}_1 + (1-\alpha)\boldsymbol{D}_2 + \alpha(1-\alpha)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top,$$

where

$$\alpha := \frac{\mathrm{vol}(A_1)}{\mathrm{vol}(A_1) + \mathrm{vol}(A_2)}.$$

Recall that $T_1 = T_{S_1}(\boldsymbol{s})$ and $T_2 = T_{S_2}(\boldsymbol{s}_2)$, and define $A'_k = B_r \cap T_k$, so that $B_r \cap (T_1 \cup T_2) = A'_1 \cup A'_2$. Note that $\mathbb{E}(\lambda_{A'_1}) = \boldsymbol{s}$ and $\mathbb{E}(\lambda_{A'_2}) = \boldsymbol{s}_2$, and by Lemma 13, $\boldsymbol{D}'_1 := \mathrm{Cov}(\lambda_{A'_1}) = cr^2 \boldsymbol{P}_{T_1}$ and $\boldsymbol{D}'_2 := \mathrm{Cov}(\lambda_{A'_2}) = c(r^2 - \delta^2)\boldsymbol{P}_{T_2}$, where $c := 1/(d+2)$. Applying (72), we have

$$\boldsymbol{\Sigma}_T(\boldsymbol{s}) = \alpha' \boldsymbol{D}'_1 + (1-\alpha')\boldsymbol{D}'_2 + \alpha'(1-\alpha')(\boldsymbol{s} - \boldsymbol{s}_2)(\boldsymbol{s} - \boldsymbol{s}_2)^\top,$$

where

$$\alpha' := \frac{\mathrm{vol}(A'_1)}{\mathrm{vol}(A'_1) + \mathrm{vol}(A'_2)}.$$

By the triangle inequality, we have

$$\|\alpha \boldsymbol{D}_1 - \alpha' \boldsymbol{D}'_1\| \leq |\alpha' - \alpha| \|\boldsymbol{D}'_1\| + \alpha \|\boldsymbol{D}_1 - \boldsymbol{D}'_1\| \leq cr^2 |\alpha' - \alpha| + \|\boldsymbol{D}_1 - \boldsymbol{D}'_1\|,$$

and similarly,

$$\|(1-\alpha)\boldsymbol{D}_2 - (1-\alpha')\boldsymbol{D}'_2\| \leq cr^2 |\alpha' - \alpha| + (1-\alpha)\|\boldsymbol{D}_2 - \boldsymbol{D}'_2\|.$$

Assuming that $\kappa r \leq 1/C_{16}$, by Lemma 16, we have $\|\boldsymbol{D}_1 - \boldsymbol{D}'_1\| \vee \|\boldsymbol{D}_2 - \boldsymbol{D}'_2\| \leq C_{16}\kappa r^3$. Because $|\alpha'(1-\alpha') - \alpha(1-\alpha)| \leq |\alpha' - \alpha|$ and the identity $\|\boldsymbol{a}\boldsymbol{a}^\top - \boldsymbol{b}\boldsymbol{b}^\top\| \leq (\|\boldsymbol{a}\| + \|\boldsymbol{b}\|)\|\boldsymbol{a} - \boldsymbol{b}\|$, we also have

$$\begin{aligned}
&\|\alpha(1-\alpha)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top - \alpha'(1-\alpha')(\boldsymbol{s} - \boldsymbol{s}_2)(\boldsymbol{s} - \boldsymbol{s}_2)^\top\| \\
&\leq |\alpha' - \alpha| \|\boldsymbol{s} - \boldsymbol{s}_2\|^2 + \alpha(1-\alpha) 3r\big(\|\boldsymbol{\mu}_1 - \boldsymbol{s}\| + \|\boldsymbol{\mu}_2 - \boldsymbol{s}_2\|\big).
\end{aligned}$$

By Lemma 16, we have $\|\boldsymbol{\mu}_1 - \boldsymbol{s}\| \vee \|\boldsymbol{\mu}_2 - \boldsymbol{s}_2\| \leq C_{16}\kappa r^2$. Assuming that $\kappa r \leq 1/3$, $P_{T_k}^{-1}$ is well-defined and $(1+\kappa r)$-Lipschitz on $S_k \cap B_r$. And being an orthogonal projection, P_{T_k} is 1-Lipschitz . Hence, applying Lemma 6, we have

$$1 \leq \frac{\mathrm{vol}(A_k)}{\mathrm{vol}(P_{T_k}(A_k))} \leq 1 + \kappa r, \quad k = 1, 2.$$

Then by Lemma 9,

$$\frac{\mathrm{vol}(P_{T_k}(A_k) \triangle A'_k)}{\mathrm{vol}(A'_k)} \leq C_9\kappa(r + \kappa r^2) \leq 2C_9\kappa r, \quad k = 1, 2,$$

when κr is small enough, implying

$$1 - 2C_9\kappa r \leq \frac{\mathrm{vol}(P_{T_k}(A_k))}{\mathrm{vol}(A'_k)} \leq 1 + 2C_9\kappa r, \quad k = 1, 2.$$

Hence,

$$1 - C\kappa r \leq 1 - 2C_9\kappa r \leq \frac{\mathrm{vol}(A_k)}{\mathrm{vol}(A'_k)} \leq (1+\kappa r)(1 + 2C_9\kappa r) \leq 1 + C\kappa r, \quad k = 1, 2,$$

for some constant $C > 0$. Since for all $a, b, a', b' > 0$ we have

$$\left|\frac{a}{a+b} - \frac{a'}{a'+b'}\right| \leq \frac{|a - a'| \vee |b - b'|}{(a+b) \vee (a'+b')} \leq \big|1 - a/a'\big| \vee \big|1 - b/b'\big|,$$

we then get

$$|\alpha - \alpha'| \leq C\kappa r.$$

Applying the triangle inequality to $\|\boldsymbol{\Sigma}(\boldsymbol{s}) - \boldsymbol{\Sigma}_T(\boldsymbol{s})\|$ and using the bounds above, we conclude.

We assumed above that $\delta \leq r$. When $\delta > r$, $A_2 = A'_2 = \emptyset$ and $\alpha = \alpha' = 1$. In this case, $\boldsymbol{\mu}_2, \boldsymbol{D}_2, \mathrm{D}'_2$ are not well-defined, but one can easily check that the above calculations, and the resulting bound, remain valid. In fact, in this case $\boldsymbol{\Sigma}(\boldsymbol{s}) = \boldsymbol{D}_1$ and $\boldsymbol{\Sigma}_T(\boldsymbol{s}) = \boldsymbol{D}'_1$. □

Lemma 27. *Consider $S_1, S_2 \in \mathcal{S}_d(\kappa)$ intersecting at a positive angle, with $\mathrm{reach}(S_1 \cap S_2) \geq 1/\kappa$. There is a constant $C_{27} \geq 1$, depending on S_1 and S_2, such that the following holds. For $\boldsymbol{s} \in S_1$ with $\mathrm{dist}(\boldsymbol{s}, S_2) \leq 2r$, assuming $r \leq \frac{1}{C_{27}}$, let $\boldsymbol{\Sigma}(\boldsymbol{s})$ and $\boldsymbol{\Sigma}^0_T(\boldsymbol{s})$ denote the covariance matrices of the uniform distributions over $B(\boldsymbol{s}, r) \cap (S_1 \cup S_2)$ and $B(P_{T^0_1}(\boldsymbol{s}), r) \cap (T^0_1 \cup T^0_2)$, where $T^0_k := T_{S_k}(\boldsymbol{s}^0)$ and $\boldsymbol{s}^0 := P_{S_1 \cap S_2}(\boldsymbol{s})$. Then*

$$\|\boldsymbol{\Sigma}(\boldsymbol{s}) - \boldsymbol{\Sigma}^0_T(\boldsymbol{s})\| \leq C_{27}\, r^{5/2}.$$

Proof. We use the notation of Lemma 26 and its proof. In addition, we let $C > 0$ denote a constant that depends only on S_1 and S_2 whose value increases with each appearance.

Define $\boldsymbol{t}_1 = P_{T^0_1}(\boldsymbol{s})$ and $\boldsymbol{t}_2 = P_{T^0_2}(\boldsymbol{t}_1)$. Let $\delta_0 = \|\boldsymbol{t}_1 - \boldsymbol{t}_2\|$, $\delta_1 = \|\boldsymbol{s} - \boldsymbol{t}_1\|$, $\delta_2 = \|\boldsymbol{s}_2 - \boldsymbol{t}_2\|$. Define $A^0_1 = T^0_1 \cap B(\boldsymbol{t}_1, r)$ and $A^0_2 = T^0_2 \cap B(\boldsymbol{t}_1, r)$. Note that $A^0_2 = T^0_2 \cap B(\boldsymbol{t}_2, \sqrt{r^2 - \delta_0^2})$

when $\delta_0 \le r$, and $A_2^0 = \emptyset$ when $\delta_0 > r$. We have $\|\boldsymbol{s} - \boldsymbol{s}^0\| \le Cr$ by Lemma 21, which then implies $\delta_1 \le Cr^2$ by Lemma 2. Assuming that κr is small enough,

$$\begin{aligned}\delta_2 = \|\boldsymbol{s}_2 - \boldsymbol{t}_2\| &\le \|P_{S_2}(\boldsymbol{s}) - P_{T_2^0}(\boldsymbol{s})\| + \|P_{T_2^0}(\boldsymbol{s}) - P_{T_2^0}(\boldsymbol{t}_1)\| \\ &\le C_4\kappa\|\boldsymbol{s} - \boldsymbol{s}^0\|^2 + \|\boldsymbol{s} - \boldsymbol{t}_1\| \le Cr^2,\end{aligned}$$

where in the second line we used Lemma 4 and the fact that any metric projection is 1-Lipschitz. Hence,

$$|\delta_0 - \delta| \le \big|\|\boldsymbol{t}_1 - \boldsymbol{t}_2\| - \|\boldsymbol{s} - \boldsymbol{s}_2\|\big| \le \|\boldsymbol{t}_1 - \boldsymbol{s}\| + \|\boldsymbol{t}_2 - \boldsymbol{s}_2\| = \delta_1 + \delta_2 \le Cr^2.$$

Note that $\mathbb{E}(\lambda_{A_1^0}) = \boldsymbol{t}_1$ and, by Lemma 13 (with c denoting the constant defined there), $\boldsymbol{D}_1^0 := \mathrm{Cov}(\lambda_{A_1^0}) = cr^2\boldsymbol{P}_{T_1^0}$. If $\delta_0 \le r$, $\mathbb{E}(\lambda_{A_2^0}) = \boldsymbol{t}_2$ and $\boldsymbol{D}_2^0 := \mathrm{Cov}(\lambda_{A_2^0}) = c(r^2 - \delta_0^2)\boldsymbol{P}_{T_2^0}$; if $\delta_0 > r$, then $A_2^0 = \emptyset$, and we define $\boldsymbol{D}_2^0 = 0$ in that case. Applying (72), we have

$$\boldsymbol{\Sigma}_T^0(\boldsymbol{s}) = \alpha^0\boldsymbol{D}_1^0 + (1-\alpha^0)\boldsymbol{D}_2^0 + \alpha^0(1-\alpha^0)(\boldsymbol{t}_1 - \boldsymbol{t}_2)(\boldsymbol{t}_1 - \boldsymbol{t}_2)^\top,$$

where

$$\alpha^0 := \frac{\mathrm{vol}(A_1^0)}{\mathrm{vol}(A_1^0) + \mathrm{vol}(A_2^0)}.$$

Note that $\alpha^0 = 1$ when $\delta_0 > r$ (i.e., when $A_2^0 = \emptyset$). We focus on the case where $\max(\delta, \delta_0) \le r$, which is the most complex one. We have

$$\|\boldsymbol{D}_1' - \boldsymbol{D}_1^0\| = \|cr^2\boldsymbol{P}_{T_1} - cr^2\boldsymbol{P}_{T_1^0}\| = cr^2\|\boldsymbol{P}_{T_1} - \boldsymbol{P}_{T_1^0}\|,$$

and

$$\|\boldsymbol{D}_2' - \boldsymbol{D}_2^0\| = \|c(r^2 - \delta^2)\boldsymbol{P}_{T_2} - c(r^2 - \delta_0^2)_+\boldsymbol{P}_{T_2^0}\| \le cr^2\|\boldsymbol{P}_{T_2} - \boldsymbol{P}_{T_2^0}\| + c|\delta^2 - \delta_0^2|,$$

by the triangle inequality and the fact that $\|\boldsymbol{P}_T\| \le 1$ for any affine subspace T. By Lemma 3 and Lemma 18, we have

$$\|\boldsymbol{P}_{T_1} - \boldsymbol{P}_{T_1^0}\| \le 6\kappa\|\boldsymbol{s} - \boldsymbol{s}^0\| \le Cr, \qquad \|\boldsymbol{P}_{T_2} - \boldsymbol{P}_{T_2^0}\| \le 6\kappa\|\boldsymbol{s}_2 - \boldsymbol{s}^0\| \le Cr,$$

since

$$\|\boldsymbol{s}_2 - \boldsymbol{s}^0\| \le \|\boldsymbol{s}_2 - \boldsymbol{s}\| + \|\boldsymbol{s} - \boldsymbol{s}^0\| \le 2r + Cr.$$

And since $|\delta^2 - \delta_0^2| \le 2r|\delta - \delta_0| \le Cr^3$, we have $\|\boldsymbol{D}_k' - \boldsymbol{D}_k^0\| \le Cr^3$ for $k = 1, 2$. Let ω_d denote the volume of the d-dimensional unit ball. Then

$$\mathrm{vol}(A_1') = \omega_d r^d, \quad \mathrm{vol}(A_2') = \omega_d(r^2 - \delta^2)^{d/2}, \quad \mathrm{vol}(A_1^0) = \omega_d r^2, \quad \mathrm{vol}(A_2^0) = \omega_d(r^2 - {\delta_0}^2)^{d/2},$$

so that

$$\begin{aligned}|\alpha' - \alpha^0| &= \left|\frac{1}{1 + (1-\delta/r)^{d/2}} - \frac{1}{1 + (1-\delta_0/r)^{d/2}}\right| \\ &\le \big|(1-\delta/r)^{d/2} - (1-\delta_0/r)^{d/2}\big|.\end{aligned}$$

Proceeding as when we bounded ζ in the proof of Lemma 23, we get

$$|\alpha' - \alpha^0| \le d\sqrt{|\delta - \delta_0|/r} \le C\sqrt{r}.$$

Hence, we proved that

$$\|\boldsymbol{\Sigma}_T(\boldsymbol{s}) - \boldsymbol{\Sigma}_T^0(\boldsymbol{s})\| \le Cr^{5/2}.$$

We use this inequality, (83), and the triangle inequality, to get

$$\|\boldsymbol{\Sigma}(\boldsymbol{s}) - \boldsymbol{\Sigma}_T^0(\boldsymbol{s})\| \le \|\boldsymbol{\Sigma}(\boldsymbol{s}) - \boldsymbol{\Sigma}_T(\boldsymbol{s})\| + \|\boldsymbol{\Sigma}_T(\boldsymbol{s}) - \boldsymbol{\Sigma}_T^0(\boldsymbol{s})\| \le Cr^{5/2},$$

which is what we needed to prove. □

6.2 Performance Guarantees for Algorithm 2

We deal with the case where there is no noise, that is, $\tau = 0$ in (7), so that the data points are $\boldsymbol{s}_1, \dots, \boldsymbol{s}_N$ and sampled exactly on $S_1 \cup S_2$ according to the uniform distribution. We explain how things change when there is noise, meaning $\tau > 0$, in Section 6.4.

We start with some notation. Let $\Xi_i = \{j \ne i : \boldsymbol{s}_j \in N_r(\boldsymbol{s}_i)\}$, with (random) cardinality $N_i = |\Xi_i|$. For $i \in [n]$, let $K_i = 1$ if $\boldsymbol{s}_i \in S_1$ and $= 2$ otherwise, and let $T_i = T_{S_{K_i}}(\boldsymbol{s}_i)$, which is the tangent subspace associated with data point $\boldsymbol{s}_i$. Let $\boldsymbol{P}_i$ be short for $\boldsymbol{P}_{T_i}$. Let

$$I_\star = \{i : K_j = K_i,\ \forall j \in \Xi_i\},$$

or equivalently,

$$I_\star^c = \{i : \exists j \text{ s.t. } K_j \ne K_i \text{ and } \|\boldsymbol{s}_j - \boldsymbol{s}_i\| \le r\}.$$

Thus $I_\star$ indexes the points whose neighborhoods do not contain points from the other cluster.

We will soon define a constant $C_\bullet$ and will require the following

$$C_\bullet \ge C, \quad r \le \varepsilon/C, \quad \varepsilon \le \eta/C, \quad \eta \le 1/C, \tag{90}$$

for a large enough constant $C \ge 1$ that depends only on S_1 and S_2.

6.2.1 A Concentration Bound for Local Covariances

Objective. *We derive a concentration bound for the local covariances.*

Let $\Xi_i = \{j \ne i : \boldsymbol{s}_j \in N_r(\boldsymbol{s}_i)\}$, with (random) cardinality $N_i = |\Xi_i|$. When there is no noise, $\boldsymbol{C}_i$ is the sample covariance of $\{\boldsymbol{s}_j : j \in \Xi_i\}$. Also, let $\boldsymbol{\Sigma}(\boldsymbol{s})$ denote the covariance matrix of the uniform distribution over $B(\boldsymbol{s}, r) \cap (S_1 \cup S_2)$, and note that $\boldsymbol{\Sigma}_i := \boldsymbol{\Sigma}(\boldsymbol{s}_i) = \mathbb{E}(\boldsymbol{C}_i|\boldsymbol{s}_i)$. Given N_i, $\{\boldsymbol{s}_j : j \in \Xi_i\}$ are uniformly distributed on $B(\boldsymbol{s}_i, r) \cap (S_1 \cup S_2)$, and applying Lemma 22, we get that for any $t > 0$

$$\mathbb{P}\left(\|\boldsymbol{C}_i - \boldsymbol{\Sigma}_i\| > r^2t + 3\kappa r^3 \,\middle|\, \boldsymbol{s}_i, N_i\right) \le C_{22}\exp\left(-\frac{N_i t}{C_{22}}\min\left(t, N_i\right)\right). \tag{91}$$

Assume that $r < 1/(C_{11}\kappa)$ and let $n_\ddagger := nr^d/(2C_{11}/\chi)$ where $\chi := \frac{\mathrm{vol}_d(S_1) \wedge \mathrm{vol}_d(S_2)}{\mathrm{vol}_d(S_1) + \mathrm{vol}_d(S_2)}$. Define

$$\Omega_1 = \{\forall i : N_i > n_\ddagger\}.$$

Note that, for any i,

$$N_i \geq N_i' := \#\{j \neq i : K_i = K_j \text{ and } \boldsymbol{s}_j \in B(\boldsymbol{s}_i, r)\},$$

meaning that N_i' counts the number of neighboring points from the same surface. We have that $\#\{i : K_i = k\} \sim \mathrm{Bin}(n, \chi_k)$ where $\chi_k := \frac{\mathrm{vol}_d(S_k)}{\mathrm{vol}_d(S_1)+\mathrm{vol}_d(S_2)}$, and using a standard concentration inequality for the binomial distribution, there is a numeric constant $C > 0$ such that

$$\mathbb{P}(\#\{i : K_i = k\} \geq \chi n/2) \geq 1 - \exp(-\chi n/C).$$

(We also used the fact that $\chi = \chi_1 \wedge \chi_2$.) With this, we may apply Lemma 11, to get

$$\begin{aligned}\mathbb{P}(\Omega_1^c) &\leq 2\exp(-\chi n/C) + C_{11} r^{-d} \exp(-(\chi n/2) r^d / C_{11}) \\ &\leq Cn \exp(-nr^d/C),\end{aligned}$$

for some other constant $C > 0$. (We used the fact that the left-hand side can be made greater than 1 when $nr^d \leq 1$ by choosing C large enough, so that we may focus on the case where $nr^d \geq 1$.) Define the event

$$\Omega_2 = \Big\{\forall i : \|\boldsymbol{C}_i - \boldsymbol{\Sigma}_i\| \leq r^2\eta/C_\bullet + 3\kappa r^3\Big\},$$

where $C_\bullet$ will be chosen large enough, but fixed, later on. Assume that $n_\ddagger \geq \eta/C_\bullet$. Note that $\eta \leq 1$ by assumption, and we may assume $n_\ddagger$ to be larger than any given constant, for if not, the statement in Theorem 1 is void. Based on the union bound and (91), we have

$$\begin{aligned}\mathbb{P}(\Omega_2^c) &\leq \mathbb{P}(\Omega_2^c|\Omega_1) + \mathbb{P}(\Omega_1^c) \\ &\leq nC_{22}\exp(-n_\ddagger(\eta/C_\bullet)^2/C_{22}) + Cn\exp(-nr^d/C) \\ &\leq Cn\exp(-nr^d\eta^2/C),\end{aligned}$$

for another constant C that depends on $C_\bullet$ and d.

6.2.2 Away from the Intersection

Objective. *We show that there is an appropriate choice of parameters as in Theorem 1 such that, away from the intersection—when confined to $I_\star$—two points belonging to the same surface and within distance ε are neighbors, while two points belonging to different surfaces are not neighbors.*

For $i \in [n]$, let $K_i = 1$ if $\boldsymbol{s}_i \in S_1$ and $= 2$ otherwise. Let $T_i = T_{S_{K_i}}(\boldsymbol{s}_i)$, which is the tangent subspace associated with data point $\boldsymbol{s}_i$. Let

$$I_\star = \{i : K_j = K_i,\ \forall j \in \Xi_i\},$$

or equivalently,

$$I_\star^c = \{i : \exists j \text{ s.t. } K_j \neq K_i \text{ and } \|\boldsymbol{s}_j - \boldsymbol{s}_i\| \leq r\}.$$

By definition, $I_\star$ indexes the points whose neighborhoods do not contain points from the other cluster.

For $i \in I_\star$, let $\boldsymbol{\Sigma}_{T,i}$ denote the covariance of the uniform distribution on $T_i \cap B(\boldsymbol{s}_i, r)$. Applying Lemma 16, this leads to

$$\|\boldsymbol{\Sigma}_i - \boldsymbol{\Sigma}_{T,i}\| \leq C_{16}\kappa r^3, \quad \forall i \in I_\star. \tag{95}$$

Under Ω_2, by the triangle inequality and (95), we have

$$\|\boldsymbol{C}_i - \boldsymbol{\Sigma}_{T,i}\| \leq \|\boldsymbol{C}_i - \boldsymbol{\Sigma}_i\| + \|\boldsymbol{\Sigma}_i - \boldsymbol{\Sigma}_{T,i}\| \leq \zeta r^2, \quad \forall i \in I_\star, \tag{96}$$

where

$$\zeta := \eta/C_\bullet + (3 + C_{16})\kappa r. \tag{97}$$

The inequality (96) leads, via the triangle inequality, to the decisive bound

$$\|\boldsymbol{C}_i - \boldsymbol{C}_j\| \leq \|\boldsymbol{\Sigma}_{T,i} - \boldsymbol{\Sigma}_{T,j}\| + 2\zeta r^2, \qquad \forall i, j \in I_\star. \tag{98}$$

Take $i, j \in I_\star$ such that $K_i = K_j$ and $\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \leq \varepsilon$. Then by Lemma 13, Lemma 18 and Lemma 3, and the triangle inequality, we have

$$\frac{1}{cr^2}\|\boldsymbol{\Sigma}_{T,i} - \boldsymbol{\Sigma}_{T,j}\| = \sin\theta_{\max}(T_i, T_j) \leq 6\kappa\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \leq 6\kappa\varepsilon, \tag{99}$$

where $c := 1/(d+2)$. This implies that

$$\frac{1}{r^2}\|\boldsymbol{C}_i - \boldsymbol{C}_j\| \leq 6c\kappa\varepsilon + 2\zeta,$$

by the triangle inequality and (98). Therefore, if $\eta > 6c\kappa\varepsilon + 2\zeta$, then any pair of points indexed by $i, j \in I_\star$ from the same cluster and within distance ε are direct neighbors in the graph built by Algorithm 2.

Take $i, j \in I_\star$ such that $K_i \neq K_j$ and $\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \leq \varepsilon$. By Lemma 21,

$$\max\big[\operatorname{dist}(\boldsymbol{s}_i, S_1 \cap S_2), \operatorname{dist}(\boldsymbol{s}_j, S_1 \cap S_2)\big] \leq C_{21}\|\boldsymbol{s}_i - \boldsymbol{s}_j\|.$$

Let $\boldsymbol{z}$ be the mid-point of $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$. By convexity and the display above,

$$\operatorname{dist}(\boldsymbol{z}, S_1 \cap S_2) \leq \frac{1}{2}\operatorname{dist}(\boldsymbol{s}_i, S_1 \cap S_2) + \frac{1}{2}\operatorname{dist}(\boldsymbol{s}_j, S_1 \cap S_2) \leq C_{21}\varepsilon.$$

Assuming $C_{21}\varepsilon < 1/\kappa$, let $\boldsymbol{s} = P_{S_1 \cap S_2}(\boldsymbol{z})$. Then, by the triangle inequality again,

$$\max\big[\|\boldsymbol{s} - \boldsymbol{s}_i\|, \|\boldsymbol{s} - \boldsymbol{s}_j\|\big] \leq \operatorname{dist}(\boldsymbol{z}, S_1 \cap S_2) + \frac{1}{2}\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \leq C_{21}\varepsilon + \frac{1}{2}\varepsilon.$$

Let T_i' denote the tangent subspace of S_{K_i} at $\boldsymbol{s}$ and let $\boldsymbol{\Sigma}_i'$ be the covariance of the uniform distribution over $T_i' \cap B(\boldsymbol{s}, r)$. Define T_j' and $\boldsymbol{\Sigma}_j'$ similarly. Then, as in (99) we have

$$\frac{1}{cr^2}\|\boldsymbol{\Sigma}_{T,i} - \boldsymbol{\Sigma}_i'\| \leq 6\kappa\|\boldsymbol{s}_i - \boldsymbol{s}\| \leq 6\kappa(C_{21} + 1/2)\varepsilon,$$

and similarly,

$$\frac{1}{cr^2}\|\boldsymbol{\Sigma}_{T,j} - \boldsymbol{\Sigma}_j'\| \leq 6\kappa(C_{21} + 1/2)\varepsilon.$$

Moreover, by Lemma 13 and Lemma 18,

$$\frac{1}{cr^2}\|\mathbf{\Sigma}'_i - \mathbf{\Sigma}'_j\| = \sin\theta_{\max}(T'_i, T'_j) \geq \sin\theta_S,$$

where θ_S is short for $\theta(S_1, S_2)$, defined in (6). Hence, by the triangle inequality,

$$\frac{1}{cr^2}\|\mathbf{\Sigma}_{T,i} - \mathbf{\Sigma}_{T,j}\| \geq \sin\theta_S - 12\kappa(C_{21} + 1/2)\varepsilon,$$

and then

$$\frac{1}{r^2}\|\boldsymbol{C}_i - \boldsymbol{C}_j\| \geq c\sin\theta_S - 12c\kappa(C_{21} + 1/2)\varepsilon - 2\zeta, \tag{100}$$

by the triangle inequality and (98). Therefore, if $\eta < c\sin\theta_S - 12c\kappa(C_{21} + 1/2)\varepsilon - 2\zeta$, then any pair of points indexed by $i, j \in I_\star$ from different clusters are *not* direct neighbors in the graph built by Algorithm 2.

In summary, we would like to choose η such that

$$c\kappa\varepsilon + 2\zeta < \eta < c\sin\theta_S - 12c\kappa(C_{21} + 1/2)\varepsilon - 2\zeta.$$

Recalling the definition of ζ in (97), this holds under (90).

6.2.3 The Different Clusters are Disconnected in the Graph

Objective. *We show that Step 2 in Algorithm 2 eliminates all points $i \notin I_\star$, implying by our conclusion in Section 6.2.2 that the two clusters are not connected in the graph.*

Hence, take $i \notin I_\star$ with $K_i = 1$ (say), so that $\mathrm{dist}(\boldsymbol{s}_i, S_2) \leq r$. By Lemma 21, we have $\mathrm{dist}(\boldsymbol{s}_i, S_1 \cap S_2) \leq C_{21}r$. Assume r is small enough that $C_{21}\kappa r < 1/2$. Let $\boldsymbol{s}^0 = P_{S_1\cap S_2}(\boldsymbol{s}_i)$ and define $T_k^0 = T_{S_k}(\boldsymbol{s}^0)$ for $k \in \{1, 2\}$. For $\boldsymbol{s} \in S_1$ such that $\mathrm{dist}(\boldsymbol{s}, S_2) \leq 2r$, define $\mathbf{\Sigma}_T^0(\boldsymbol{s})$ as in Lemma 27.

Assuming that $\boldsymbol{s}_i \neq \boldsymbol{s}^0$ (which is true with probability one) and $\boldsymbol{s}^0 = 0$ (by translation invariance), let $h = \|\boldsymbol{s}_i - \boldsymbol{s}^0\|$ and $\boldsymbol{v} = (\boldsymbol{s}_i - \boldsymbol{s}^0)/h$. Note that $\boldsymbol{s}_i = h\boldsymbol{v}$. Because $\boldsymbol{v} \perp T_1^0 \cap T_2^0$ by Lemma 1, and that $\theta_{\min}(T_1^0, T_2^0) \geq \theta_S$, we apply Lemma 24 to find $\ell \in h \pm r/2$ such that $\|\mathbf{\Sigma}_T^0(\ell\boldsymbol{v}) - \mathbf{\Sigma}_T^0(h\boldsymbol{v})\| \geq r^2/C_{24}$, where $C_{24} \geq 1$ depends only on θ_S and d. Letting $\tilde{\boldsymbol{s}} = \ell\boldsymbol{v}$, we have $\|\tilde{\boldsymbol{s}} - \boldsymbol{s}_i\| = |h - \ell| \leq r/2$, so that

$$\mathrm{dist}(\tilde{\boldsymbol{s}}, S_2) \leq \|\tilde{\boldsymbol{s}} - \boldsymbol{s}_i\| + \mathrm{dist}(\boldsymbol{s}_i, S_2) \leq r/2 + r \leq 3r/2.$$

By Lemma 21, we have $\mathrm{dist}(\tilde{\boldsymbol{s}}, S_1 \cap S_2) \leq C_{21}2r < 1/\kappa$, and consequently, $P_{S_1\cap S_2}(\tilde{\boldsymbol{s}}) = \boldsymbol{s}^0$, by Lemma 1. Hence, by the triangle inequality and Lemma 27,

$$\begin{aligned}\|\mathbf{\Sigma}(\boldsymbol{s}_i) - \mathbf{\Sigma}(\tilde{\boldsymbol{s}})\| &\geq \|\mathbf{\Sigma}_T^0(\boldsymbol{s}) - \mathbf{\Sigma}_T^0(\tilde{\boldsymbol{s}})\| - \|\mathbf{\Sigma}(\boldsymbol{s}_i) - \mathbf{\Sigma}_T^0(\boldsymbol{s}_i)\| - \|\mathbf{\Sigma}(\tilde{\boldsymbol{s}}) - \mathbf{\Sigma}_T^0(\tilde{\boldsymbol{s}})\| \\ &\geq r^2/C_{24} - 2C_{27}r^{5/2}.\end{aligned}$$

Take any $\bar{\boldsymbol{s}} \in S_1$ such that $\|\bar{\boldsymbol{s}} - \tilde{\boldsymbol{s}}\| \leq r/2$. By Lemma 23,

$$\|\mathbf{\Sigma}_T^0(\bar{\boldsymbol{s}}) - \mathbf{\Sigma}_T^0(\tilde{\boldsymbol{s}})\| \leq 5dr\|\bar{\boldsymbol{s}} - \tilde{\boldsymbol{s}}\| + 2\sqrt{r^3\|\bar{\boldsymbol{s}} - \tilde{\boldsymbol{s}}\|} \leq (5d + 2)r^{3/2}\sqrt{\|\bar{\boldsymbol{s}} - \tilde{\boldsymbol{s}}\|}.$$

Noting that

$$\mathrm{dist}(\bar{\boldsymbol{s}}, S_2) \leq \|\bar{\boldsymbol{s}} - \tilde{\boldsymbol{s}}\| + \mathrm{dist}(\tilde{\boldsymbol{s}}, S_2) \leq r/2 + 3r/2 \leq 2r,$$

we apply the triangle inequality and Lemma 27, and we get

$$\begin{aligned}\|\mathbf{\Sigma}(\bar{\boldsymbol{s}}) - \mathbf{\Sigma}(\tilde{\boldsymbol{s}})\| &\leq \|\mathbf{\Sigma}_T^0(\bar{\boldsymbol{s}}) - \mathbf{\Sigma}_T^0(\tilde{\boldsymbol{s}})\| + \|\mathbf{\Sigma}(\bar{\boldsymbol{s}}) - \mathbf{\Sigma}_T^0(\bar{\boldsymbol{s}})\| + \|\mathbf{\Sigma}(\tilde{\boldsymbol{s}}) - \mathbf{\Sigma}_T^0(\tilde{\boldsymbol{s}})\| \\ &\leq (5d + 2)r^{3/2}\sqrt{\|\bar{\boldsymbol{s}} - \tilde{\boldsymbol{s}}\|} + 2C_{27}r^{5/2}.\end{aligned}$$

When $\|\bar{\boldsymbol{s}} - \tilde{\boldsymbol{s}}\| \leq r/C_\ddagger$, we have

$$\begin{aligned}\|\mathbf{\Sigma}(\bar{\boldsymbol{s}}) - \mathbf{\Sigma}(\boldsymbol{s}_i)\| &\geq \|\mathbf{\Sigma}(\boldsymbol{s}_i) - \mathbf{\Sigma}(\tilde{\boldsymbol{s}})\| - \|\mathbf{\Sigma}(\bar{\boldsymbol{s}}) - \mathbf{\Sigma}(\tilde{\boldsymbol{s}})\| \\ &\geq r^2\big(1/C_{24} - (5d + 2)/\sqrt{C_\ddagger} - 4C_{27}r^{1/2}\big) \geq r^2/(2C_{24}),\end{aligned}$$

assuming $C_\ddagger$ is large enough and r is small enough. Fixing such a constant $C_\ddagger \geq 2$, define the event

$$\Omega_3 = \bigcap_{k=1}^{2}\Big\{\forall \boldsymbol{s} \in S_k : \#\{i : K_i = k \text{ and } \boldsymbol{s}_i \in B(\boldsymbol{s}, r/C_\ddagger)\} \geq 2\Big\}.$$

By the same arguments bounding $\mathbb{P}(\Omega_1^c)$ in Section 6.2.1, we have that there is a constant C such that, when $r \leq 1/(C\kappa)$,

$$\mathbb{P}(\Omega_3^c) \leq Cn\exp(-nr^d/C).$$

Under Ω_3, there is $\boldsymbol{s}_j \in S_1 \cap B(\tilde{\boldsymbol{s}}, r/C_\ddagger)$. Taking $\bar{\boldsymbol{s}} = \boldsymbol{s}_j$ above, we have that $\|\mathbf{\Sigma}_j - \mathbf{\Sigma}_i\| \geq r^2/(2C_{24})$. Under Ω_2, by the triangle inequality,

$$\begin{aligned}\|\boldsymbol{C}_j - \boldsymbol{C}_i\| &\geq \|\mathbf{\Sigma}_j - \mathbf{\Sigma}_i\| - \|\boldsymbol{C}_i - \mathbf{\Sigma}_i\| - \|\boldsymbol{C}_j - \mathbf{\Sigma}_j\| \\ &\geq r^2/(2C_{24}) - 2r^2\eta/C_\bullet \geq r^2/(3C_{24}),\end{aligned}$$

choosing $C_\bullet$ sufficiently large. (Recall that $\eta \leq 1$ by assumption.) Therefore, choosing η such that $\eta < 1/(3C_{24})$, we see that $\|\boldsymbol{C}_j - \boldsymbol{C}_i\| > \eta r^2$, even though

$$\|\boldsymbol{s}_j - \boldsymbol{s}_i\| \leq \|\boldsymbol{s}_j - \tilde{\boldsymbol{s}}\| + \|\tilde{\boldsymbol{s}} - \boldsymbol{s}_i\| \leq r/C_\ddagger + r/2 \leq r.$$

6.2.4 The Points that are Removed are close to the Intersection

Objective. *We show that the points removed by Step 2 are within distance Cr of the intersection.*

Let $I_\circ = \{i \in I_\star : \Xi_i \cap I_\star\}$. By our choice of parameters in (90), we see that $i \in I_\circ$ is neighbor with any $j \in \Xi_i$, so that i survives Step 2. Hence, the nodes removed at Step 2 are in $I_\circ^c = \{i \in I_\star : \Xi_i \cap I_\star^c \neq \emptyset\} \cup I_\star^c$, with the possibility that some nodes in $I_\circ^c$ survive. Now, for any $i \in I_\star^c$, there is j with $K_j \neq K_i$ such that $\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \leq r$, so by Lemma 21,

$$\mathrm{dist}(\boldsymbol{s}_i, S_1 \cap S_2) \leq C_{21}\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \leq C_{21}r.$$

And for any $i \in I_\circ \setminus I_\star^c$, there is $j \in I_\star^c$ such that $\|\boldsymbol{s}_j - \boldsymbol{s}_i\| \leq r$, by definition of Ξ_i, so that $\mathrm{dist}(\boldsymbol{s}_i, S_1 \cap S_2) \leq C_{21}r + r$, by the triangle inequality. In fact, we just proved that the last inequality holds for all $i \in I_\circ^c$. Hence, the points removed in Step 2 are within distance $(C_{21} + 1)r$ of $S_1 \cap S_2$.

6.2.5 Each Cluster is (essentially) Connected in the Graph

Objective. *We show that the points that survive Step 2 and belong to the same cluster are connected in the graph, except for possible spurious points within distance Cr of the intersection.*

Take, for example, the cluster generated from sampling S_1. Before we start, we recall that $I_\star^c$ was eliminated in Step 2, so that by our choice of η in (90), to show that two points $\boldsymbol{s}_i, \boldsymbol{s}_j$ sampled from S_1 are neighbors it suffices to show that $\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \le \varepsilon$.

Define $R = S_1 \setminus B(S_1 \cap S_2, r')$, where $r' := (C_{21}+2)r$. By our conclusion in Section 6.2.4, we have that

$$\text{any } \boldsymbol{s}_i \in S_1 \text{ such that } \operatorname{dist}(\boldsymbol{s}_i, R) < r \text{ survives Step 2,} \tag{109}$$

so that it suffices to show that the subset of nodes $\{i : \boldsymbol{s}_i \in R\}$ is connected.

Define $W = S_1 \setminus S_2$ and let $\{W_m\}$ denote the connected components of W. (Note that there might only be one, for example, when S_1 and S_2 are circles in dimension $D = 3$, not in the same plane, that intersect at a single point.) Define $R_m = W_m \setminus B(S_1 \cap S_2, r')$. Since $W = \sqcup_m W_m = S_1 \setminus (S_1 \cap S_2)$, we have $R = \sqcup_m R_m$. In fact, when r (and therefore r') is small enough, R_m is connected, and assuming this is the case, $\{R_m\}$ are the connected components of R.

Suppose R_1 and R_2 are two adjacent connected components of R, meaning that $R_1 \cup R_2 \cup S_2$ is connected. We show that there is at least one pair j_1, j_2 of direct neighbors in the graph such that $\boldsymbol{s}_{j_m} \in R_m$. Take $\boldsymbol{s}$ on the connected component of $S_1 \cap S_2$ separating R_1 and R_2. We construct two points, $\boldsymbol{s}^1 \in R_1$ and $\boldsymbol{s}^2 \in R_2$, that are within distance of order ε from $\boldsymbol{s}$, and then select $\boldsymbol{s}_{j_m}$ as the closest data point to $\boldsymbol{s}^m$. Let $T^k = T_{S_k}(\boldsymbol{s})$ and let H be the affine subspace parallel to $(T^1 \cap T^2)^\perp$ passing through $\boldsymbol{s}$. Take $\boldsymbol{t}^m \in P_{T^1}(R_m) \cap H \cap \partial B(\boldsymbol{s}, \varepsilon_1)$, where $\varepsilon_1 := \varepsilon/4$, which exists when ε is sufficiently small. Note that $\boldsymbol{t}^1, \boldsymbol{t}^2 \in T^1$ and assuming that ε is sufficiently small that $\varepsilon_1 < 1/\kappa$, we have $P_{S_1 \cap S_2}(\boldsymbol{t}^m) = \boldsymbol{s}$ by Lemma 1. Define $\boldsymbol{s}^m = P_{T^1}^{-1}(\boldsymbol{t}^m)$ and note that $\boldsymbol{s}^1, \boldsymbol{s}^2 \in S_1$. Lemma 5 not only justifies this construction when $\kappa\varepsilon_1 < 1/C_5$, it also says that $P_{T^1}^{-1}$ has Lipschitz constant bounded by $1 + \kappa\varepsilon_1$, which implies that

$$\|\boldsymbol{s}^m - \boldsymbol{s}\| \le (1 + \kappa\varepsilon_1)\|\boldsymbol{t}^m - \boldsymbol{s}\| = (1 + \kappa\varepsilon_1)\varepsilon_1 \le \varepsilon/3.$$

We also have

$$\begin{aligned}
\operatorname{dist}(\boldsymbol{s}^m, S_1 \cap S_2) &\ge \operatorname{dist}(\boldsymbol{t}^m, S_1 \cap S_2) - \|\boldsymbol{s}^m - \boldsymbol{t}^m\| \\
&= \|\boldsymbol{t}^m - \boldsymbol{s}\| - \|\boldsymbol{s}^m - \boldsymbol{t}^m\| \\
&\ge \varepsilon_1 - \frac{\kappa}{2}\|\boldsymbol{s}^m - \boldsymbol{s}\|^2 \\
&\ge \big(1 - \frac{\kappa}{2}(1 + \kappa\varepsilon_1)^2\varepsilon_1\big)\varepsilon_1 \\
&\ge \varepsilon/5,
\end{aligned}$$

when ε is sufficiently small. We used (10) in the second inequality. We assume r/ε is sufficiently small that $\varepsilon/5 \ge r' + r$. Then under Ω_3, there are j_1, j_2 such that $\boldsymbol{s}_{j_m} \in B(\boldsymbol{s}^m, r) \cap S_1$, and by the triangle inequality, we then have that $\operatorname{dist}(\boldsymbol{s}_{j_m}, S_1 \cap S_2) \ge \varepsilon/5 - r \ge r'$, so

that $\boldsymbol{s}_{j_m} \in R_m$. Moreover,

$$\begin{aligned}
\|\boldsymbol{s}_{j_1} - \boldsymbol{s}_{j_2}\| &\le \|\boldsymbol{s}_{j_1} - \boldsymbol{s}^1\| + \|\boldsymbol{s}^1 - \boldsymbol{s}\| + \|\boldsymbol{s} - \boldsymbol{s}^2\| + \|\boldsymbol{s}^2 - \boldsymbol{s}_{j_2}\| \\
&\le r + \varepsilon/3 + \varepsilon/3 + r \\
&= \frac{2}{3}\varepsilon + 2r \le \varepsilon,
\end{aligned}$$

assuming r/ε is sufficiently small.

Now, we show that the points sampled from any connected component, say R_1, form a connected subgraph. Take $\boldsymbol{s}^1, \dots, \boldsymbol{s}^M$ an r-packing of R_1, so that

$$\bigsqcup_m (R_1 \cap B(\boldsymbol{s}^m, r/2)) \subset R_1 \subset \bigcup_m (R_1 \cap B(\boldsymbol{s}^m, r)).$$

Because R_1 is connected, $\cup_m B(\boldsymbol{s}^m, r)$ is necessarily connected. Under Ω_1, and $C_{26} \ge 2$, all the balls $B(\boldsymbol{s}^m, r), m = 1, \dots, M$, contain at least one $\boldsymbol{s}_i \in S_1$, and any such point survives Step 2, because of $\operatorname{dist}(\boldsymbol{s}_i, R_1) < r$ and (109). Assume $r/\varepsilon \le 1/4$. Then two points $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ such that $\boldsymbol{s}_i, \boldsymbol{s}_j \in B(\boldsymbol{s}^m, r)$ are connected, since $\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \le 2r \le \varepsilon$. And when $B(\boldsymbol{s}^{m_1}, r) \cap B(\boldsymbol{s}^{m_2}, r) \ne \emptyset$, $\boldsymbol{s}_i \in B(\boldsymbol{s}^{m_1}, r)$ and $\boldsymbol{s}_j \in B(\boldsymbol{s}^{m_2}, r)$ are such that $\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \le 4r \le \varepsilon$. Hence, the points sampled from R_1 are connected in the graph under Ω_1.

6.2.6 Summary and Conclusion

We worked under the assumption that (90), with a constant C large enough and depending only on S_1 and S_2. This is compatible with the statement of Theorem 1. In Section 6.2.1, we derived a concentration inequality for local covariances. This lead to a uniform bound (defining the event Ω_2) which holds with probability at least $1 - Cn\exp(-nr^d\eta^2/C)$ for some constant $C > 0$. In Section 6.2.2, we showed that among points away from the intersection (indexed by $I_\star$), those that are in the same cluster are neighbors in the graph if they are within distance ε, while those in different clusters cannot be neighbors in the graph. In Section 6.2.3, we showed that Step 2 removes points not indexed by $I_\star$, thus implying that the two clusters are not connected in the graph. In the process, we assumed that an event (denoted Ω_3) held, which happens with probability at least $1 - Cn\exp(-nr^d/C)$ for some constant $C > 0$. In Section 6.2.4, we showed that the points removed in Step 2 are within distance $O(r)$ from the intersection of the surfaces. In Section 6.2.5, we showed that, except for points within distance $O(r)$ of the intersection, two points in the same cluster are connected in the graph.

We conclude that, after Step 4, Algorithm 2 returns two groups, and each group covers an entire cluster except for points within distance $O(r)$ of the intersection. In Step 5, each point removed in Step 2 is included in the group which contains its closest point (in Euclidean distance). Theorem 1 is silent on the accuracy of doing this step. So the proof of Theorem 1, as it relates to Algorithm 2, is complete.

6.3 Performance Guarantees for Algorithm 3

We keep the same notation as in Section 6.2 and go a little faster here as the arguments are parallel. Let d_i denote the estimated dimensionality at point $\boldsymbol{s}_i$, meaning the number

of eigenvalues of $\boldsymbol{C}_i$ exceeding $\sqrt{\eta}\,\|\boldsymbol{C}_i\|$. Recall that $\boldsymbol{Q}_i$ denotes the orthogonal projection onto the top d_i eigenvectors of $\boldsymbol{C}_i$.

6.3.1 Estimation of the Intrinsic Dimension

Objective. *We show that, for $i \in I_\star$, $d_i = d$ (the correct intrinsic dimension) with high probability.*

Take $i \in I_\star$. Under Ω_2, (96) holds, and applying Weyl's inequality (Stewart and Sun, 1990, Cor. IV.4.9), we have

$$|\beta_m(\boldsymbol{C}_i) - \beta_m(\boldsymbol{\Sigma}_{T,i})| \le \zeta r^2, \quad \forall m = 1, \dots, D.$$

By Lemma 13, $\boldsymbol{\Sigma}_{T,i} = cr^2\boldsymbol{P}_i$, so that $\beta_m(\boldsymbol{\Sigma}_{T,i}) = cr^2$ when $m \le d$ and $\beta_m(\boldsymbol{\Sigma}_{T,i}) = 0$ when $m > d$. Hence,

$$\beta_1(\boldsymbol{C}_i) \le (c+\zeta)r^2, \qquad \beta_d(\boldsymbol{C}_i) \ge (c-\zeta)r^2, \qquad \beta_{d+1}(\boldsymbol{C}_i) \le \zeta r^2.$$

This implies that

$$\frac{\beta_d(\boldsymbol{C}_i)}{\beta_1(\boldsymbol{C}_i)} \ge \frac{c-\zeta}{c+\zeta} > \sqrt{\eta}, \qquad \frac{\beta_{d+1}(\boldsymbol{C}_i)}{\beta_1(\boldsymbol{C}_i)} \le \frac{\zeta}{c+\zeta} < \sqrt{\eta},$$

when $\zeta \le \eta/2$ and η is sufficiently small, which is the case under (90). When this is so, $d_i = d$ by definition of d_i. (Note that $\|\boldsymbol{C}_i\| = \beta_1(\boldsymbol{C}_i)$.)

6.3.2 Each Cluster is (essentially) Connected in the Graph

Objective. *We show that each cluster is connected in the graph, except possibly for some points near the intersection.*

Note that the top d eigenvectors of $\boldsymbol{\Sigma}_{T,i}$ generate T_i. Hence, applying Lemma 19, and (96) again, we get that (recall that $c = 1/(d+1)$)

$$\|\boldsymbol{Q}_i - \boldsymbol{P}_i\| \le \frac{\sqrt{2}\,\zeta r^2}{cr^2} = \zeta' := \sqrt{2}(d+2)\zeta, \quad \forall i \in I_\star.$$

This is the equivalent of (96), which leads to the equivalent of (98):

$$\|\boldsymbol{Q}_i - \boldsymbol{Q}_j\| \le \frac{1}{cr^2}\|\boldsymbol{\Sigma}_{T,i} - \boldsymbol{\Sigma}_{T,j}\| + 2\zeta', \qquad \forall i, j \in I_\star,$$

using the fact that $\boldsymbol{\Sigma}_{T,i} = cr^2\boldsymbol{P}_i$. When $i, j \in I_\star$ are such that $K_i = K_j$, based on (99), we have

$$\|\boldsymbol{Q}_i - \boldsymbol{Q}_j\| \le 6\kappa\varepsilon + 2\zeta'.$$

Hence, when $\eta > 6\kappa\varepsilon + 2\zeta'$ (which is the case under (90)), two nodes $i, j \in I_\star$ such that $K_i = K_j$ and $\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \le \varepsilon$ are neighbors in the graph. The arguments provided in Section 6.2.5 now apply in exactly the same way to show that nodes $i \in I_\circ$ such that $K_i = 1$ are connected in the graph. The same is true of nodes $i \in I_\circ$ such that $K_i = 2$.

6.3.3 The Clusters are (essentially) Disconnected in the Graph

Objective. *We show that the two sets of nodes $\{i \in I_\circ : K_i = 1\}$ and $\{i \in I_\circ : K_i = 2\}$—shown to be connected above—are disconnected in the graph.*

When we take $i, j \in I_\star$ such that $K_i \ne K_j$, we have the equivalent of (100), meaning,

$$\|\boldsymbol{Q}_i - \boldsymbol{Q}_j\| \ge \sin\theta_S - 12\kappa(C_{21} + 1/2)\varepsilon - 2\zeta'.$$

We choose η smaller than the RHS, so that these nodes are not neighbors in the graph.

We next prove that a point indexed by $I_\star$ is not neighbor to a point near the intersection because of different estimates for the local dimension. Let C denote a positive constant which increases with each appearance. Take $\boldsymbol{s} \in S_1$ such that $\delta(\boldsymbol{s}) := \mathrm{dist}(\boldsymbol{s}, S_2) \le r$. Reinstate the notation used in Section 6.2.3, in particular $\boldsymbol{s}^0$, T_1^0 and T_2^0, as well as $\boldsymbol{\Sigma}_T^0(\boldsymbol{s})$. Define $\boldsymbol{\Sigma}_T(\boldsymbol{s})$ as in Lemma 26. Also, let $T_1' = T_{S_1}(\boldsymbol{s})$ and $T_2' = T_{S_2}(\boldsymbol{s}^2)$ where $\boldsymbol{s}^2 := P_{S_2}(\boldsymbol{s})$. By Lemma 26 and Weyl's inequality, we have

$$\beta_{d+1}(\boldsymbol{\Sigma}(\boldsymbol{s})) \ge \beta_{d+1}(\boldsymbol{\Sigma}_T(\boldsymbol{s})) - C_{26}r^3, \qquad \beta_1(\boldsymbol{\Sigma}(\boldsymbol{s})) \le \beta_1(\boldsymbol{\Sigma}_T(\boldsymbol{s})) + C_{26}r^3,$$

which together with Lemma 25 (and proper scaling), implies that

$$\frac{\beta_{d+1}(\boldsymbol{\Sigma}(\boldsymbol{s}))}{\beta_1(\boldsymbol{\Sigma}(\boldsymbol{s}))} \ge \frac{\frac{c}{8}(1-\cos\theta_{\max}(T_1', T_2'))^2(1-\delta(\boldsymbol{s})^2/r^2)^{d/2+1} - C_{26}r^3}{c + (\delta(\boldsymbol{s})/r)^2(1-\delta(\boldsymbol{s})^2/r^2)^{d/2} + C_{26}r^3}.$$

Then, by the triangle inequality,

$$\theta_{\max}(T_1', T_2') \ge \theta_{\max}(T_1^0, T_2^0) - \theta_{\max}(T_1', T_1^0) - \theta_{\max}(T_2', T_2^0).$$

By definition, $\theta_{\max}(T_1^0, T_2^0) \ge \theta_S$, and by Lemma 3,

$$\theta_{\max}(T_1', T_1^0) \le \sin^{-1}\left(6\kappa\|\boldsymbol{s} - \boldsymbol{s}^0\| \wedge 1\right) \le Cr,$$

and similarly,

$$\theta_{\max}(T_2', T_2^0) \le \sin^{-1}\left(6\kappa\|\boldsymbol{s}^2 - \boldsymbol{s}^0\| \wedge 1\right) \le Cr,$$

because $\|\boldsymbol{s} - \boldsymbol{s}^0\| \le Cr$ by Lemma 21 and then $\|\boldsymbol{s}^2 - \boldsymbol{s}^0\| \le r + \|\boldsymbol{s} - \boldsymbol{s}^0\| \le Cr$. Hence, for r small enough, $\theta_{\max}(T_1', T_2') \ge \theta_S/2$, and furthermore,

$$\frac{\beta_{d+1}(\boldsymbol{\Sigma}(\boldsymbol{s}))}{\beta_1(\boldsymbol{\Sigma}(\boldsymbol{s}))} \ge \sqrt{\eta} \quad \text{when} \quad 1 - \delta(\boldsymbol{s})^2/r^2 \ge \xi := C_*\eta^{1/(d+2)}, \tag{118}$$

where C_* is a large enough constant, assuming r/η and η are small enough. The same is true for points on $\boldsymbol{s} \in S_2$ if we redefine $\delta(\boldsymbol{s}) = \mathrm{dist}(\boldsymbol{s}, S_1)$. Hence, for $\boldsymbol{s}_i$ close enough to the intersection that $\delta(\boldsymbol{s}_i)$ satisfies (118), $d_i > d$. Then, by Lemma 18, $\|\boldsymbol{Q}_i - \boldsymbol{Q}_j\| = 1$ for any $j \in I_\star$. By our choice of $\eta < 1$, this means that i and j are not neighbors.

So the only way $\{i \in I_\star : K_i = 1\}$ and $\{i \in I_\star : K_i = 2\}$ are connected in the graph is if there are $\boldsymbol{s}_i \in S_1$ and $\boldsymbol{s}_j \in S_2$ such that $\|\boldsymbol{s}_i - \boldsymbol{s}_j\| \le \varepsilon$ and both $\delta(\boldsymbol{s}_i)$ and $\delta(\boldsymbol{s}_j)$ fail to satisfy (118). We now show this is not possible. Let $\boldsymbol{\Sigma}_{T,i} = \boldsymbol{\Sigma}_T(\boldsymbol{s}_i)$ and define $\boldsymbol{\Sigma}_{T,j}$ similarly. (These are well-defined when $\varepsilon < 1/\kappa$, which we assume.) By Lemma 26, we have

$$\|\boldsymbol{\Sigma}_i - \boldsymbol{\Sigma}_{T,i}\| \le C_{26}\, r^3. \tag{119}$$

Define $\alpha_i = (1+t_i^d)^{-1}$ with $t_i := (1-\delta^2(\boldsymbol{s}_i))_+^{1/2}$. As in (77), with the triangle inequality, we get

$$\begin{aligned}\|\boldsymbol{\Sigma}_{T,i} - \alpha_i c r^2 \boldsymbol{P}_i\| &\leq c(1-\alpha_i)t_i^2 r^2 + \alpha_i(1-\alpha_i)\delta^2(\boldsymbol{s}_i) \\ &\leq 2(1-\alpha_i)r^2 \leq 2(1-\delta(\boldsymbol{s}_i)^2/r^2)_+^{d/2} r^2 \leq 2\xi^{d/2} r^2,\end{aligned}$$

where the second inequality we used the fact that $\alpha_i = 1$ if $\delta(\boldsymbol{s}_i) > r$, and in very last inequality comes from $\delta(\boldsymbol{s}_i)$ not satisfying (118). Hence, under Ω_2, with (119) and the triangle inequality, we get

$$\|\boldsymbol{C}_i - \alpha_i c r^2 \boldsymbol{P}_i\| \leq r^2\eta/C_\bullet + 3\kappa r^3 + 2\xi^{d/2} r^2 + C_{26}\, r^3.$$

Since $\beta_d(\boldsymbol{P}_i) = 1$ and $\beta_{d+1}(\boldsymbol{P}_i) = 0$, by Lemma 19, we have

$$\|\boldsymbol{Q}_i - \boldsymbol{P}_i\| \leq \frac{1}{\alpha_i c r^2}\left[r^2\eta/C_\bullet + 2\xi^{d/2} r^2 + (3\kappa + C_{26})r^3\right] \leq C(\eta/C_\bullet + \xi^{d/2} + r).$$

Similarly,

$$\|\boldsymbol{Q}_j - \boldsymbol{P}_j\| \leq C(\eta/C_\bullet + \xi^{d/2} + r).$$

By Lemma 18, $\|\boldsymbol{P}_i - \boldsymbol{P}_j\| = \sin\theta_{\max}(T_i, T_j)$. Let $\boldsymbol{s}^0 = P_{S_1\cap S_2}(\boldsymbol{s}_i)$, and define T_1^0 and T_2^0 as before. Not that $\|\boldsymbol{s}_i - \boldsymbol{s}^0\| \leq C\varepsilon$ by Lemma 21 and the fact that $\mathrm{dist}(\boldsymbol{s}_i, S_2) \leq \|\boldsymbol{s}_i - \boldsymbol{s}_j\| \leq \varepsilon$. Also, $\|\boldsymbol{s}_j - \boldsymbol{s}^0\| \leq \|\boldsymbol{s}_i - \boldsymbol{s}^0\| + \|\boldsymbol{s}_j - \boldsymbol{s}_i\| \leq C\varepsilon$. We then have

$$\theta_{\max}(T_i, T_j) \geq \theta_{\max}(T_1^0, T_2^0) - \theta_{\max}(T_i, T_1^0) - \theta_{\max}(T_j, T_2^0) \geq \theta_S - C\varepsilon,$$

by Lemma 3. Hence, by the triangle inequality,

$$\begin{aligned}\|\boldsymbol{Q}_i - \boldsymbol{Q}_j\| &\geq \|\boldsymbol{P}_i - \boldsymbol{P}_j\| - \|\boldsymbol{Q}_i - \boldsymbol{P}_i\| - \|\boldsymbol{Q}_j - \boldsymbol{P}_j\| \\ &\geq \sin(\theta_S - C\varepsilon) - C(\eta/C_\bullet + \xi^{d/2} + r) \geq \frac{1}{2}\sin\theta_S > \eta,\end{aligned}$$

when the constant in (90) is large enough. (Recall the definition of ξ above.) Therefore i and j are not neighbors, as we needed to show.

6.4 Noisy Case

So far we only dealt with the case where $\tau = 0$ in (7). When $\tau > 0$, a sample point $\boldsymbol{x}_i$ is in general different than its corresponding point $\boldsymbol{s}_i$ sampled from one of the surfaces. However, when τ/r is small, this does not change things much. For one thing, the points are close to each other, since we have $\|\boldsymbol{x}_i - \boldsymbol{s}_i\| \leq \tau$ by assumption, and τ is small compared to r. And the corresponding covariance matrices are also close to each other. To see this, redefine $\Xi_i = \{j \neq i : \boldsymbol{x}_j \in N_r(\boldsymbol{x}_i)\}$ and $\boldsymbol{C}_i$ as the sample covariance of $\{\boldsymbol{x}_j : j \in \Xi_i\}$. Let $\boldsymbol{D}_i$ denote the sample covariance of $\{\boldsymbol{s}_j : j \in \Xi_i\}$. Let X be uniform over $\{\boldsymbol{x}_j : j \in \Xi_i\}$ and define $Y = \sum_j \boldsymbol{s}_j \mathbb{1}_{\{X=\boldsymbol{x}_j\}}$. Starting from (33), we have

$$\begin{aligned}\|\boldsymbol{D}_i - \boldsymbol{C}_i\| &= \|\operatorname{Cov}(X) - \operatorname{Cov}(Y)\| \\ &\leq \mathbb{E}\left[\|X-Y\|^2\right]^{1/2} \cdot \left(\mathbb{E}\left[\|X-\boldsymbol{x}_i\|^2\right]^{1/2} + \mathbb{E}\left[\|Y-\boldsymbol{x}_i\|^2\right]^{1/2}\right) \\ &\leq \tau\cdot(r+(r+\tau)) = r^2\big(2\tau/r + (\tau/r)^2\big),\end{aligned}$$

which is small compared to r^2, and r^2 is the operating scale for covariance matrices in our setting.

Using these facts, the arguments are virtually the same, except for some additional terms due to triangle inequalities, for example, $\|\boldsymbol{s}_i - \boldsymbol{s}_j\| - 2\tau \leq \|\boldsymbol{x}_i - \boldsymbol{x}_j\| \leq \|\boldsymbol{s}_i - \boldsymbol{s}_j\| + 2\tau$. In particular, this results in ζ in (97) being now redefined as $\zeta = \frac{3\tau}{r} + \eta/C_\bullet + (3 + C_{16})\kappa r$. We omit further technical details.

Acknowledgments

This work was partially supported by grants from the National Science Foundation (DMS 0915160, 0915064, 0956072, 1418386, 1513465). We would like to thank Jan Rataj for helpful discussion around Lemma 3 and Xu Wang for his sharp proofreading. We also gratefully acknowledge the comments, suggestions, and scrutiny of an anonymous referee. We would also like to acknowledge support from the Institute for Mathematics and its Applications (IMA). For one thing, the authors first learned about the research of Goldberg et al. (2009) there, at the *Multi-Manifold Data Modeling and Applications* workshop in the Fall of 2008, and this was the main inspiration for our paper. Also, part of our work was performed while TZ was a postdoctoral fellow at the IMA, and also while EAC and GL were visiting the IMA.

References

S. N. Afriat. Orthogonal and oblique projectors and the characteristics of pairs of vector spaces. *Proceedings of the Cambridge Philosophical Society*, 53:800–816, 1957.

S. Agarwal, J. Lim, L. Zelnik-Manor, P. Perona, D. Kriegman, and S. Belongie. Beyond pairwise clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 838–845, 2005.

S. Agarwal, K. Branson, and S. Belongie. Higher order learning with graphs. In *International Conference on Machine Learning (ICML)*, pages 17–24, 2006.

E. Arias-Castro. Clustering based on pairwise distances when the data is of mixed dimensions. *IEEE Transactions on Information Theory*, 57(3):1692 –1706, 2011.

E. Arias-Castro, G. Chen, and G. Lerman. Spectral clustering based on local linear approximations. *Electronic Journal of Statistics*, 5:1537–1587, 2011.

R. Basri and D. W. Jacobs. Lambertian reflectance and linear subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(2):218–233, 2003.

J.-D. Boissonnat, R. Dyer, and A. Ghosh. Constructing intrinsic delaunay triangulations of submanifolds. *arXiv preprint arXiv:1303.6493*, 2013.

P. Bradley, K. Bennett, and A. Demiriz. Constrained k-means clustering. Technical Report MSR-TR-2000-65, Microsoft Research, 2000.

M. R. Brito, E. L. Chávez, A. J. Quiroz, and J. E. Yukich. Connectivity of the mutual k-nearest-neighbor graph in clustering and outlier detection. *Statistics & Probability Letters*, 35(1):33–42, 1997.

G. Chen and G. Lerman. Spectral curvature clustering (scc). *International Journal of Computer Vision*, 81(3):317–330, 2009a.

G. Chen and G. Lerman. Foundations of a multi-way spectral clustering framework for hybrid linear modeling. *Foundations of Computational Mathematics*, 9(5):517–558, 2009b.

G. Chen, S. Atev, and G. Lerman. Kernel spectral curvature clustering (KSCC). In *IEEE International Conference on Computer Vision (ICCV) Workshops*, pages 765–772, 2009.

A. Cuevas, R. Fraiman, and B. Pateiro-López. On statistical properties of sets fulfilling rolling-type conditions. *Advances in Applied Probability*, 44(2):311–329, 2012.

C. Davis and W. M. Kahan. The rotation of eigenvectors by a perturbation. III. *SIAM Journal on Numerical Analysis*, 7:1–46, 1970.

E. Elhamifar and R. Vidal. Sparse manifold clustering and embedding. In *Advances in Neural Information Processing Systems (NIPS)*, volume 24, pages 55–63, 2011.

R. Epstein, P. Hallinan, and A. Yuille. 5 ± 2 eigenimages suffice: An empirical investigation of low-dimensional lighting models. In *IEEE Workshop on Physics-based Modeling in Computer Vision*, pages 108–116, 1995.

Z. Fan, J. Zhou, and Y. Wu. Multibody grouping by inference of multiple subspaces from high-dimensional data using oriented-frames. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):91–105, 2006.

H. Federer. Curvature measures. *Transactions of the American Mathematical Society*, 93: 418–491, 1959.

Z. Fu, W. Hu, and T. Tan. Similarity based vehicle trajectory clustering and anomaly detection. In *IEEE International Conference on Image Processing*, volume 2, pages 602–605, 2005.

A. Gionis, A. Hinneburg, S. Papadimitriou, and P. Tsaparas. Dimension induced clustering. In *International Conference on Knowledge Discovery in Data Mining (KDD)*, pages 51–60, 2005.

A. Goh and R. Vidal. Segmenting motions of different types by unsupervised manifold clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–6, 2007.

A.B. Goldberg, X. Zhu, A. Singh, Z. Xu, and R. Nowak. Multi-manifold semi-supervised learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 169–176, 2009.

D. Gong, X. Zhao, and G. Medioni. Robust multiple manifolds structure learning. In *International Conference on Machine Learning (ICML)*, pages 321–328, 2012.

Q. Guo, H. Li, W. Chen, I-F. Shen, and J. Parkkinen. Manifold clustering via energy minimization. In *International Conference on Machine Learning and Applications (ICMLA)*, pages 375–380, 2007.

G. Haro, G. Randall, and G. Sapiro. Stratification learning: Detecting mixed density and dimensionality in high dimensional point clouds. *Advances in Neural Information Processing Systems (NIPS)*, 19:553, 2007.

R. Heckel and H. Bölcskei. Noisy subspace clustering via thresholding. In *IEEE International Symposium on Information Theory (ISIT)*, pages 1382–1386, 2013.

J. Ho, M.-H. Yang, J. Lim, K.-C. Lee, and D. Kriegman. Clustering appearances of objects under varying illumination conditions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 1–11, 2003.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

D. N. Kaslovsky and F. G. Meyer. Optimal tangent plane recovery from noisy manifold samples. arXiv preprint arXiv:1111.4601v2, 2011.

D. Kushnir, M. Galun, and A. Brandt. Fast multiscale clustering and manifold identification. *Pattern Recognition*, 39(10):1876–1891, 2006.

K.-C. Lee, J. Ho, and D. J. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27 (5):684–698, 2005.

A. V. Little, Y. M. Jung, and M. Maggioni. Multiscale estimation of intrinsic dimensionality of data sets. In *Manifold Learning and its Applications*, pages 26–33, 2009.

U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

Y. Ma, A. Y. Yang, H. Derksen, and R. Fossum. Estimation of subspace arrangements with applications in modeling and segmenting mixed data. *SIAM Review*, 50(3):413–458, 2008.

M. Maier, M. Hein, and U. von Luxburg. Optimal construction of k-nearest-neighbor graphs for identifying noisy clusters. *Theoretical Computer Science*, 410(19):1749–1764, 2009.

V. J. Martínez and E. Saar. *Statistics of the Galaxy Distribution*. CRC press, Boca Raton, 2002.

A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems (NIPS)*, 2:849–856, 2002.

P. Niyogi, S. Smale, and S. Weinberger. Finding the homology of submanifolds with high confidence from random samples. *Discrete & Computational Geometry*, 39(1):419–441, 2008.

M. Polito and P. Perona. Grouping and dimensionality reduction by locally linear embedding. *Advances in Neural Information Processing Systems (NIPS)*, 14:1255–1262, 2001.

L. K. Saul and S. T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *Journal of Machine Learning Research*, 4:119–155, 2003.

A. Shashua, R. Zass, and T. Hazan. Multi-way clustering using super-symmetric non-negative tensor factorization. In *European Conference on Computer Vision (ECCV)*, pages 595–608, 2006.

M. Soltanolkotabi and E. J. Candès. A geometric analysis of subspace clustering with outliers. *Annals of Statistics*, 40(4):2195–2238, 2012.

M. Soltanolkotabi, E. Elhamifar, and E. J. Candès. Robust subspace clustering. *Annals of Statistics*, 42(2):669–699, 2014.

R. Souvenir and R. Pless. Manifold clustering. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 648–653, 2005.

G. W. Stewart and J. G. Sun. *Matrix Perturbation Theory*. Computer Science and Scientific Computing. Academic Press Inc., Boston, MA, 1990.

M. Tipping and C. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11(2):443–482, 1999.

J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12:389–434, 2012.

M. C. Tsakiris and R. Vidal. Filtrated spectral algebraic subspace clustering. In *IEEE International Conference on Computer Vision Workshops*, pages 28–36, 2015.

R. Valdarnini. Detection of non-random patterns in cosmological gravitational clustering. *Astronomy & Astrophysics*, 366:376–386, 2001.

R. Vidal and Y. Ma. A unified algebraic approach to 2-D and 3-D motion segmentation and estimation. *Journal of Mathematical Imaging and Vision*, 25(3):403–421, 2006.

G. Walther. Granulometric smoothing. *Annals of Statistics*, 25(6):2273–2299, 1997.

Y. Wang, Y. Jiang, Y. Wu, and Z.-H. Zhou. Spectral clustering on multiple manifolds. *IEEE Transactions on Neural Networks*, 22(7):1149–1161, 2011.

Y.-X. Wang and H. Xu. Noisy sparse subspace clustering. In *International Conference on Machine Learning (ICML)*, pages 89–97, 2013.

L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1601–1608, 2005.

T. Zhang, A. Szlam, Y. Wang, and G. Lerman. Hybrid linear modeling via local best-fit flats. *International Journal of Computer Vision*, pages 1–24, 2012.

Journal of Machine Learning Research 18 (2017) 1-33 Submitted 1/15; Revised 11/16; Published 2/17

On Perturbed Proximal Gradient Algorithms

Yves F. Atchadé YVESA@UMICH.EDU
University of Michigan,
1085 South University,
Ann Arbor, 48109, MI, United States,

Gersende Fort GERSENDE.FORT@TELECOM-PARISTECH.FR
LTCI, CNRS, Telecom ParisTech, Université Paris-Saclay,
46, rue Barrault 75013 Paris, France,

Eric Moulines ERIC.MOULINES@POLYTECHNIQUE.EDU
CMAP, INRIA XPOP, Ecole Polytechnique,
91128 Palaiseau, France

Editor: Leon Bottou

Abstract

We study a version of the proximal gradient algorithm for which the gradient is intractable and is approximated by Monte Carlo methods (and in particular Markov Chain Monte Carlo). We derive conditions on the step size and the Monte Carlo batch size under which convergence is guaranteed: both increasing batch size and constant batch size are considered. We also derive non-asymptotic bounds for an averaged version. Our results cover both the cases of biased and unbiased Monte Carlo approximation. To support our findings, we discuss the inference of a sparse generalized linear model with random effect and the problem of learning the edge structure and parameters of sparse undirected graphical models.

Keywords: Proximal Gradient Methods; Stochastic Optimization; Monte Carlo approximations; Perturbed Majorization-Minimization algorithms.

1. Introduction

This paper deals with statistical optimization problems of the form:

$$\textbf{(P)} \quad \min_{\theta\in\mathbb{R}^d} F(\theta) \qquad \text{with } F = f + g \; .$$

This problem occurs in a variety of statistical and machine learning problems, where f is a measure of fit depending implicitly on some observed data and g is a regularization term that imposes structure to the solution. Typically, f is a differentiable function with a Lipschitz gradient, whereas g might be non-smooth (typical examples include sparsity inducing penalty).

H1 *The function $g:\ \mathbb{R}^d \to [0,+\infty]$ is convex, not identically $+\infty$, and lower semi-continuous. The function $f:\mathbb{R}^d\to\mathbb{R}$ is convex, continuously differentiable on $\mathbb{R}^d$ and there exists a finite non-negative constant L such that, for all $\theta,\theta'\in\mathbb{R}^d$,*

$$\|\nabla f(\theta) - \nabla f(\theta')\| \le L\|\theta-\theta'\| \; ,$$

where ∇f denotes the gradient of f.

We denote by Θ the domain of g: $\Theta \stackrel{\text{def}}{=} \{\theta\in\mathbb{R}^d : g(\theta) < \infty\}$.

H2 *The set $\operatorname{argmin}_{\theta\in\Theta} F(\theta)$ is a non empty subset of Θ.*

In this paper, we focus on the case where $f+g$ and ∇f are both intractable. This setting has not been widely considered despite the considerable importance of such models in statistics and machine learning. Intractable likelihood problems naturally occur for example in inference for bayesian networks (e.g. learning the edge structure and the parameters in an undirected graphical models), regression with latent variables or random effets, missing data, etc... In such applications, f is the negated log-likelihood of a conditional Gibbs measure π_θ known only up to a normalization constant and the gradient of $\nabla f(\theta)$ is typically expressed as a very high-dimensional integral w.r.t. the associated Gibbs measure $\nabla f(\theta) = \int H_\theta(x)\pi_\theta(\mathrm{d}x)$. Of course, this integral cannot be computed in closed form and should be approximated. Most often, some forms of Monte Carlo integration (such as Markov Chain Monte Carlo, or MCMC) is the only option.

To cope with problems where $f+g$ is intractable and possibly non-smooth, various methods have been proposed. Some of these works focused on stochastic sub-gradient and mirror descent algorithms; see Nemirovski et al. (2008); Duchi et al. (2011); Cotter et al. (2011); Lan (2012); Juditsky and Nemirovski (2012a,b). Other authors have proposed algorithms based on proximal operators to better exploit the smoothness of f and the properties of g (see e.g. Combettes and Wajs (2005); Hu et al. (2009); Xiao (2010); Juditsky and Nemirovski (2012a,b)).

The current paper focuses on the proximal gradient algorithm (see e.g. Beck and Teboulle (2010); Combettes and Pesquet (2011); Parikh and Boyd (2013) for literature review and further references). The proximal map (Moreau (1962)) associated to g is defined for $\gamma>0$ and $\theta\in\mathbb{R}^d$ by:

$$\operatorname{Prox}_{\gamma,g}(\theta) \stackrel{\text{def}}{=} \operatorname{argmin}_{\vartheta\in\Theta}\left\{ g(\vartheta) + \frac{1}{2\gamma}\|\vartheta-\theta\|^2 \right\}. \tag{1}$$

Note that under H1, there exists an unique point ϑ minimizing the RHS of (1) for any $\theta\in\mathbb{R}^d$ and $\gamma>0$. The proximal gradient algorithm is an iterative algorithm which, given an initial value $\theta_0\in\Theta$ and a sequence of positive step sizes $\{\gamma_n, n\in\mathbb{N}\}$, produces a sequence of parameters $\{\theta_n, n\in\mathbb{N}\}$ as follows:

Algorithm 1 (Proximal gradient algorithm) *Given θ_n, compute*

$$\theta_{n+1} = \operatorname{Prox}_{\gamma_{n+1},g}\left(\theta_n - \gamma_{n+1}\nabla f(\theta_n)\right) \; . \tag{2}$$

When $\gamma_n=\gamma$ for any n, it is known that the iterates of the proximal gradient algorithm $\{\theta_n, n\in\mathbb{N}\}$ (Algorithm 1) converges to θ_∞, this point is a fixed point of the proximal-gradient map

$$T_\gamma(\theta) \stackrel{\text{def}}{=} \operatorname{Prox}_{\gamma,g}\left(\theta - \gamma\nabla f(\theta)\right) \; . \tag{3}$$

©2017 Yves Atchadé, Gersende Fort and Eric Moulines.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-038.html.

Under H1 and H2, when $\gamma_n \, (0, 2/L]$ and $\inf_n \gamma_n > 0$, it is indeed known that the iterates of the proximal gradient algorithm $\{\theta_n,\, n \in \mathbb{N}\}$ defined in (2) converges to a point in the set $\mathcal{L}$ of the solutions of (P) which coincides with the fixed points of the mapping T_γ for any $\gamma \in (0, 2/L)$

$$\mathcal{L} \stackrel{\text{def}}{=} \operatorname{argmin}_{\theta\in\Theta} F(\theta) = \{\theta \in \Theta : \theta = T_\gamma(\theta)\} \; . \tag{4}$$

(see e.g. (Combettes and Wajs, 2005, Theorem 3.4. and Proposition 3.1.(iii))).

Since $\nabla f(\theta)$ is intractable, the gradient $\nabla f(\theta_n)$ at n-th iteration is replaced by an approximation H_{n+1}:

Algorithm 2 (Perturbed Proximal Gradient algorithm) *Let $\theta_0 \in \Theta$ be the initial solution and $\{\gamma_n,\, n \in \mathbb{N}\}$ be a sequence of posi–tive step–sizes. For $n \geq 1$, given $(\theta_0, \ldots, \theta_n)$ construct an approximation H_{n+1} of $\nabla f(\theta_n)$ and compute*

$$\theta_{n+1} = \operatorname{Prox}_{\gamma_{n+1},g} \left(\theta_n - \gamma_{n+1} H_{n+1}\right) \; . \tag{5}$$

We provide in Theorem 2 sufficient conditions on the perturbation $\eta_{n+1} = H_{n+1} - \nabla f(\theta_n)$ to obtain the convergence of the perturbed proximal gradient sequence given by (5). We then consider an averaging scheme of the perturbed proximal gradient algorithm: given non-negative weights $\{a_n,\, n \in \mathbb{N}\}$, Theorem 3 provides non-asymptotic bound of the deviation between $\sum_{k=1}^n a_k F(\theta_k) / \sum_{k=1}^n a_k$ and the minimum of F. Our results complement and extend Rosasco et al. (2014); Nitanda (2014); Xiao and Zhang (2014).

We then consider the case where the gradient $\nabla f(\theta) = \int_{\mathsf{X}} H_\theta(x) \pi_\theta(\mathrm{d}x)$ is defined as an expectation (see H3 in section 3). In this case, at each iteration $\nabla f(\theta_n)$ is approximated by a Monte Carlo average $H_{n+1} = m_{n+1}^{-1} \sum_{j=1}^{m_{n+1}} H_{\theta_n}(X_{n+1}^{(j)})$ where m_{n+1} is the size of the Monte Carlo batch and $\{X_{n+1}^{(j)}, 1 \leq j \leq m_{n+1}\}$ is the Monte Carlo batch. Two different settings are covered. In the first setting, the samples $\{X_{n+1}^{(j)}, 1 \leq j \leq m_{n+1}\}$ are conditionally independent and identically distributed (i.i.d.) with distribution π_{θ_n}. In such case, the conditional expectation of H_{n+1} given all the past iterations, denoted by $\mathbb{E}\left[H_{n+1} \,|\, \mathcal{F}_n\right]$ (see section 3), is equal to $\nabla f(\theta_n)$. In the second setting, the Monte Carlo batch $\{X_{n+1}^{(j)}, 1 \leq j \leq m_{n+1}\}$ is produced by running a MCMC algorithm. In such case, the conditional distribution of $X_{n+1}^{(j)}$ given the past is no longer exactly equal to π_{θ_n} which implies that $\mathbb{E}\left[H_{n+1} \,|\, \mathcal{F}_n\right] \neq \nabla f(\theta_n)$.

Theorem 4 (resp. Theorem 6) establish the convergence of the sequence $\{\theta_n, n \in \mathbb{N}\}$ when the batch size m_n is either fixed or increases with the number of iterations n. When the Monte Carlo batch $\{X_{n+1}^{(j)}, 1 \leq j \leq m_{n+1}\}$ is i.i.d. conditionally to the past the two theorems essentially say that with probability one, $\{\theta_n, n \in \mathbb{N}\}$ converges to an element of the set of minimizer $\mathcal{L}$ as soon as $\sum_n \gamma_n = +\infty$ and $\sum_n \gamma_{n+1}^2 / m_{n+1} < \infty$. Hence, one can choose either a fixed step size $\gamma_n = \gamma$ and a batch size $\{m_n, n \in \mathbb{N}\}$ increasing at least linearly (up to a logarithmic factor); or a decreasing step size and a fixed batch size $m_n = m$. When $\{X_{n+1}^{(j)}, 1 \leq j \leq m_{n+1}\}$ is produced by a MCMC algorithm (under appropriate assumptions) our theorems essentially say that the same convergence result holds if $\sum_n \gamma_n = \infty$ and $\sum_n \gamma_{n+1}^2 < \infty$ when $m_n = m$ is constant across iterations or $\sum_n \gamma_{n+1} / m_{n+1} < \infty$ if the batch size is increased.

Theorem 4 and Theorem 6 also provide non asymptotic bounds for the difference $\Delta_n = \sum_{k=1}^n a_k F(\theta_k) / \sum_{k=1}^n a_k - \min F$ in L^q-norm for $q \geq 1$. When the batch size sequence m_{n+1} increases linearly at each iteration while the step size γ_{n+1} is held constant, $\Delta_n = O(\ln n / n)$. We recover (up to a logarithmic factor) the rate of the proximal gradient algorithm. If we now compare the complexity of the algorithms in terms of the number of simulations N needed (and not the number of iterations), the error bound decreases like $O(N^{-1/2})$. The same error bound can be achieved by choosing a fixed batch size and a decreasing step size $\gamma_n = O(1/\sqrt{n})$.

In section 4, these results are illustrated with the problem of estimating a high-dimensional discrete graphical models. In section 5, we consider high-dimensional random effect logistic regression model. All the proofs are postponed to section 6.

2. Perturbed proximal gradient algorithms

The key property to study the behavior of the sequence the perturbed proximal gradient algorithm is the following elementary lemma which might be seen as a deterministic version of the Robbins-Siegmund lemma (see e.g. (Polyak, 1987, Lemma 11, Chapter 2)). It replaces in our analysis (Combettes, 2001, Lemma 3.1) for quasi-Fejer sequences and modified Fejer monotone sequences (see Lin et al. (2015)). Compared to the Robbins-Siegmund Lemma, the sequence $(\xi_n)_n$ is not assumed to be nonnegative. When applied in the stochastic context as in Section 3, the fact that the result is purely deterministic and deals with signed perturbations ξ_n allows more flexibility in the study of the dynamics.

Lemma 1 *Let $\{v_n,\, n \in \mathbb{N}\}$ and $\{\chi_n,\, n \in \mathbb{N}\}$ be non-negative sequences and $\{\xi_n,\, n \in \mathbb{N}\}$ be such that $\sum_n \xi_n$ exists. If for any $n \geq 0$,*

$$v_{n+1} \leq v_n - \chi_{n+1} + \xi_{n+1}$$

then $\sum_n \chi_n < \infty$ and $\lim_n v_n$ exists.

Proof See Section 6.2.1 ∎

Applied with $v_n = \|\theta_n - \theta_\star\|$ for some $\theta_\star \in \mathcal{L}$, this lemma is the key result for the proof of the following theorem, which provides sufficient conditions on the stepsize sequence $\{\gamma_n,\, n \in \mathbb{N}\}$ and on the *approximation error*:

$$\eta_{n+1} \stackrel{\text{def}}{=} H_{n+1} - \nabla f(\theta_n) \; , \tag{6}$$

for the sequence $\{\theta_n,\, n \in \mathbb{N}\}$ to converge to a point θ_∞ in the set $\mathcal{L}$ of the minimizers of F. Denote by $\langle \cdot, \cdot \rangle$ the usual inner product on $\mathbb{R}^d$ associated to the norm $\|\cdot\|$.

Theorem 2 *Assume H1 and H2. Let $\{\theta_n,\, n \in \mathbb{N}\}$ be given by Algorithm 2 with step sizes satisfying $\gamma_n \in (0, 1/L]$ for any $n \geq 1$ and $\sum_n \gamma_n = +\infty$. If the following series converge*

$$\sum_{n\geq 0} \gamma_{n+1} \left\langle T_{\gamma_{n+1}}(\theta_n), \eta_{n+1} \right\rangle \; , \quad \sum_{n\geq 0} \gamma_{n+1} \eta_{n+1} \; , \quad \sum_{n\geq 0} \gamma_{n+1}^2 \|\eta_{n+1}\|^2 \; , \tag{7}$$

then there exists $\theta_\infty \in \mathcal{L}$ such that $\lim_n \theta_n = \theta_\infty$.

Proof See Section 6.2.2. ■

Theorem 2 applied with $\eta_{n+1} = 0$ provides sufficient conditions for the convergence of Algorithm 1 to $\mathcal{L}$: the algorithm converges as soon as $\gamma_n \in (0, 1/L]$ and $\sum_n \gamma_n = +\infty$.

Sufficient conditions for the convergence of $\{\theta_n,\ n \in \mathbb{N}\}$ are also provided in Combettes and Wajs (2005). When applied to our settings (Combettes and Wajs, 2005, Theorem 3.4.) requires $\sum_n \|\eta_{n+1}\| < \infty$ and $\inf_n \gamma_n > 0$, which for instance cannot accommodate the fixed Monte Carlo batch size stochastic algorithms considered in this paper. The same limitation applies to the analysis of the stochastic quasi-Fejer iterations (see Combettes and Pesquet (2015a)) which in our particular case requires $\sum_n \gamma_{n+1}\|\eta_{n+1}\| < \infty$. These conditions are weakened in Theorem 2. However in all fairness we should mention that unlike the present work, Combettes and Wajs (2005) and Combettes and Pesquet (2015a) deal with infinite-dimensional problems which raises additional technical difficulties, and study algorithms that include a relaxation parameter. Furthermore, in the case where $\eta_n \equiv 0$, larger values of the stepsize γ_n are allowed ($\gamma_n \in (0, 2/L]$).

Let $\{a_0, \cdots, a_n\}$ be non-negative real numbers. Theorem 3 provides a control of the weighted sum $\sum_{k=1}^n a_k(F(\theta_k) - \min F)$.

Theorem 3 *Assume H1 and H2. Let $\{\theta_n,\ n \in \mathbb{N}\}$ be given by Algorithm 2 with $\gamma_n \in (0, 1/L]$ for any $n \geq 1$. For any non-negative weights $\{a_0, \cdots, a_n\}$, any $\theta_\star \in \mathcal{L}$ and any $n \geq 1$,*

$$\sum_{k=1}^n a_k \left\{F(\theta_k) - \min F\right\} \leq U_n(\theta_\star)$$

where T_γ and η_n are given by (3) and (6) respectively and

$$U_n(\theta_\star) \stackrel{\text{def}}{=} \frac{1}{2}\sum_{k=1}^n \left(\frac{a_k}{\gamma_k} - \frac{a_{k-1}}{\gamma_{k-1}}\right)\|\theta_{k-1} - \theta_\star\|^2 + \frac{a_0}{2\gamma_0}\|\theta_0 - \theta_\star\|^2 - \sum_{k=1}^n a_k \left\langle T_{\gamma_k}(\theta_{k-1}) - \theta_\star, \eta_k\right\rangle + \sum_{k=1}^n a_k\gamma_k\|\eta_k\|^2 \,. \quad (8)$$

Proof See Section 6.2.3. ■

When applied with $\eta_n = 0$, Theorem 3 gives an explicit bound of the difference $\Delta_n = A_n^{-1}\sum_{j=1}^n a_j F(\theta_j) - \min F$ where $A_n = \sum_{k=1}^n a_k$ for the (exact) proximal gradient sequence $\{\theta_n,\ n \in \mathbb{N}\}$ given by Algorithm 1. When the sequence $\{a_n/\gamma_n, n \geq 1\}$ is non decreasing, (8) shows that $\Delta_n = O(a_n A_n^{-1}\gamma_n^{-1})$.

Taking $a_k = 1$ for any $k \geq 0$ provides a bound for the cumulative regret. When $a_k = 1$, $\gamma_k = 1/L$ for any $k \geq 0$, (Schmidt et al., 2011, Proposition 1) provides a bound of order $O(1)$ under the assumption that $\sum_n \|\eta_{n+1}\| < \infty$. Using the inequality $|\left\langle T_{1/L}(\theta_k) - \theta_\star, \eta_{k+1}\right\rangle| \leq \|\theta_k - \theta_\star\|\|\eta_{k+1}\|$ (see Lemma 9), the upper bound $U_n(\theta_\star)$ in (8) is also $O(1)$.

When $a_n = \gamma_n$ for any $n \geq 0$, then $\sup_n U_n(\theta_\star) < \infty$ under the assumptions that the series

$$\sum_n \gamma_n \left\langle T_{\gamma_n}(\theta_{n-1}) - \theta_\star, \eta_n\right\rangle \,, \qquad \sum_n \gamma_n^2\|\eta_n\|^2 \,,$$

converge. In this case, we have

$$\left(\frac{\sum_{k=1}^n \gamma_k F(\theta_k)}{\sum_{k=1}^n \gamma_k} - \min F\right) = O\left(\left(\sum_{k=1}^n \gamma_k\right)^{-1}\right).$$

Consider the weighted averaged sequence $\{\bar{\theta}_n,\ n \in \mathbb{N}\}$ defined by

$$\bar{\theta}_n \stackrel{\text{def}}{=} \frac{1}{A_n}\sum_{k=1}^n a_k\theta_k \,. \quad (9)$$

Under H1 and H2, F is convex so that $F\left(\bar{\theta}_n\right) \leq A_n^{-1}\sum_{k=1}^n a_k F(\theta_k)$. Therefore, Theorem 3 also provides convergence rates for $F(\bar{\theta}_n) - \min F$.

3. Stochastic Proximal Gradient algorithm

In this section, it is assumed that H_{n+1} is a Monte Carlo approximation of $\nabla f(\theta_n)$, where $\nabla f(\theta)$ satisfies the following assumption:

H3 *for all $\theta \in \Theta$,*

$$\nabla f(\theta) = \int_{\mathsf{X}} H_\theta(x)\pi_\theta(\mathrm{d}x) \,, \quad (10)$$

for some probability measure π_θ on a measurable space $(\mathsf{X}, \mathcal{X})$ and an integrable function $(\theta, x) \mapsto H_\theta(x)$ from $\Theta \times \mathsf{X}$ to Θ.

Note that X is not necessarily a topological space, even if, in many applications, $\mathsf{X} \subseteq \mathbb{R}^d$.

Assumption H3 holds in many problems (see section 4 and section 5). To approximate $\nabla f(\theta)$, several options are available. Of course, when the dimension of the state space X is small to moderate, it is always possible to perform a numerical integration using either Gaussian quadratures or low-discrepancy sequences. Another possibility is to approximate these integrals: nested Laplace approximations have been considered recently for example in Schelldorfer et al. (2014) and further developed in Ogden (2015). Such approximations necessarily introduce some bias, which might be difficult to control. In addition, these techniques are not applicable when the dimension of the state space X becomes large. In this paper, we rather consider some form of Monte Carlo approximation.

When sampling π_θ is doable, then an obvious choice is to use a naive Monte Carlo estimator which amounts to sample a batch $\{X_{n+1}^{(j)}, 1 \leq j \leq m_{n+1}\}$ independently of the past values of the parameters $\{\theta_j, j \leq n\}$ and of the past draws i.e. independently of the σ-algebra

$$\mathcal{F}_n \stackrel{\text{def}}{=} \sigma(\theta_0, X_k^{(j)}, 0 \leq k \leq n, 0 \leq j \leq m_k) \,. \quad (11)$$

We then form

$$H_{n+1} = m_{n+1}^{-1}\sum_{j=1}^{m_{n+1}} H_{\theta_n}(X_{n+1}^{(j)}) \,.$$

Conditionally to $\mathcal{F}_n$, H_{n+1} is an unbiased estimator of $\nabla f(\theta_n)$. The batch size m_{n+1} can either be chosen to be fixed across iterations or to increase with n at a certain rate. In the first case, H_{n+1} is not converging. In the second case, the approximation error is vanishing. The fixed batch-size case is closely related to Robbins-Monro stochastic approximation (the mitigation of the error is performed by letting the stepsize $\gamma_n \to 0$); the increasing batch-size case is related to Monte Carlo assisted optimization; see for example Geyer (1994).

The situation that we are facing in section 4 and section 5 is more complicated because direct sampling from π_θ is not an option. Nevertheless, it is fairly easy to construct a Markov kernel P_θ with invariant distribution π_θ. Monte Carlo Markov Chains (MCMC) provide a set of principled tools to sample from complex distributions over large dimensional spaces. In such case, conditional to the past, $\{X_{n+1}^{(j)}, 1 \le j \le m_{n+1}\}$ is a realization of a Markov chain with transition kernel P_{θ_n} and started from $X_n^{(m_n)}$ (the last sample draws in the previous minibatch).

Recall that a Markov kernel P is an application on $\mathsf{X} \times \mathcal{X}$, taking values in $[0,1]$ such that for any $x \in \mathsf{X}$, $P(x,\cdot)$ is a probability measure on $\mathcal{X}$; and for any $A \in \mathcal{X}$, $x \mapsto P(x,A)$ is measurable. Furthermore, if P is a Markov kernel on X, we denote by P^k the k-th iterate of P defined recursively as $P^0(x,A) \stackrel{\text{def}}{=} \mathbb{1}_A(x)$, and $P^k(x,A) \stackrel{\text{def}}{=} \int P^{k-1}(x,\mathrm{d}z)P(z,A)$, $k \ge 1$. Finally, the kernel P acts on probability measure: for any probability measure μ on $\mathcal{X}$, μP is a probability measure defined by

$$\mu P(A) \stackrel{\text{def}}{=} \int \mu(\mathrm{d}x)P(x,A), \qquad A \in \mathcal{X};$$

and P acts on positive measurable functions: for a measurable function $f : \mathsf{X} \to \mathbb{R}_+$, Pf is a function defined by

$$Pf(x) \stackrel{\text{def}}{=} \int f(y)\, P(x,\mathrm{d}y).$$

We refer the reader to Meyn and Tweedie (2009) for the definitions and basic properties of Markov chains.

In this Markovian setting, it is possible to consider the fixed batch case and the increasing batch case. From a mathematical standpoint, the fixed batch case is trickier, because H_{n+1} is no longer an unbiased estimator of $\nabla f(\theta_n)$, i.e. the bias B_n defined by

$$\begin{aligned} B_n \stackrel{\text{def}}{=} \mathbb{E}\left[H_{n+1} \,|\, \mathcal{F}_n\right] - \nabla f(\theta_n) &= m_{n+1}^{-1} \textstyle\sum_{j=1}^{m_{n+1}} \mathbb{E}\left[H_{\theta_n}(X_{n+1}^{(j)}) \,\middle|\, \mathcal{F}_n \right] - \nabla f(\theta_n) \\ &= m_{n+1}^{-1} \textstyle\sum_{j=1}^{m_{n+1}} P_{\theta_n}^j H_{\theta_n}(X_{n+1}^{(0)}) - \nabla f(\theta_n)\,, \end{aligned} \quad (12)$$

does not vanish. When $m_n = m$ is small, the bias can even be pretty large, and the way the bias is mitigated in the algorithm requires substantial mathematical developments, which are not covered by the results currently available in the literature (see e.g. Combettes and Pesquet (2015a); Rosasco et al. (2014); Combettes and Pesquet (2015b); Rosasco et al. (2015); Lin et al. (2015)).

To capture in a common unifying framework these two different situations we assume that

H4 *H_{n+1} is a Monte Carlo approximation of the expectation $\nabla f(\theta_n)$:*

$$H_{n+1} = m_{n+1}^{-1} \sum_{j=1}^{m_{n+1}} H_{\theta_n}(X_{n+1}^{(j)})\,;$$

for all $n \ge 0$, conditionally to the past, $\{X_{n+1}^{(j)}, 1 \le j \le m_{n+1}\}$ is a Markov chain started from $X_n^{(m_n)}$ and with transition kernel P_{θ_n} (we set $X_0^{(m_0)} = x_\star \in \mathsf{X}$). For all $\theta \in \Theta$, P_θ is a Markov kernel with invariant distribution π_θ.

For a measurable function $V : \mathsf{X} \to [1,\infty)$, a signed measure μ on the σ-field of X, and a function $f : \mathsf{X} \to \mathbb{R}$, define

$$|f|_V \stackrel{\text{def}}{=} \sup_{x \in \mathsf{X}} \frac{|f(x)|}{V(x)}\,, \qquad \|\mu\|_V \stackrel{\text{def}}{=} \sup_{f,|f|_V \le 1} \left| \int f \,\mathrm{d}\mu \right|\,.$$

H5 *There exist $\lambda \in (0,1)$, $b < \infty$, $p \ge 2$ and a measurable function $W : \mathsf{X} \to [1,+\infty)$ such that*

$$\sup_{\theta \in \Theta} |H_\theta|_W < \infty\,, \qquad \sup_{\theta \in \Theta} P_\theta W^p \le \lambda W^p + b\,.$$

In addition, for any $\ell \in (0,p]$, there exist $C < \infty$ and $\rho \in (0,1)$ such that for any $x \in \mathsf{X}$,

$$\sup_{\theta \in \Theta} \|P_\theta^n(x,\cdot) - \pi_\theta\|_{W^\ell} \le C\rho^n W^\ell(x)\,. \quad (13)$$

Sufficient conditions for the uniform-in-θ ergodic behavior (13) are given e.g. in (Fort et al., 2011, Lemma 2.3), in terms of aperiodicity, irreducibility and minorization conditions on the kernels $\{P_\theta, \theta \in \Theta\}$. Examples of MCMC kernels P_θ satisfying this assumption can be found in (Andrieu and Moulines, 2006, Proposition 12), (Saksman and Vihola, 2010, Proposition 15), (Fort et al., 2011, Proposition 3.1.), (Schreck et al., 2013, Proposition 3.2.), (Allassonnière and Kuhn, 2015, Proposition 1), and (Fort et al., 2015, Proposition 3.1.).

The proof of the results below consists in verifying the conditions of Theorem 2 with the error term defined by $\eta_{n+1} = m_{n+1}^{-1} \sum_{j=1}^{m_{n+1}} H_{\theta_n}(X_{n+1}^{(j)}) - \nabla f(\theta_n)$. If the approximation is unbiased in the sense that $\mathbb{E}\left[\eta_{n+1} \,|\, \mathcal{F}_n\right] = 0$, then $\{\eta_n,\, n \in \mathbb{N}\}$ is a martingale increment sequence. In all the other cases, we decompose η_{n+1} as the sum of a martingale increment term and a remainder term. When the batch size $\{m_n,\, n \in \mathbb{N}\}$ is increasing, the martingale increment sequence can be set to $\eta_{n+1} - \mathbb{E}\left[\eta_{n+1} \,|\, \mathcal{F}_n\right]$ and the remainder term $\mathbb{E}\left[\eta_{n+1} \,|\, \mathcal{F}_n\right]$ will be shown to be vanishingly small. When the batch size $\{m_n,\, n \in \mathbb{N}\}$ is constant, then $\mathbb{E}\left[\eta_{n+1} \,|\, \mathcal{F}_n\right]$ does not vanish. A more subtle definition of the martingale increment has to be done, introducing the Poisson equation for Markov chain (see Proposition 19 in section 6).

3.1 Monte Carlo approximation with fixed batch-size

We first study the case when $m_n = m$ for any $n \in \mathbb{N}$. Theorem 4 provides sufficient conditions for the convergence towards the limiting set $\mathcal{L}$ and for a bound for $\sum_{k=1}^n a_k F(\theta_k) - \min F$. Consider the following assumption

H6 *(i) there exists a constant C such that for any $\theta, \theta' \in \Theta$*

$$|H_\theta - H_{\theta'}|_W + \sup_x \frac{\|P_\theta(x,\cdot) - P_{\theta'}(x,\cdot)\|_W}{W(x)} + \|\pi_\theta - \pi_{\theta'}\|_W \le C\,\|\theta - \theta'\| .$$

(ii) $\sup_{\gamma\in(0,1/L]} \sup_{\theta\in\Theta} \gamma^{-1}\ \|\mathrm{Prox}_{\gamma,g}(\theta) - \theta\| < \infty$.

(iii) $\sum_n |\gamma_{n+1} - \gamma_n| < \infty$.

Assumption H6-(i) requires a Lipschitz-regularity in the parameter θ of the Markov kernel P_θ which, for MCMC algorithms, is inherited under mild additional conditions from the Lipschitz regularity in W-norm of the target distribution. Such conditions have been worked out for general families of MCMC kernels including Hastings-Metropolis dynamics, Gibbs samplers, and hybrid MCMC algorithm; see for example Proposition 12 in Andrieu and Moulines (2006), the proof of Theorem 3.4. in Fort et al. (2011), Lemmas 4.6. and 4.7. in Fort et al. (2015) and the references therein. It is a classical assumption when studying Stochastic Approximation with conditionally Markovian dynamic (see e.g. Benveniste et al. (1990), Andrieu et al. (2005), Fort et al. (2014)).

We prove in Proposition 11 that when g is proper, convex, Lipschitz on Θ, then H6-(ii) is satisfied. In particular, if Θ is a closed convex set, H6-(ii) is satisfied with the Lasso or fused Lasso penalty. If Θ is a compact convex set, then H6-(ii) is satisfied by the elastic-net penalty.

For a random variable Y, denote by $\|Y\|_{L^q} = (\mathbb{E}[|Y|^q])^{1/q}$.

Theorem 4 *Assume Θ is bounded. Let $\{\theta_n, n \ge 0\}$ be given by Algorithm 2 with $\gamma_n \in (0, 1/L]$ for any $n \ge 0$. Assume H1–H5, $m_n = m \ge 1$ and, if the Monte Carlo approximation is biased, assume also H6.*

(i) Assume that $\sum_n \gamma_n = \infty$ and $\sum_n \gamma_n^2 < \infty$. With probability one, there exists $\theta_\infty \in \mathcal{L}$ such that $\lim_{n\to\infty} \theta_n = \theta_\infty$.

(ii) For any $q \in (1, p/2]$ there exists a constant C such that for any non-negative numbers $\{a_0, \cdots, a_n\}$

$$\left\| \sum_{k=1}^n a_k \{F(\theta_k) - \min F\} \right\|_{L^q} \le C \left(\frac{a_0}{\gamma_0} + \sum_{k=1}^n \left| \frac{a_k}{\gamma_k} - \frac{a_{k-1}}{\gamma_{k-1}} \right| + \left(\sum_{k=1}^n a_k^2 \right)^{1/2} + \sum_{k=1}^n a_k\gamma_k + \upsilon \sum_{k=1}^n |a_k - a_{k-1}| \right)$$

and

$$\sum_{k=1}^n a_k \{\mathbb{E}[F(\theta_k)] - \min F\} \le C \left(\frac{a_0}{\gamma_0} + \sum_{k=1}^n \left| \frac{a_k}{\gamma_k} - \frac{a_{k-1}}{\gamma_{k-1}} \right| + \sum_{k=1}^n a_k\gamma_k + \upsilon \sum_{k=1}^n |a_k - a_{k-1}| \right)$$

where $\upsilon = 0$ if the Monte Carlo approximation is unbiased and $\upsilon = 1$ otherwise.

Proof The proof is postponed to Section 6.3. ■

When $a_n = 1$ and $\gamma_n = (n+1)^{-1/2}$, Theorem 4 shows that when $n \to \infty$,

$$\left\| n^{-1} \sum_{k=1}^n F(\theta_k) - \min F \right\|_{L^q} = O\left(\frac{1}{\sqrt{n}} \right) .$$

An upper bound $O(\ln n/\sqrt{n})$ can be obtained from Theorem 4 by choosing $a_n = \gamma_n = (n+1)^{-1/2}$.

3.2 Monte Carlo approximation with increasing batch size

The key property to discuss the asymptotic behavior of the algorithm is the following result

Proposition 5 *Assume H3, H4 and H5. There exists a constant C such that w.p. 1 for any $n \ge 0$,*

$$\|\mathbb{E}[\,\eta_{n+1} \,|\, \mathcal{F}_n]\,\| \le C\, m_{n+1}^{-1} W(X_n^{(m_n)}) , \qquad \mathbb{E}[\|\eta_{n+1}\|^p | \mathcal{F}_n] \le C\ m_{n+1}^{-p/2} W^p(X_n^{(m_n)}) .$$

Proof The first inequality follows from (12) and (13). The second one is established in (Fort and Moulines, 2003, Proposition 12). ■

Theorem 6 *Assume Θ is bounded. Let $\{\theta_n, n \ge 0\}$ be given by Algorithm 2 with $\gamma_n \in (0, 1/L]$ for any $n \ge 0$. Assume H1–H5.*

(i) Assume $\sum_n \gamma_n = +\infty$, $\sum_n \gamma_{n+1}^2 m_{n+1}^{-1} < \infty$ and, if the approximation is biased, $\sum_n \gamma_{n+1} m_{n+1}^{-1} < \infty$. With probability one, there exists $\theta_\infty \in \mathcal{L}$ such that $\lim_{n\to\infty} \theta_n = \theta_\infty$.

(ii) For any $q \in (1, p/2]$, there exists a constant C such that for any non-negative numbers $\{a_0, \cdots, a_n\}$

$$\left\| \sum_{k=1}^n a_k \{F(\theta_k) - \min F\} \right\|_{L^q} \le C \left(\frac{a_0}{\gamma_0} + \sum_{k=1}^n \left| \frac{a_k}{\gamma_k} - \frac{a_{k-1}}{\gamma_{k-1}} \right| + \left(\sum_{k=1}^n a_k^2 m_k^{-1} \right)^{1/2} + \sum_{k=1}^n a_k\gamma_k m_k^{-1} + \upsilon \sum_{k=1}^n a_k m_k^{-1} \right)$$

and

$$\sum_{k=1}^n a_k \{\mathbb{E}[F(\theta_k)] - \min F\} \le C \left(\frac{a_0}{\gamma_0} + \sum_{k=1}^n \left| \frac{a_k}{\gamma_k} - \frac{a_{k-1}}{\gamma_{k-1}} \right| + \sum_{k=1}^n a_k\gamma_k m_k^{-1} + \upsilon \sum_{k=1}^n a_k m_k^{-1} \right) ,$$

where $\upsilon = 0$ if the Monte-Carlo approximation is unbiased and $\upsilon = 1$ otherwise.

Proof See Section 6.4. ■

Theorem 6 shows that when $n \to \infty$,

$$\left\| \left(\sum_{k=1}^{n} a_k \right)^{-1} \sum_{k=1}^{n} a_k F(\theta_k) - \min F \right\|_{L^q} = O\left(\frac{\ln n}{n} \right)$$

by choosing a fixed stepsize $\gamma_n = \gamma$, a linearly increasing batch-size $m_n \sim n$ and a uniform weight $a_n = 1$. Note that this is the rate after n iterations of the Stochastic Proximal Gradient algorithm but $\sum_{k=1}^{n} m_k = O(n^2)$ Monte Carlo samples. Therefore, the rate of convergence expressed in terms of complexity is $O(\ln n/\sqrt{n})$.

4. Application to network structure estimation

To illustrate the algorithm we consider the problem of fitting discrete graphical models in a setting where the number of nodes in the graph is large compared to the sample size. Let X be a nonempty finite set, and $p \geq 1$ an integer. We consider a graphical model on X^p with joint probability mass function

$$f_\theta(x_1, \ldots, x_p) = \frac{1}{Z_\theta} \exp\left\{ \sum_{k=1}^{p} \theta_{kk} B_0(x_k) + \sum_{1 \leq j < k \leq p} \theta_{kj} B(x_k, x_j) \right\}, \qquad (14)$$

for a non-zero function $B_0 : \mathsf{X} \to \mathbb{R}$ and a symmetric non-zero function $B : \mathsf{X} \times \mathsf{X} \to \mathbb{R}$. The term Z_θ is the normalizing constant of the distribution (the partition function), which cannot (in general) be computed explicitly. The real-valued symmetric matrix θ defines the graph structure and is the parameter of interest. It has the same interpretation as the precision matrix in a multivariate Gaussian distribution.

We consider the problem of estimating θ from N realizations $\{x^{(i)}, 1 \leq i \leq N\}$ from (14) where $x^{(i)} = (x_1^{(i)}, \ldots, x_p^{(i)}) \in \mathsf{X}^p$, and where the true value of θ is assumed sparse. This problem is relevant for instance in biology (Ekeberg et al. (2013); Kamisetty et al. (2013)), and has been considered by many authors in statistics and machine learning (Banerjee et al. (2008); Höfling and Tibshirani (2009); Ravikumar et al. (2010); Guo et al. (2010); Xue et al. (2012)).

The main difficulty in dealing with this model is the fact that the log-partition function $\log Z_\theta$ is intractable in general. As a result, most of the existing works estimate θ by using the sub-optimal approach of replacing the likelihood function by a pseudo-likelihood function. One notable exception that tackles the log-likelihood function is Höfling and Tibshirani (2009), using an active set strategy (to preserve sparsity), and the junction tree algorithm for computing the partial derivatives of the log-partition function. However, the success of this strategy depends crucially on the sparsity of the solution[1]. We will see that Algorithm 2 implemented with a MCMC

1. Indeed the implementation of their algorithm in the BMN package is very sensitive to the sparsity of the solution, and their solver typically fails to converge if the regularization parameter is not large enough to produce a sufficiently sparse solution. In our numerical experiments, we were not able to obtain a successful run from their package for $p = 100$.

approximation of the gradient gives a simple and effective approach for computing the penalized maximum likelihood estimate of θ.

Let $\mathcal{M}_p$ denote the space of $p \times p$ symmetric matrices equipped with the (modified) Frobenius inner product

$$\langle \theta, \vartheta \rangle \stackrel{\text{def}}{=} \sum_{1 \leq k \leq j \leq p} \theta_{jk} \vartheta_{jk}, \text{ with norm } \|\theta\| \stackrel{\text{def}}{=} \sqrt{\langle \theta, \theta \rangle}.$$

Equipped with this norm, $\mathcal{M}_p$ is the same space as the Euclidean space $\mathbb{R}^d$ where $d = p(p+1)/2$. Using a ℓ^1-penalty on θ, we see that the computation of the penalized maximum likelihood estimate of θ is a problem of the form (P) with $F = -\ell + g$ where

$$\ell(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left\langle \theta, \bar{B}(x^{(i)}) \right\rangle - \log Z_\theta \text{ and } \quad g(\theta) = \lambda \sum_{1 \leq k \leq j \leq p} |\theta_{jk}| \, ;$$

the matrix-valued function $\bar{B} : \mathsf{X}^p \to \mathbb{R}^{p \times p}$ is defined by

$$\bar{B}_{kk}(x) = B_0(x_k) \qquad \bar{B}_{kj}(x) = B(x_k, x_j) \, , k \neq j \, .$$

It is easy to see that in this example, Problem (P) admits at least one solution $\theta_\star$ that satisfies $\lambda \sum_{1 \leq k \leq j \leq p} |\theta_{jk}| \leq p \log |\mathsf{X}|$, where $|\mathsf{X}|$ denotes the size of X. To see this, note that since $f_\theta(x)$ is a probability, $-\ell(\theta) = -N^{-1} \sum_{i=1}^{N} \log f_\theta(x^{(i)}) \geq 0$. Hence $F(\theta) \geq g(\theta) \to \infty$, as $\sum_{1 \leq k \leq j \leq p} |\theta_{jk}| \to \infty$ and since F is continuous, we conclude that it admits at least one minimizer $\theta_\star$ that satisfies $F(\theta_\star) \leq F(\mathbf{0}) = \log Z_{\mathbf{0}} = p \log |\mathsf{X}|$. As a result, and without any loss of generality, we consider Problem (P) with the penalty g replaced by $g(\theta) = \lambda \sum_{1 \leq k \leq j \leq p} |\theta_{jk}| + \mathbb{1}(\theta)$, where $\mathbb{1}(\theta) = 0$ if $\max_{ij} |\theta_{ij}| \leq (p/\lambda) \log |\mathsf{X}|$, and $\mathbb{1}(\theta) = +\infty$ otherwise. Hence in this problem, the domain of g is $\Theta = \{\theta \in \mathcal{M}_p : \max_{ij} |\theta_{ij}| \leq (p/\lambda) \log |\mathsf{X}|\}$.

Upon noting that (14) is a canonical exponential model, (Shao, 2003, Section 4.4.2) shows that $\theta \mapsto -\ell(\theta)$ is convex and

$$\nabla \ell(\theta) = \frac{1}{N} \sum_{i=1}^{N} \bar{B}(x^{(i)}) - \int_{\mathsf{X}^p} \bar{B}(z) f_\theta(z) \mu(\mathrm{d}z) \, , \qquad (15)$$

where μ is the counting measure on X^p. In addition, (see section B)

$$\|\nabla \ell(\theta) - \nabla \ell(\vartheta)\| \leq p \left((p-1) \mathsf{osc}^2(B) + \mathsf{osc}^2(B_0) \right) \|\theta - \vartheta\|, \qquad (16)$$

where for a function $\tilde{B} : \mathsf{X} \times \mathsf{X} \to \mathbb{R}$, $\mathsf{osc}(\tilde{B}) = \sup_{x,y,u,v \in \mathsf{X}} |\tilde{B}(x,y) - \tilde{B}(u,v)|$.

Therefore, in this example, the assumption H1 and H2 are satisfied.

The representation of the gradient in (15) shows that H3 holds, with $\pi_\theta(\mathrm{d}z) = f_\theta(z)\mu(\mathrm{d}z)$, and $H_\theta(z) = N^{-1} \sum_{i=1}^{N} \bar{B}(x^{(i)}) - \bar{B}(z)$. Direct simulation from the distribution f_θ is rarely feasible, so we turn to MCMC. These Markov kernels are easy to construct, and can be constructed in many ways. For instance if the set X is not too large, then a Gibbs sampler (see e.g. Robert and Casella (2005)) that samples from the full conditional distributions of f_θ can be easily implemented. In the case of the

Gibbs sampler, since X^p is a finite set, Θ is compact, $f_\theta(x) > 0$ for all $(x, \theta) \in \mathsf{X}^p \times \Theta$, and, $\theta \mapsto f_\theta(x)$ is continuously differentiable, the assumptions H4, H5 and H6(i)-(ii) automatically hold with $W \equiv 1$. We should point out that the Gibbs sampler is a generic algorithm that in some cases is known to mix poorly. Whenever possible we recommend the use of specialized problem-specific MCMC algorithms with better mixing properties.

Illustrative example We consider the particular case where $\mathsf{X} = \{1, \dots, M\}$, $B_0(x) = 0$, and $B(x, y) = \mathbb{1}_{\{x=y\}}$, which corresponds to the well known Potts model. We report in this section some simulation results showing the performances of the stochastic proximal gradient algorithm. We use $M = 20$, $B_0(x) = x$, $N = 250$ and for $p \in \{50, 100, 200\}$. We generate the 'true' matrix θ_{true} such that it has on average p non-zero elements below the diagonal which are simulated from a uniform distribution on $(-4, -1) \cup (1, 4)$. All the diagonal elements are set to 0.

By trial-and-error we set the regularization parameter to $\lambda = 2.5\sqrt{\log(p)/n}$ for all the simulations. We implement Algorithm 2, drawing samples from a Gibbs sampler to approximate the gradient. We compare the following two versions of Algorithm 2:

1. Solver 1: A version with a fixed Monte Carlo batch size $m_n = 500$, and decreasing step size $\gamma_n = \frac{25}{p}\frac{1}{n^{0.7}}$.

2. Solver 2: A version with increasing Monte Carlo batch size $m_n = 500 + n^{1.2}$, and fixed step size $\gamma_n = \frac{25}{p}\frac{1}{\sqrt{50}}$.

We run Solver 2 for $\mathsf{Niter} = 5p$ iterations, where $p \in \{50, 100, 200\}$ is as above. And we set the number of iterations of Solver 1 so that both solvers draw approximately the same number of Monte Carlo samples. For stability in the results, we repeat the solvers 30 times and average the sample paths. We evaluate the convergence of each solver by computing the relative error $\|\theta_n - \theta_\infty\|/\|\theta_\infty\|$, along the iterations, where θ_∞ denotes the value returned by the solver on its last iteration. Note that we compare the optimizer output to θ_∞, not θ_{true}. Ideally, we would like to compare the iterates to the solution of the optimization problem. However in the present setting a solution is not available in closed form (and there could be more than one solution). Furthermore, whether the solution of the optimization problem approaches $\theta_\star$ is a complicated statistical problem[2] that is beyond the scope of this work. The relative errors are presented on Figure 1 and suggest that, when measured as function of resource used, Solver 1 and Solver 2 have roughly the same convergence rate.

We also compute the statistic $\mathsf{F}_n \stackrel{\text{def}}{=} \frac{2\mathsf{Sen}_n \mathsf{Prec}_n}{\mathsf{Sen}_n + \mathsf{Prec}_n}$ which measures the recovery of the sparsity structure of θ_∞ along the iteration. In this definition Sen_n is the sensitivity, and Prec_n is the precision defined as

$$\mathsf{Sen}_n = \frac{\sum_{j<i} \mathbb{1}_{\{|\theta_{n,ij}|>0\}} \mathbb{1}_{\{|\theta_{\infty,ij}|>0\}}}{\sum_{j<i} \mathbb{1}_{\{|\theta_{\infty,ij}|>0\}}}, \quad \text{and} \quad \mathsf{Prec}_n = \frac{\sum_{j<i} \mathbb{1}_{\{|\theta_{n,ij}|>0\}} \mathbb{1}_{\{|\theta_{\infty,ij}|>0\}}}{\sum_{j<i} \mathbb{1}_{\{|\theta_{n,ij}|>0\}}}.$$

2. this depends heavily on n, p, the actual true matrix θ_{true}, and depends also heavily the choice of the regularization parameter λ

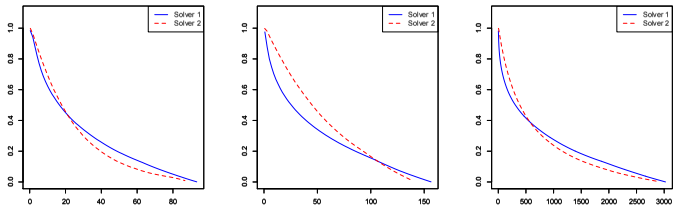


Figure 1: Relative errors plotted as function of computing time for Solver 1 and Solver 2.

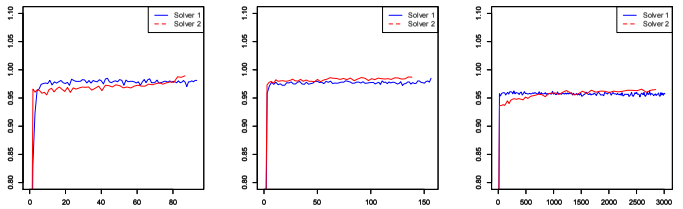


Figure 2: Statistic F_n plotted as function of computing time for Solver 1 and Solver 2.

The values of F_n are presented on Figure 2 as function of computing time. It shows that for both solvers, the sparsity structure of θ_n converges very quickly towards that of θ_∞. We note also that Figure 2 seems to suggest that Solver 2 tends to produce solutions with slightly more stable sparsity structure than Solver 1 (less variance on the red curves). Whether such subtle differences exist between the two algorithms (a diminishing step-size and fixed Monte Carlo size versus a fixed step-size and increasing Monte Carlo size) is an interest question. Our analysis does not deal with the sparsity structure of the solutions, hence cannot offer any explanation.

5. A non convex example: High-dimensional logistic regression with random effects

We numerically investigate the extension of our results to a situation where the assumptions H2 and H3 hold but H1 is not in general satisfied and the domain Θ is not bounded. The numerical study below shows that the conclusions reached in sec-

tion 2 and section 3 provide useful information to tune the design parameters of the algorithms.

5.1 The model

We model binary responses $\{Y_i\}_{i=1}^N \in \{0,1\}$ as N conditionally independent realizations of a random effect logistic regression model,

$$Y_i|\mathbf{U} \overset{ind.}{\sim} \mathsf{Ber}\left(s(x_i'\beta + \sigma z_i'\mathbf{U})\right), \quad 1 \leq i \leq N\,, \tag{17}$$

where $x_i \in \mathbb{R}^p$ is the vector of covariates, $z_i \in \mathbb{R}^q$ are (known) loading vector, $\mathsf{Ber}(\alpha)$ denotes the Bernoulli distribution with parameter $\alpha \in [0,1]$, $s(x) = \mathrm{e}^x/(1+\mathrm{e}^x)$ is the cumulative distribution function of the standard logistic distribution. The random effect $\mathbf{U}$ is assumed to be standard Gaussian $\mathbf{U} \sim \mathrm{N}_q(0, I)$.

The log-likelihood of the observations at $\theta = (\beta, \sigma) \in \mathbb{R}^p \times (0,\infty)$ is given by

$$\ell(\theta) = \log \int \prod_{i=1}^N s(x_i'\beta + \sigma z_i'\mathbf{u})^{Y_i} \left(1 - s(x_i'\beta + \sigma z_i'\mathbf{u})\right)^{1-Y_i} \phi(\mathbf{u})\mathrm{d}\mathbf{u}\,, \tag{18}$$

where ϕ is the density of a $\mathbb{R}^q$-valued standard Gaussian random vector. The number of covariates p is possibly larger than N, but only a very small number of these covariates are relevant which suggests to use the elastic-net penalty

$$\lambda\left(\frac{1-\alpha}{2}\|\beta\|_2^2 + \alpha\|\beta\|_1\right)\,, \tag{19}$$

where $\lambda > 0$ is the regularization parameter, $\|\beta\|_r = (\sum_{i=1}^p |\beta_i|^r)^{1/r}$ and $\alpha \in [0,1]$ controls the trade-off between the ℓ^1 and the ℓ^2 penalties. In this example,

$$g(\theta) = \lambda\left(\frac{1-\alpha}{2}\|\beta\|_2^2 + \alpha\|\beta\|_1\right) + \mathbb{1}_{(0,+\infty)}(\sigma)\,, \tag{20}$$

where $\mathbb{1}_A(x) = +\infty$ is $x \notin A$ and 0 otherwise. Define the conditional log-likelihood of $\mathbf{Y} = (Y_1, \ldots, Y_N)$ given $\mathbf{U}$ (the dependence upon $\mathbf{Y}$ is omitted) by

$$\ell_c(\theta|\mathbf{u}) = \sum_{i=1}^N \left\{Y_i\left(x_i'\beta + \sigma z_i'\mathbf{u}\right) - \ln\left(1 + \exp\left(x_i'\beta + \sigma z_i'\mathbf{u}\right)\right)\right\}\quad,$$

and the conditional distribution of the random effect $\mathbf{U}$ given the observations $\mathbf{Y}$ and the parameter θ

$$\pi_\theta(\mathbf{u}) = \exp\left(\ell_c(\theta|\mathbf{u}) - \ell(\theta)\right)\phi(\mathbf{u})\,. \tag{21}$$

The Fisher identity implies that the gradient of the log-likelihood (18) is given by

$$\nabla\ell(\theta) = \int \nabla_\theta \ell_c(\theta|\mathbf{u})\ \pi_\theta(\mathbf{u})\ \mathrm{d}\mathbf{u} = \int \left\{\sum_{i=1}^N (Y_i - s(x_i'\beta + \sigma z_i'\mathbf{u}))\begin{bmatrix} x_i \\ z_i'\mathbf{u}\end{bmatrix}\right\}\pi_\theta(\mathbf{u})\ \mathrm{d}\mathbf{u}\,.$$

The Hessian of the log-likelihood ℓ is given by (see e.g.(McLachlan and Krishnan, 2008, Chapter 3))

$$\nabla^2\ell(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta^2\ell_c(\theta|\mathbf{U})\right] + \mathrm{Cov}_{\pi_\theta}\left(\nabla_\theta\ell_c(\theta|\mathbf{U})\right)$$

where $\mathbb{E}_{\pi_\theta}$ and $\mathrm{Cov}_{\pi_\theta}$ denotes the expectation and the covariance with respect to the distribution π_θ, respectively. Since

$$\nabla_\theta^2\ell_c(\theta|\mathbf{u}) = -\sum_{i=1}^N s(x_i'\beta + \sigma z_i'\mathbf{u})\ \left(1 - s(x_i'\beta + \sigma z_i'\mathbf{u})\right)\begin{bmatrix} x_i \\ z_i'\mathbf{u}\end{bmatrix}\begin{bmatrix} x_i \\ z_i'\mathbf{u}\end{bmatrix}'\,,$$

and $\sup_{\theta\in\Theta}\int \|\mathbf{u}\|^2\pi_\theta(\mathbf{u})\,\mathrm{d}\mathbf{u} < \infty$ (see section A), $\nabla^2\ell(\theta)$ is bounded on Θ. Hence, $\nabla\ell(\theta)$ satisfies the Lipschitz condition showing that H1 is satisfied.

5.2 Numerical application

The assumption H3 is satisfied with π_θ given by (21) and

$$H_\theta(\mathbf{u}) = -\sum_{i=1}^N (Y_i - F(x_i'\beta + \sigma z_i'\mathbf{u}))\begin{bmatrix} x_i \\ z_i'\mathbf{u}\end{bmatrix}\,. \tag{22}$$

The distribution π_θ is sampled using the MCMC sampler proposed in Polson et al. (2013) based on data-augmentation. We write $-\nabla\ell(\theta) = \int_{\mathbb{R}^q\times\mathbb{R}^N} H_\theta(\mathbf{u})\tilde{\pi}_\theta(\mathbf{u},\mathbf{w})\,\mathrm{d}\mathbf{u}\mathrm{d}\mathbf{w}$ where $\tilde{\pi}_\theta(\mathbf{u},\mathbf{w})$ is defined for $\mathbf{u} \in \mathbb{R}^q$ and $\mathbf{w} = (w_1, \cdots, w_N) \in \mathbb{R}^N$ by

$$\tilde{\pi}_\theta(\mathbf{u},\mathbf{w}) = \left(\prod_{i=1}^N \bar{\pi}_{\mathsf{PG}}\left(w_i; x_i'\beta + \sigma z_i'\mathbf{u}\right)\right)\pi_\theta(\mathbf{u})\,;$$

in this expression, $\bar{\pi}_{\mathsf{PG}}(\cdot; c)$ is the density of the Polya-Gamma distribution on the positive real line with parameter c given by

$$\bar{\pi}_{\mathsf{PG}}(w; c) = \cosh(c/2)\ \exp\left(-wc^2/2\right)\ \rho(w)\mathbb{1}_{\mathbb{R}^+}(w)\,,$$

where $\rho(w) \propto \sum_{k\geq 0}(-1)^k(2k+1)\exp(-(2k+1)^2/(8w))w^{-3/2}$ (see (Biane et al., 2001, Section 3.1)). Thus, we have

$$\tilde{\pi}_\theta(\mathbf{u},\mathbf{w}) = C_\theta\phi(\mathbf{u})\prod_{i=1}^N \exp\left(\sigma(Y_i - 1/2)z_i'\mathbf{u} - w_i(x_i'\beta + \sigma z_i'\mathbf{u})^2/2\right)\ \rho(w_i)\mathbb{1}_{\mathbb{R}^+}(w_i)\,,$$

where $\ln C_\theta = -N\ln 2 - \ell(\theta) + \sum_{i=1}^N (Y_i - 1/2)x_i'\beta$. This target distribution can be sampled using a Gibbs algorithm: given the current value $(\mathbf{u}^t, \mathbf{w}^t)$ of the chain, the next point is obtained by sampling $\mathbf{u}^{t+1}$ under the conditional distribution of $\mathbf{u}$ given $\mathbf{w}^t$, and $\mathbf{w}^{t+1}$ under the conditional distribution of $\mathbf{w}$ given $\mathbf{u}^{t+1}$. In the present case, these conditional distributions are given respectively by

$$\tilde{\pi}_\theta(\mathbf{u}|\mathbf{w}) \equiv \mathrm{N}_q\left(\mu_\theta(\mathbf{w}); \Gamma_\theta(\mathbf{w})\right) \qquad \tilde{\pi}_\theta(\mathbf{w}|\mathbf{u}) = \prod_{i=1}^N \bar{\pi}_{\mathsf{PG}}(w_i; |x_i'\beta + \sigma z_i'\mathbf{u}|)$$

with

$$\Gamma_\theta(\mathbf{w}) = \left(I + \sigma^2 \sum_{i=1}^N w_i z_i z_i' \right)^{-1}, \qquad \mu_\theta(\mathbf{w}) = \sigma \Gamma_\theta(\mathbf{w}) \sum_{i=1}^N \left((Y_i - 1/2) - w_i x_i' \beta \right) z_i \,. \tag{23}$$

Exact samples of these conditional distributions can be obtained (see (Polson et al., 2013, Algorithm 1) for sampling under a Polya-Gamma distribution). It has been shown by Choi and Hobert (2013) that the Polya-Gamma Gibbs sampler is uniformly ergodic. Hence H5 is satisfied with $W \equiv 1$. Checking H6 is also straightforward.

We test the algorithms with $N = 500$, $p = 1,000$ and $q = 5$. We generate the $N \times p$ covariates matrix X columnwise, by sampling a stationary $\mathbb{R}^N$-valued autoregressive model with parameter $\rho = 0.8$ and Gaussian noise $\sqrt{1-\rho^2}\mathcal{N}_N(0, I)$. We generate the vector of regressors β_{true} from the uniform distribution on $[1, 5]$ and randomly set 98% of the coefficients to zero. The variance of the random effect is set to $\sigma^2 = 0.1$. We consider a repeated measurement setting so that $z_i = e_{\lceil iq/N \rceil}$ where $\{e_j, j \leq q\}$ is the canonical basis of $\mathbb{R}^q$ and $\lceil \cdot \rceil$ denotes the upper integer part. With such a simple expression for the random effect, we will be able to approximate the value $F(\theta)$ in order to illustrate the theoretical results obtained in this paper. We use the Lasso penalty ($\alpha = 1$ in (19)) with $\lambda = 30$.

We first illustrate the ability of Monte Carlo Proximal Gradient algorithms to find a minimizer of F. We compare the Monte Carlo proximal gradient algorithm

(i) with fixed batch size: $\gamma_n = 0.01/\sqrt{n}$ and $m_n = 275$ (Algo 1); $\gamma_n = 0.5/n$ and $m_n = 275$ (Algo 2).

(ii) with increasing batch size: $\gamma_n = \gamma = 0.005$, $m_n = 200 + n$ (Algo 3); $\gamma_n = \gamma = 0.001$, $m_n = 200 + n$ (Algo 4); and $\gamma_n = 0.05/\sqrt{n}$ and $m_n = 270 + \lceil \sqrt{n} \rceil$ (Algo 5).

Each algorithm is run for 150 iterations. The batch sizes $\{m_n, n \geq 0\}$ are chosen so that after 150 iterations, each algorithm used approximately the same number of Monte Carlo samples. We denote by β_∞ the value obtained at iteration 150. A path of the relative error $\|\beta_n - \beta_\infty\|/\|\beta_\infty\|$ is displayed on Figure 3[right] for each algorithm; a path of the sensitivity Sen_n and of the precision Prec_n (see section 4 for the definition) are displayed on Figure 4. All these sequences are plotted versus the total number of Monte Carlo samples up to iteration n. These plots show that with a fixed batch-size (Algo 1 or Algo 2), the best convergence is obtained with a step size decreasing as $O(1/\sqrt{n})$; and for an increasing batch size (Algo 3 to Algo 5), it is better to choose a fixed step size. These findings are consistent with the results in section 3. On Figure 3[left], we report on the bottom row the indices j such that $\beta_{\mathsf{true},j}$ is non null and on the rows above, the indices j such that $\beta_{\infty,j}$ given by Algo 1 to Algo 5 is non null.

We now study the convergence of $\{F(\theta_n), n \in \mathbb{N}\}$ where θ_n is obtained by one of the algorithms described above. We repeat 50 independent runs for each algorithm and estimate $\mathbb{E}[F(\theta_n)]$ by the empirical mean over these runs. On Figure 5[left], $n \mapsto F(\theta_n)$ is displayed for several runs of Algo 1 and Algo 3. The figure shows

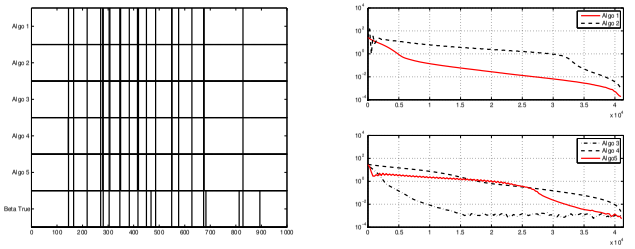


Figure 3: [left] The support of the sparse vector β_∞ obtained by Algo 1 to Algo 5; for comparison, the support of β_{true} is on the bottom row. [right] Relative error along one path of each algorithm as a function of the total number of Monte Carlo samples.

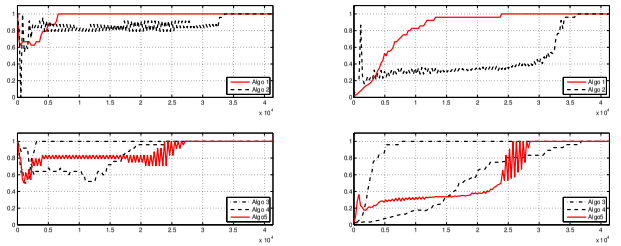


Figure 4: The sensitivity Sen_n [left] and the precision Prec_n [right] along a path, versus the total number of Monte Carlo samples up to time n

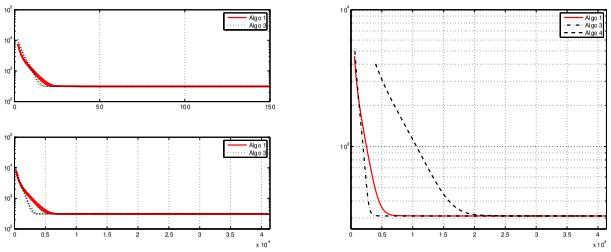


Figure 5: [left] $n \mapsto F(\theta_n)$ for several independent runs. [right] $\mathbb{E}\left[F(\theta_n)\right]$ versus the total number of Monte Carlo samples up to iteration n

that all the paths have the same limiting value, which is approximately $F_\star = 311$; we observed the same behavior on the 50 runs of each algorithm. On Figure 5[right], we report the Monte Carlo estimation of $\mathbb{E}[F(\theta_n)]$ versus the total number of Monte Carlo samples used up to iteration n for the best strategies in the fixed batch size case (Algo 1) and in the increasing batch size case (Algo 3 and Algo 4).

6. Proofs

6.1 Preliminary lemmas

Lemma 7 *Assume that g is lower semi-continuous and convex. For $\theta, \theta' \in \Theta$ and $\gamma > 0$*

$$g\Big(\operatorname{Prox}_{\gamma,g}(\theta)\Big) - g(\theta') \leq -\frac{1}{\gamma}\left\langle \operatorname{Prox}_{\gamma,g}(\theta) - \theta', \operatorname{Prox}_{\gamma,g}(\theta) - \theta \right\rangle . \tag{24}$$

For any $\gamma > 0$ and for any $\theta, \theta' \in \Theta$,

$$\|\operatorname{Prox}_{\gamma,g}(\theta) - \operatorname{Prox}_{\gamma,g}(\theta')\|^2 + \|\big(\operatorname{Prox}_{\gamma,g}(\theta) - \theta\big) - \big(\operatorname{Prox}_{\gamma,g}(\theta') - \theta'\big)\|^2 \leq \|\theta - \theta'\|^2 . \tag{25}$$

Proof See (Bauschke and Combettes, 2011, Propositions 4.2., 12.26 and 12.27). ■

Lemma 8 *Assume H1 and let $\gamma \in (0, 1/L]$. Then for all $\theta, \theta' \in \Theta$,*

$$-2\gamma\Big(F(\operatorname{Prox}_{\gamma,g}(\theta)) - F(\theta')\Big) \geq \|\operatorname{Prox}_{\gamma,g}(\theta) - \theta'\|^2 + 2\left\langle \operatorname{Prox}_{\gamma,g}(\theta) - \theta', \theta' - \gamma\nabla f(\theta') - \theta \right\rangle . \tag{26}$$

If in addition f is convex, then for all $\theta, \theta', \xi \in \Theta$,

$$-2\gamma\Big(F\big(\operatorname{Prox}_{\gamma,g}(\theta)\big) - F(\theta')\Big) \geq \|\operatorname{Prox}_{\gamma,g}(\theta) - \theta'\|^2 + 2\left\langle \operatorname{Prox}_{\gamma,g}(\theta) - \theta', \xi - \gamma\nabla f(\xi) - \theta \right\rangle - \|\theta' - \xi\|^2 . \tag{27}$$

Proof Since ∇f is Lipschitz, the descent lemma implies that for any $\gamma^{-1} \geq L$

$$f(p) - f(\theta') \leq \left\langle \nabla f(\theta'), p - \theta' \right\rangle + \frac{1}{2\gamma}\|p - \theta'\|^2 . \tag{28}$$

This inequality applied with $p = \operatorname{Prox}_{\gamma,g}(\theta)$ combined with (24) yields (26). When f is convex, $f(\xi) + \langle \nabla f(\xi), \theta' - \xi \rangle - f(\theta') \leq 0$ which, combined again with (24) and (28) applied with $(p, \theta') \leftarrow (\operatorname{Prox}_{\gamma,g}(\theta), \xi)$ yields the result. ■

Lemma 9 *Assume H1. Then for any $\gamma > 0$, $\theta, \theta' \in \Theta$,*

$$\|\theta - \gamma\nabla f(\theta) - \theta' + \gamma\nabla f(\theta')\| \leq (1 + \gamma L)\|\theta - \theta'\| , \tag{29}$$

$$\|T_\gamma(\theta) - T_\gamma(\theta')\| \leq (1 + \gamma L)\|\theta - \theta'\| . \tag{30}$$

If in addition f is convex then for any $\gamma \in (0, 2/L]$,

$$\|\theta - \gamma\nabla f(\theta) - \theta' + \gamma\nabla f(\theta')\| \leq \|\theta - \theta'\| , \tag{31}$$

$$\|T_\gamma(\theta) - T_\gamma(\theta')\| \leq \|\theta - \theta'\| . \tag{32}$$

Proof (30) and (32) follows from (29) and (31) respectively by the Lipschitz property of the proximal map $\operatorname{Prox}_{\gamma,g}$ (see Lemma 7). (29) follows directly from the Lipschitz property of f. It remains to prove (31). Since f is a convex function with Lipschitz-continuous gradients, (Nesterov, 2004, Theorem 2.1.5) shows that, for all $\theta, \theta' \in \Theta$, $L\left\langle \nabla f(\theta) - \nabla f(\theta'), \theta - \theta' \right\rangle \geq \|\nabla f(\theta) - \nabla f(\theta')\|^2$. The result follows. ■

Lemma 10 *Assume H1. Set $S_\gamma(\theta) \stackrel{\text{def}}{=} \operatorname{Prox}_{\gamma,g}(\theta - \gamma H)$ and $\eta \stackrel{\text{def}}{=} H - \nabla f(\theta)$. For any $\theta \in \Theta$ and $\gamma > 0$,*

$$\|T_\gamma(\theta) - S_\gamma(\theta)\| \leq \gamma\|\eta\| . \tag{33}$$

Proof We have $\|T_\gamma(\theta) - S_\gamma(\theta)\| = \|\operatorname{Prox}_{\gamma,g}(\theta - \gamma\nabla f(\theta)) - \operatorname{Prox}_{\gamma,g}(\theta - \gamma H)\|$ and (33) follows from Lemma 7. ■

6.2 Proof of section 2

6.2.1 Proof of Lemma 1

Set $w_n = v_n + \sum_{k \geq n+1} \xi_k + M$ with $M \stackrel{\text{def}}{=} -\inf_n \sum_{k \geq n} \xi_k$ so that $\inf_n w_n \geq 0$. Then

$$0 \leq w_{n+1} \leq v_n - \chi_{n+1} + \xi_{n+1} + \sum_{k \geq n+2} \xi_k + M \leq w_n - \chi_{n+1} .$$

$\{w_n,\ n \in \mathbb{N}\}$ is non-negative and non increasing; therefore it converges. Furthermore, $0 \leq \sum_{k=0}^{n} \chi_k \leq w_0$ so that $\sum_n \chi_n < \infty$. The convergence of $\{w_n,\ n \in \mathbb{N}\}$ also implies the convergence of $\{v_n,\ n \in \mathbb{N}\}$. This concludes the proof.

6.2.2 Proof of Theorem 2

Let $\theta_\star \in \mathcal{L}$, which is not empty by H2; note that $F(\theta_\star) = \min F$. We have by (27) applied with $\theta \leftarrow \theta_n - \gamma_{n+1} H_{n+1}$, $\xi \leftarrow \theta_n$, $\theta' \leftarrow \theta_\star$, $\gamma \leftarrow \gamma_{n+1}$

$$\|\theta_{n+1} - \theta_\star\|^2 \leq \|\theta_n - \theta_\star\|^2 - 2\gamma_{n+1}\left(F(\theta_{n+1}) - \min F\right) - 2\gamma_{n+1}\left\langle \theta_{n+1} - \theta_\star, \eta_{n+1}\right\rangle \ .$$

We write $\theta_{n+1} - \theta_\star = \theta_{n+1} - T_{\gamma_{n+1}}(\theta_n) + T_{\gamma_{n+1}}(\theta_n) - \theta_\star$. By Lemma 10, $\|\theta_{n+1} - T_{\gamma_{n+1}}(\theta_n)\| \leq \gamma_{n+1}\|\eta_{n+1}\|$ so that,

$$-\left\langle \theta_{n+1} - \theta_\star, \eta_{n+1}\right\rangle \leq \gamma_{n+1}\|\eta_{n+1}\|^2 - \left\langle T_{\gamma_{n+1}}(\theta_n) - \theta_\star, \eta_{n+1}\right\rangle \ .$$

Hence,

$$\begin{aligned}\|\theta_{n+1} - \theta_\star\|^2 \leq \|\theta_n - \theta_\star\|^2 - 2\gamma_{n+1}\left(F(\theta_{n+1}) - \min F\right) \\ + 2\gamma_{n+1}^2\|\eta_{n+1}\|^2 - 2\gamma_{n+1}\left\langle T_{\gamma_{n+1}}(\theta_n) - \theta_\star, \eta_{n+1}\right\rangle \ . \quad (34)\end{aligned}$$

Under (7) and (34), Lemma 1 shows that $\sum_n \gamma_n \left(F(\theta_n) - \min F\right) < \infty$ and $\lim_n \|\theta_n - \theta_\star\|$ exists. This implies that $\sup_n \|\theta_n\| < \infty$. Since $\sum_n \gamma_n = +\infty$, there exists a subsequence $\{\theta_{\phi_n}, n \in \mathbb{N}\}$ such that $\lim_n F(\theta_{\phi_n}) = \min F$. The sequence $\{\theta_{\phi_n}, n \geq 0\}$ being bounded, we can assume without loss of generality that there exists $\theta_\infty \in \mathbb{R}^d$ such that $\lim_n \theta_{\phi_n} = \theta_\infty$.

Let us prove that $\theta_\infty \in \mathcal{L}$. Since g is lower semi-continuous on Θ, $\liminf_n g(\theta_{\phi_n}) \geq g(\theta_\infty)$ so that $\theta_\infty \in \Theta$. Since F is lower semi-continuous on Θ, we have

$$\min F = \liminf_{n\to\infty} F(\theta_{\phi_n}) \geq F(\theta_\infty) \geq \min F \ ,$$

showing that $F(\theta_\infty) = \min F$.

By (34), for any m and $n \geq \phi_m$

$$\|\theta_{n+1} - \theta_\infty\|^2 \leq \|\theta_{\phi_m} - \theta_\infty\|^2 - 2\sum_{k=\phi_m}^{n} \gamma_{k+1}\{\left\langle T_{\gamma_{k+1}}(\theta_k) - \theta_\infty, \eta_{k+1}\right\rangle + \gamma_{k+1}\|\eta_{k+1}\|^2\} \ .$$

For any $\epsilon > 0$, there exists m such that the RHS is upper bounded by ϵ. Hence, for any $n \geq \phi_m$, $\|\theta_{n+1} - \theta_\infty\|^2 \leq \epsilon$, which proves the convergence of $\{\theta_n,\ n \in \mathbb{N}\}$ to θ_∞.

6.2.3 Proof of Theorem 3

Let $\theta_\star \in \mathcal{L}$; note that $F(\theta_\star) = \min F$. We first apply (27) with $\theta \leftarrow \theta_j - \gamma_{j+1} H_{j+1}$, $\xi \leftarrow \theta_j$, $\theta' \leftarrow \theta_\star$, $\gamma \leftarrow \gamma_{j+1}$:

$$F(\theta_{j+1}) - \min F \leq (2\gamma_{j+1})^{-1}\left(\|\theta_j - \theta_\star\|^2 - \|\theta_{j+1} - \theta_\star\|^2\right) - \left\langle \theta_{j+1} - \theta_\star, \eta_{j+1}\right\rangle \ .$$

Multiplying both sides by a_{j+1} gives:

$$\begin{aligned}a_{j+1}\Big(F(\theta_{j+1}) - \min F\Big) \leq \frac{1}{2}\left(\frac{a_{j+1}}{\gamma_{j+1}} - \frac{a_j}{\gamma_j}\right)\|\theta_j - \theta_\star\|^2 + \frac{a_j}{2\gamma_j}\|\theta_j - \theta_\star\|^2 \\ - \frac{a_{j+1}}{2\gamma_{j+1}}\|\theta_{j+1} - \theta_\star\|^2 - a_{j+1}\left\langle \theta_{j+1} - \theta_\star, \eta_{j+1}\right\rangle .\end{aligned}$$

Summing from $j = 0$ to $n-1$ gives

$$\begin{aligned}\frac{a_n}{2\gamma_n}\|\theta_n - \theta_\star\|^2 + \sum_{j=1}^{n} a_j\{F(\theta_j) - \min F\} \leq \frac{1}{2}\sum_{j=1}^{n}\left(\frac{a_j}{\gamma_j} - \frac{a_{j-1}}{\gamma_{j-1}}\right)\|\theta_{j-1} - \theta_\star\|^2 \\ - \sum_{j=1}^{n} a_j\left\langle \theta_j - \theta_\star, \eta_j\right\rangle + \frac{a_0}{2\gamma_0}\|\theta_0 - \theta_\star\|^2 \ . \quad (35)\end{aligned}$$

We decompose $\left\langle \theta_j - \theta_\star, \eta_j\right\rangle$ as follows:

$$\left\langle \theta_j - \theta_\star, \eta_j\right\rangle = \left\langle \theta_j - T_{\gamma_j}(\theta_{j-1}), \eta_j\right\rangle + \left\langle T_{\gamma_j}(\theta_{j-1}) - \theta_\star, \eta_j\right\rangle \ .$$

By Lemma 10, we get $\left|\left\langle \theta_j - T_{\gamma_j}(\theta_{j-1}), \eta_j\right\rangle\right| \leq \gamma_j\|\eta_j\|^2$ which concludes the proof.

6.3 Proof of Section 3.1

The proof of Theorem 4 is given in the case $m = 1$; we simply denote by X_n the sample $X_n^{(1)}$. The proof for the case $m > 1$ can be adapted from the proof below, by substituting the functions $H_\theta(x)$ and $W(x)$ by

$$\overline{H}_\theta(x_1, \cdots, x_m) = \frac{1}{m}\sum_{k=1}^{m} H_\theta(x_k) \qquad \overline{W}(x_1, \cdots, x_m) = \frac{1}{m}\sum_{k=1}^{m} W(x_k) \ ;$$

the kernel P_θ and its invariant measure π_θ by

$$\begin{aligned}\overline{P}_\theta(x_1, \cdots, x_m; B) &= \int \cdots \int P_\theta(x_m, \mathrm{d}y_1)\prod_{k=2}^{m} P_\theta(y_{k-1}, \mathrm{d}y_k)\mathbb{1}_B(y_1, \ldots, y_m) \ , \\ \overline{\pi}_\theta(B) &= \int \cdots \int \pi_\theta(\mathrm{d}y_1)\prod_{k=2}^{m} P_\theta(y_{k-1}, \mathrm{d}y_k)\mathbb{1}_B(y_1, \ldots, y_m) \ ,\end{aligned}$$

for any $(x_1, \ldots, x_m) \in \mathsf{X}^n$ and $B \in \mathcal{X}^{\times n}$.

6.3.1 Preliminary results

Proposition 11 *Assume that g is proper convex and Lipschitz on Θ with Lipschitz constant K. Then, for all $\theta \in \Theta$,*

$$\|\mathrm{Prox}_{\gamma,g}(\theta) - \theta\| \leq K\gamma \ . \quad (36)$$

Proof For all $\theta \in \Theta$, we get by Lemma 7

$$0 \leq \gamma^{-1}\left\|\theta - \mathrm{Prox}_{\gamma,g}(\theta)\right\|^2 \leq g(\theta) - g(\mathrm{Prox}_{\gamma,g}(\theta)) \leq K\left\|\theta - \mathrm{Prox}_{\gamma,g}(\theta)\right\| \ .$$

■

Proposition 12 *Assume H1, H2 and Θ is bounded. Then*

$$\sup_{\gamma\in(0,1/L]}\sup_{\theta\in\Theta}\|T_\gamma(\theta)\|<\infty\ .$$

If in addition H6-(ii) holds, then there exists a constant C such that for any $\theta,\bar\theta\in\Theta$, $\gamma,\bar\gamma\in(0,1/L]$

$$\left\|T_\gamma(\theta)-T_{\bar\gamma}(\bar\theta)\right\|\le C\left(\gamma+\bar\gamma+\|\theta-\bar\theta\|\right)\ .$$

Proof Let $\theta_\star$ such that for any $\gamma>0$, $\theta_\star=T_\gamma(\theta_\star)$ (such a point exists by H2 and (4)). We write $T_\gamma(\theta)=(T_\gamma(\theta)-\theta_\star)+\theta_\star$. By Lemma 9, there exists a constant C such that for any $\theta\in\Theta$ and any $\gamma\in(0,1/L]$, $\|T_\gamma(\theta)-\theta_\star\|\le 2\,\|\theta-\theta_\star\|\le 2\,\|\theta\|+2\,\|\theta_\star\|$. This concludes the proof of the first statement. We write $T_\gamma(\theta)-T_{\bar\gamma}(\bar\theta)=T_\gamma(\theta)-T_{\bar\gamma}(\theta)+T_{\bar\gamma}(\theta)-T_{\bar\gamma}(\bar\theta)$. By Lemma 7

$$\left\|T_{\bar\gamma}(\theta)-T_{\bar\gamma}(\bar\theta)\right\|\le\left\|\theta-\bar\theta-\bar\gamma\nabla f(\theta)+\bar\gamma\nabla f(\bar\theta)\right\|\le\|\theta-\bar\theta\|+\bar\gamma\sup_{\theta\in\Theta}\|\nabla f(\theta)\|\ .$$

By H1 and since Θ is bounded, $\sup_{\theta\in\Theta}\|\nabla f(\theta)\|<\infty$. In addition, using again Lemma 7,

$$\|T_\gamma(\theta)-T_{\bar\gamma}(\theta)\|\le(\gamma+\bar\gamma)\ \sup_{\theta\in\Theta}\|\nabla f(\theta)\|+\|\mathrm{Prox}_{\gamma,g}(\theta)-\mathrm{Prox}_{\bar\gamma,g}(\theta)\|\ .$$

We conclude by using

$$\begin{aligned}\|\mathrm{Prox}_{\bar\gamma,g}(\theta)-\mathrm{Prox}_{\gamma,g}(\theta)\|&\le\|\mathrm{Prox}_{\bar\gamma,g}(\theta)-\theta\|+\|\theta-\mathrm{Prox}_{\gamma,g}(\theta)\|\\&\le(\gamma+\bar\gamma)\ \sup_{\gamma\in(0,1/L]}\sup_{\theta\in\Theta}\gamma^{-1}\ \|\mathrm{Prox}_{\gamma,g}(\theta)-\theta\|\ .\end{aligned}$$

■

Lemma 13 *Assume H5 and H6-(i).*

(i) There exists a measurable function $(\theta,x)\mapsto\widehat H_\theta(x)$ such that $\sup_{\theta\in\Theta}\left|\widehat H_\theta\right|_W<\infty$ and for any $(\theta,x)\in\Theta\times\mathsf X$,

$$\widehat H_\theta(x)-P_\theta\widehat H_\theta(x)=H_\theta(x)-\int H_\theta(y)\pi_\theta(\mathrm dy)\ . \tag{37}$$

(ii) There exists a constant C such that for any $\theta,\theta'\in\Theta$,

$$\left\|P_\theta\widehat H_\theta-P_{\theta'}\widehat H_{\theta'}\right\|_W\le C\ \ \|\theta-\theta'\|\ .$$

Proof See (Fort et al., 2011, Lemma 4.2). ■

Lemma 14 *Assume H4 and H5. Then, $\sup_n\mathbb E\left[W^p(X_n)\right]<\infty$.*

Proof The conditional distribution of X_j given the past $\mathcal F_{j-1}$ is $P_{\theta_{j-1}}(X_{j-1},\cdot)$. Therefore, we write

$$\mathbb E\left[W^p(X_n)\right]=\mathbb E\left[\mathbb E\left[W^p(X_n)\,|\,\mathcal F_{n-1}\right]\right]=\mathbb E\left[P_{\theta_{n-1}}W^p(X_{n-1})\right].$$

We then use the drift inequality to obtain $\mathbb E\left[W^p(X_n)\right]\le\lambda\mathbb E\left[W^p(X_{n-1})\right]+b$. The proof then follows from a trivial induction. ■

Lemma 15 *Assume H1, H6-(ii) and Θ is bounded. There exists a constant C such that w.p.1, for all $n\ge0$,*

$$\|\theta_{n+1}-\theta_n\|\le C\gamma_{n+1}\left(1+\|\eta_{n+1}\|\right)\ .$$

Proof We write

$$\theta_{n+1}-\theta_n=\theta_{n+1}-\mathrm{Prox}_{\gamma_{n+1},g}(\theta_n)+\mathrm{Prox}_{\gamma_{n+1},g}(\theta_n)-\theta_n.$$

Since by Lemma 7, $\theta\mapsto\mathrm{Prox}_{\gamma,g}(\theta)$ is Lipschitz for any $\gamma>0$, we get

$$\begin{aligned}&\left\|\theta_{n+1}-\mathrm{Prox}_{\gamma_{n+1},g}(\theta_n)\right\|\\&=\left\|\mathrm{Prox}_{\gamma_{n+1},g}(\theta_n-\gamma_{n+1}\eta_{n+1}-\gamma_{n+1}\nabla f(\theta_n))-\mathrm{Prox}_{\gamma_{n+1},g}\left(\theta_n\right)\right\|\\&\le\gamma_{n+1}\left\|\eta_{n+1}+\nabla f(\theta_n)\right\|\le\gamma_{n+1}\left(\|\eta_{n+1}\|+\sup_{\theta\in\Theta}\|\nabla f(\theta)\|\right)\ .\end{aligned}$$

By H1, w.p.1. $\sup_{\theta\in\Theta}\|\nabla f(\theta)\|<\infty$; hence, there exists C_1 such that w.p.1. for all $n\ge0$, $\left\|\theta_{n+1}-\mathrm{Prox}_{\gamma_{n+1},g}(\theta_n)\right\|\le C_1\gamma_{n+1}\left(1+\|\eta_{n+1}\|\right)$. Finally, under H6-(ii), there exists a constant C_2 such that, w.p.1.,

$$\sup_n\gamma_{n+1}^{-1}\left\|\mathrm{Prox}_{\gamma_{n+1},g}(\theta_n)-\theta_n\right\|\le\sup_{\gamma\in(0,1/L]}\sup_{\theta\in\Theta}\gamma^{-1}\left\|\mathrm{Prox}_{\gamma,g}(\theta)-\theta\right\|\le C_2\ .$$

This concludes the proof. ■

Lemma 16 *Assume H1, H4, H5 and Θ is bounded. There exists a constant C such that w.p.1, for all $n\ge0$, $\|\eta_{n+1}\|\le CW(X_{n+1})$.*

Proof By H4 and H5, $\|\eta_{n+1}\|\le\left(\sup_{\theta\in\Theta}|H_\theta|_W\right)W(X_{n+1})+\sup_{\theta\in\Theta}\|\nabla f(\theta)\|$. The result follows since ∇f is Lipschitz by H1, and since $W\ge1$. ■

6.3.2 Proof of Theorem 4

The proof of the almost-sure convergence consists in verifying the assumptions of Theorem 2. Let us start with the proof that almost-surely, $\sum_n \gamma_{n+1}^2 \|\eta_{n+1}\|^2 < \infty$. This property is a consequence of Lemma 17 applied with $a_n \leftarrow \gamma_n^2$. It remains to prove that almost-surely

$$\sum_n \gamma_n \eta_n < \infty, \qquad \sum_n \gamma_{n+1} \left\langle T_{\gamma_{n+1}}(\theta_n), \eta_{n+1} \right\rangle < \infty;$$

note that they are both of the form $\sum_n \gamma_{n+1} \mathsf{A}_{\gamma_{n+1}}(\theta_n)\eta_{n+1}$ with, respectively, $\mathsf{A}_\gamma(\theta)$ equal to the identity matrix, and $\mathsf{A}_\gamma(\theta) = T_\gamma(\theta)$. In the case the Monte Carlo is unbiased, we apply Proposition 18 with $a_n \leftarrow \gamma_n$ and $A_\gamma(\theta)$ equal to the identity matrix and we obtain the almost-sure convergence of $\sum_n \gamma_n \eta_n$; we then apply Proposition 18 with $a_n \leftarrow \gamma_n$ and $A_\gamma(\theta) = T_\gamma(\theta)$, and we obtain the almost-sure convergence of $\sum_n \gamma_{n+1} \left\langle T_{\gamma_{n+1}}(\theta_n), \eta_{n+1} \right\rangle$ - note that by Proposition 12, $T_\gamma(\theta)$ satisfies the assumptions on $\mathsf{A}_\gamma(\theta)$. In the case the Monte Carlo is biased, the steps are the same except we use Proposition 19 instead of Proposition 18.

For the control of the moments, we use Theorem 3 and again Lemma 17 and Proposition 18 for the unbiased case (or Proposition 19 for the biased case).

Lemma 17 *Assume H1, H4, H5 and Θ is bounded.*

(i) If $a_k \geq 0$ and $\sum_{k=1}^\infty a_k < \infty$ then with probability one, $\sum_{n\geq 1} a_n \|\eta_n\|^2 < \infty$.

(ii) for any $q \in [1, p/2]$, there exists a constant C such that for any non-negative numbers $\{a_1, \cdots, a_n\}$,

$$\left\| \sum_{k=1}^n a_k \|\eta_k\|^2 \right\|_{L^q} \leq C \sum_{k=1}^n a_k \ .$$

Proof We write

$$\mathbb{E}\left[\sum_{n\geq 0} a_{n+1} \|\eta_{n+1}\|^2 \right] \leq \sup_n \left(\mathbb{E}\left[\|\eta_{n+1}\|^2 \right] \right) \sum_{n\geq 0} a_{n+1} \ .$$

By Lemma 14 and Lemma 16, $\sup_n \|\eta_{n+1}\|_{L^2} < \infty$ so the RHS is finite. By the Minkovski inequality, we write since $a_k > 0$,

$$\left\| \sum_{k=0}^n a_{k+1} \|\eta_{k+1}\|^2 \right\|_{L^q} \leq \sup_n \|\eta_n\|_{L^{2q}}^2 \sum_{k=1}^{n+1} a_k \ .$$

The supremum is finite by Lemma 14 and Lemma 16. ∎

Proposition 18 *Assume H1, H3, H4, H5, Θ is bounded and the Monte Carlo approximation is unbiased. Let $\{a_n,\ n \in \mathbb{N}\}$ be a deterministic positive sequence and $\{\mathsf{A}_\gamma(\theta), \gamma \in (0, 1/L], \theta \in \Theta\}$ be deterministic matrices such that*

$$\sup_{\gamma \in (0,1/L]} \sup_{\theta \in \Theta} \|\mathsf{A}_\gamma(\theta)\| < \infty \ . \tag{38}$$

(i) If $\sum_{n\geq 0} a_n^2 < \infty$, then the series $\sum_{n\geq 0} a_{n+1} \mathsf{A}_{\gamma_{n+1}}(\theta_n)\eta_{n+1}$ converges $\mathbb{P}$-a.s.

(ii) For any $q \in (1, p/2]$, there exists a constant C such that

$$\left\| \sum_{k=0}^n a_{k+1} \mathsf{A}_{\gamma_{k+1}}(\theta_k)\eta_{k+1} \right\|_{L^q} \leq C \ \left(\sum_{k=0}^n a_{k+1}^2 \right)^{1/2} \ .$$

Proof Since $\theta_n \in \mathcal{F}_n$, we have $\mathbb{E}\left[a_{n+1} \mathsf{A}_{\gamma_{n+1}}(\theta_n)\ \eta_{n+1} | \mathcal{F}_n \right] = 0$, thus showing that $\{M_n = \sum_{k=0}^n a_{k+1} \mathsf{A}_{\gamma_{k+1}}(\theta_k)\eta_{k+1}, n \in \mathbb{N}\}$ is a martingale. This martingale converges almost-surely if $S = \sum_{n\geq 0} a_{n+1}^2 \|\mathsf{A}_{\gamma_{n+1}}(\theta_n)\|^2 \|\eta_{n+1}\|^2 < \infty$ $\mathbb{P}$-a.s. (see e.g. (Hall and Heyde, 1980, Theorem 2.17)). Using (38) and Lemma 17, $S < \infty$ $\mathbb{P}$-a.s.

Consider now the L^q-moment of M_n. We apply (Hall and Heyde, 1980, Theorem 2.10): for any $q \in (1, p/2]$, there exists a constant C such that for any $n \geq 0$,

$$\left\| \sum_{k=0}^n a_{k+1} \mathsf{A}_{\gamma_{k+1}}(\theta_k)\eta_{k+1} \right\|_{L^q} \leq C \left(\sum_{k=0}^n \left\| a_{k+1} \mathsf{A}_{\gamma_{k+1}}(\theta_k)\eta_{k+1} \right\|_{L^q}^2 \right)^{1/2} \ .$$

Lemma 14 and Lemma 16 imply that $\sup_n \|\eta_{n+1}\|_{L^q} < \infty$; we then conclude with (38). ∎

Proposition 19 *Assume H1, H3–H6 and Θ is bounded. Let $\{a_n, n \geq 0\}$ be a positive sequence and $\{\mathsf{A}_\gamma(\theta), \gamma \in (0, 1/L], \theta \in \Theta\}$ be (deterministic) function-valued matrices such that there exists C_A and for any $\gamma, \bar{\gamma} \in (0, 1/L]$ and $\theta, \bar{\theta} \in \Theta$*

$$\sup_{\gamma \in (0,1/L]} \sup_{\theta \in \Theta} \|\mathsf{A}_\gamma(\theta)\| < \infty \ , \qquad \left\| \mathsf{A}_\gamma(\theta) - \mathsf{A}_{\bar{\gamma}}(\bar{\theta}) \right\| \leq C_\mathsf{A} \left(\gamma + \bar{\gamma} + \left\| \theta - \bar{\theta} \right\| \right) \ . \tag{39}$$

(i) If $\sum_n a_n \gamma_n < \infty$, $\sum_n a_n^2 < \infty$ and $\sum_n |a_{n+1} - a_n| < \infty$ then the series $\sum_{n\geq 0} a_{n+1} \mathsf{A}_{\gamma_{n+1}}(\theta_n)\eta_{n+1}$ converges $\mathbb{P}$-a.s.

(ii) For any $q \in (1, p/2]$, there exists a constant C such that

$$\left\| \sum_{k=0}^n a_{k+1} \mathsf{A}_{\gamma_{k+1}}(\theta_k)\eta_{k+1} \right\|_{L^q} \leq C \ \left\{ 1 + \left(\sum_{k=0}^n a_{k+1}^2 \right)^{1/2} + \sum_{k=1}^n |a_{k+1} - a_k| + \sum_{k=1}^n a_k \gamma_k \right\} \ .$$

Proof

(i) By H4 and Lemma 13-(i), we write

$$\begin{aligned} \eta_{n+1} &= \widehat{H}_{\theta_n}(X_{n+1}) - P_{\theta_n} \widehat{H}_{\theta_n}(X_{n+1}) \\ &= \left(\widehat{H}_{\theta_n}(X_{n+1}) - P_{\theta_n} \widehat{H}_{\theta_n}(X_n) \right) + \left(P_{\theta_n} \widehat{H}_{\theta_n}(X_n) - P_{\theta_{n+1}} \widehat{H}_{\theta_{n+1}}(X_{n+1}) \right) \\ &+ \left(P_{\theta_{n+1}} \widehat{H}_{\theta_{n+1}}(X_{n+1}) - P_{\theta_n} \widehat{H}_{\theta_n}(X_{n+1}) \right) \ . \end{aligned}$$

We prove successively that w.p.1,

$$\sum_n a_{n+1}\mathsf{A}_{\gamma_{n+1}}(\theta_n)\ \left(\widehat{H}_{\theta_n}(X_{n+1}) - P_{\theta_n}\widehat{H}_{\theta_n}(X_n)\right) < \infty\ , \tag{40}$$

$$\sum_{n\geq 0} a_{n+1}\mathsf{A}_{\gamma_{n+1}}(\theta_n)\ \left(P_{\theta_n}\widehat{H}_{\theta_n}(X_n) - P_{\theta_{n+1}}\widehat{H}_{\theta_{n+1}}(X_{n+1})\right) < \infty\ , \tag{41}$$

$$\sum_{n\geq 0} a_{n+1}\mathsf{A}_{\gamma_{n+1}}(\theta_n)\left(P_{\theta_{n+1}}\widehat{H}_{\theta_{n+1}}(X_{n+1}) - P_{\theta_n}\widehat{H}_{\theta_n}(X_{n+1})\right) < \infty\ . \tag{42}$$

Proof [Proof of (40)] By H4, $\{\widehat{H}_{\theta_n}(X_{n+1}) - P_{\theta_n}\widehat{H}_{\theta_n}(X_n), n\in\mathbb{N}\}$ is a martingale increment w.r.t. the filtration $\{\mathcal{F}_n, n\geq 0\}$. The proof is along the same lines as the proof of Proposition 18 upon noting that by Lemma 13 and H5, there exists C such that w.p.1 for all $n\geq 0$,

$$\|\widehat{H}_{\theta_n}(X_{n+1}) - P_{\theta_n}\widehat{H}_{\theta_n}(X_n)\| \leq C\ \{W(X_{n+1}) + W(X_n)\}\ .$$

■

Proof [Proof of (41)] The sum is equal to $\sum_{n\geq 0}\Delta_{n+1}P_{\theta_n}\widehat{H}_{\theta_n}(X_n)$ with $\Delta_{n+1} = a_{n+1}\mathsf{A}_{\gamma_{n+1}}(\theta_n) - a_n\mathsf{A}_{\gamma_n}(\theta_{n-1})$. On one hand, by Lemma 13 and H5, there exists C such that w.p.1 for all $n\geq 0$,

$$\left\|P_{\theta_n}\widehat{H}_{\theta_n}(X_n)\right\| \leq C\,W(X_n)\ .$$

On the other hand, by (39), Lemma 15 and Lemma 16, there exists C such that a.s.

$$\text{for all } n\geq 0, \qquad \|\Delta_{n+1}\| \leq C\Big(\,|a_{n+1}-a_n| + a_n\,(\gamma_n+\gamma_{n+1})\,\Big)W(X_n)\ .$$

By Lemma 14, $\sup_n \mathbb{E}\left[W^2(X_n)\right] < \infty$. Therefore, by (39) and the assumptions on $\{a_n, n\geq 0\}$, we have $\sum_n \mathbb{E}\left[\left\|\Delta_{n+1}\ P_{\theta_n}\widehat{H}_{\theta_n}(X_n)\right\|\right] < \infty$; which concludes the proof. ■

Proof [Proof of (42)] By (39) and Lemma 13, there exists a constant C such that w.p.1 for any n

$$\left\|\mathsf{A}_{\gamma_{n+1}}(\theta_n)\left(P_{\theta_{n+1}}\widehat{H}_{\theta_{n+1}}(X_{n+1}) - P_{\theta_n}\widehat{H}_{\theta_n}(X_{n+1})\right)\right\| \leq C\ \|\theta_{n+1}-\theta_n\|\ W(X_{n+1})\ .$$

By Lemma 15 and Lemma 16, there exists a constant C such that w.p.1,

$$\text{for all } n\geq 0, \qquad \|\theta_{n+1}-\theta_n\|\,W(X_{n+1}) \leq C\gamma_{n+1}\,W^2(X_{n+1})\ .$$

From Lemma 14 and the assumptions on $\{a_n, n\geq 0\}$, $\sum_n a_{n+1}\,\gamma_{n+1}\mathbb{E}\left[W^2(X_{n+1})\right] < \infty$ from which (42) follows. ■

(ii) We start from the same decomposition of η_{n+1} in three terms. The first one is a martingale, and following the same lines as in the proof of Proposition 18, we obtain

$$\left\|\sum_{k=0}^n a_{k+1}\mathsf{A}_{\gamma_{n+1}}(\theta_n)\ \left(\widehat{H}_{\theta_n}(X_{n+1}) - P_{\theta_n}\widehat{H}_{\theta_n}(X_n)\right)\right\|_{L^q} \leq C\left(\sum_{k=0}^n a_{k+1}^2\right)^{1/2}\ .$$

For the second term, we write

$$\begin{aligned}\sum_{k=0}^n a_{k+1}\mathsf{A}_{\gamma_{k+1}}(\theta_k)\ &\left(P_{\theta_k}\widehat{H}_{\theta_k}(X_k) - P_{\theta_{k+1}}\widehat{H}_{\theta_{k+1}}(X_{k+1})\right)\\ &\leq a_1\mathsf{A}_{\gamma_1}(\theta_0)P_{\theta_0}\widehat{H}_{\theta_0}(X_0) - a_{n+1}\mathsf{A}_{\gamma_{n+1}}(\theta_n)P_{\theta_{n+1}}\widehat{H}_{\theta_{n+1}}(X_{n+1})\\ &+\sum_{k=1}^n \Delta_{k+1}\ P_{\theta_k}\widehat{H}_{\theta_k}(X_k)\ .\end{aligned}$$

By the Minkovski inequality, it is easily seen that there exists a constant C such that

$$\begin{aligned}&\left\|\sum_{k=0}^n a_{k+1}\mathsf{A}_{\gamma_{k+1}}(\theta_k)\ \left(P_{\theta_k}\widehat{H}_{\theta_k}(X_k) - P_{\theta_{k+1}}\widehat{H}_{\theta_{k+1}}(X_{k+1})\right)\right\|_{L^q}\\ &\qquad\leq\left(1 + a_{n+1} + \sum_{k=1}^n\Big(\,|a_{k+1}-a_k| + a_k\,(\gamma_k+\gamma_{k+1})\,\Big)\right)\ .\end{aligned}$$

Finally, for the last term, following the same computations as above, we have by the Minkovski inequality

$$\left\|\sum_{k=0}^n a_{k+1}\mathsf{A}_{\gamma_{k+1}}(\theta_k)\ \left(P_{\theta_{k+1}}\widehat{H}_{\theta_{k+1}}(X_{k+1}) - P_{\theta_k}\widehat{H}_{\theta_k}(X_{k+1})\right)\right\|_{L^q} \leq C\sum_{k=0}^n a_{k+1}\gamma_{k+1}\ .$$

■

6.4 Proof of Theorem 6

We write $\eta_{n+1} = B_n + (\eta_{n+1} - B_n)$ where B_n is given by (12). Observe that $\{\eta_{n+1} - B_n, n\in\mathbb{N}\}$ is a martingale-increment sequence. Sufficient conditions for the almost-sure convergence of a martingale and the control of L^q-moments can be found in (Hall and Heyde, 1980, Theorems 2.10 and 2.17). Then the proof follows from Proposition 5 and Lemma 14.

Appendix A. Proofs of section 4

By using the Cauchy-Schwartz inequality, it holds

$$\int \exp(\ell_c(\theta|\mathbf{u}))\,\phi(\mathbf{u})\mathrm{d}\mathbf{u} \geq \left(\int \exp(0.5\ell_c(\theta|\mathbf{u}))\,\phi(\mathbf{u})\mathrm{d}\mathbf{u}\right)^{1/2}$$

$$\left(\int \exp(\ell_c(\theta|\mathbf{u}))\,\|u\|^2\,\phi(\mathbf{u})\,\mathrm{d}\mathbf{u}\right)^2 \leq \left(\int \exp(0.5\ell_c(\theta|\mathbf{u}))\phi(\mathbf{u})\,\mathrm{d}\mathbf{u}\right)\,\left(\int \exp(3\ell_c(\theta|\mathbf{u})/2)\|u\|^4\phi(\mathbf{u})\mathrm{d}\mathbf{u}\right)$$

which implies that

$$\int \|u\|^2\pi_\theta(\mathbf{u})\mathrm{d}\mathbf{u} = \frac{\int \exp(\ell_c(\theta|\mathbf{u}))\|u\|^2\phi(\mathbf{u})\mathrm{d}\mathbf{u}}{\int \exp(\ell_c(\theta|v))\phi(v)\mathrm{d}v} \leq \left(\int \exp\left(3\ell_c(\theta|\mathbf{u})/2\right)\,\|\mathbf{u}\|^4\,\phi(\mathbf{u})\,\mathrm{d}\mathbf{u}\right)^{1/2}$$

Since $\exp(\ell_c(\theta|\mathbf{u})) \leq 1$ and $\int \|\mathbf{u}\|^4\phi(\mathbf{u})\mathrm{d}\mathbf{u} = q(2+q)$, we have

$$\sup_{\theta\in\Theta}\int \|u\|^2\pi_\theta(\mathbf{u})\mathrm{d}\mathbf{u} \leq \sqrt{q(2+q)}\,.$$

Appendix B. Proof of section 5

For $\theta, \vartheta \in \Theta$, the (i,j)-th entry of the matrix $\nabla\ell(\theta) - \nabla\ell(\vartheta)$ is given by

$$\left(\nabla\ell(\theta) - \nabla\ell(\vartheta)\right)_{ij} = \int_{\mathsf{X}^p} \bar{B}_{ij}(x)\pi_\vartheta(\mathrm{d}x) - \int_{\mathsf{X}^p} \bar{B}_{ij}(x)\pi_\theta(\mathrm{d}x).$$

For $t \in [0,1]$ let

$$\pi_t(\mathrm{d}z) \stackrel{\mathrm{def}}{=} \frac{\exp\left(\left\langle \bar{B}(z), t\vartheta + (1-t)\theta\right\rangle\right)}{\int \exp\left(\left\langle \bar{B}(x), t\vartheta + (1-t)\theta\right\rangle\right)\mu(\mathrm{d}x)},$$

defines a probability measure on X^p. It is straightforward to check that

$$\left(\nabla\ell(\theta) - \nabla\ell(\vartheta)\right)_{ij} = \int \bar{B}_{ij}(x)\pi_1(\mathrm{d}x) - \int \bar{B}_{ij}(x)\pi_0(\mathrm{d}x),$$

and that $t \mapsto \int \bar{B}_{ij}(x)\pi_t(\mathrm{d}x)$ is differentiable with derivative

$$\begin{aligned}
&\frac{\mathrm{d}}{\mathrm{d}t}\int \bar{B}_{ij}(x)\pi_t(\mathrm{d}x)\\
&= \int \bar{B}_{ij}(x)\left\langle \bar{B}(x) - \int \bar{B}(z)\pi_t(\mathrm{d}z), \vartheta - \theta\right\rangle \pi_t(\mathrm{d}x),\\
&= \mathsf{Cov}_{\pi_t}\left(\bar{B}_{ij}(X), \left\langle \bar{B}(X), \vartheta - \theta\right\rangle\right),
\end{aligned}$$

where the covariance is taken assuming that $X \sim \pi_t$. Hence

$$\left|\left(\nabla\ell(\theta) - \nabla\ell(\vartheta)\right)_{ij}\right| = \left|\int_0^1 \mathrm{d}t\;\; \mathsf{Cov}_t\left(\bar{B}_{ij}(X), \left\langle \bar{B}(X), \vartheta - \theta\right\rangle\right)\right| \leq \mathsf{osc}(\bar{B}_{ij})\sqrt{\sum_{k\leq l}\mathsf{osc}^2(\bar{B}_{kl})}\|\theta - \vartheta\|_2.$$

This implies the inequality (16).

Acknowledgments: We are grateful to George Michailidis for very helpful discussions. This work is partly supported by NSF grant DMS-1228164.

References

S. Allassonnière and E. Kuhn. Convergent Stochastic Expectation Maximization algorithm with efficient sampling in high dimension. Application to deformable template model estimation. *Comput. Stat. Data An.*, 91:4–19, 2015.

C. Andrieu and E. Moulines. On the ergodicity properties of some adaptive MCMC algorithms. *Ann. Appl. Probab.*, 16(3):1462–1505, 2006.

C. Andrieu, E. Moulines, and P. Priouret. Stability of stochastic approximation under verifiable conditions. *SIAM J. Control Optim.*, 44(1):283–312, 2005.

O. Banerjee, L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *J. Mach. Learn. Res.*, 9:485–516, 2008.

H. Bauschke and P.L. Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC. Springer, New York, 2011. ISBN 978-1-4419-9466-0. With a foreword by Hédy Attouch.

A. Beck and M. Teboulle. Gradient-based algorithms with applications to signal-recovery problems. In *Convex optimization in signal processing and communications*, pages 42–88. Cambridge Univ. Press, Cambridge, 2010.

A. Benveniste, M. Métivier, and P. Priouret. *Adaptive algorithms and stochastic approximations*, volume 22 of *Applications of Mathematics (New York)*. Springer-Verlag, Berlin, 1990.

P. Biane, J. Pitman, and M. Yor. Probability laws related to the Jacobi theta and Riemann zeta functions, and Brownian excursions. *Bull. Amer. Math. Soc. (N.S.)*, 38(4):435–465 (electronic), 2001. ISSN 0273-0979.

H.M. Choi and J. P. Hobert. The polya-gamma gibbs sampler for bayesian logistic regression is uniformly ergodic. *Electronic Journal of Statistics*, 7:2054–2064, 2013.

P.L. Combettes. *Inherently parallel Algorithms in Feasibility and Optimization and their Applications*, chapter Quasi-Fejerian analysis of some optimization algorithms, pages 115–152. Elsevier Science, 2001.

P.L. Combettes and J.C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, volume 49 of *Springer Optim. Appl.*, pages 185–212. Springer, New York, 2011.

P.L. Combettes and J.C. Pesquet. Stochastic Quasi-Fejer block-coordinate fixed point iterations with random sweeping. *SIAM J. Optim.*, 25(2):1221–1248, 2015a.

P.L. Combettes and J.C. Pesquet. Stochastic Approximations and Perturbations in Forward-Backward Splitting for Monotone Operators. Technical report, arXiv:1507.07095v1, 2015b.

P.L. Combettes and V. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, 4(4):1168–1200, 2005.

A. Cotter, O. Shamir, N. Srebro, and K. Sridharan. Better mini-batch algorithms via accelerated gradient methods. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1647–1655. Curran Associates, Inc., 2011.

J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, 2011. ISSN 1532-4435.

M. Ekeberg, C. Lövkvist, Y. Lan, M. Weigt, and E. Aurell. Improved contact prediction in proteins: Using pseudolikelihoods to infer potts models. *Phys. Rev. E*, 87: 012707, 2013.

G. Fort and E. Moulines. Convergence of the Monte Carlo expectation maximization for curved exponential families. *Ann. Statist.*, 31(4):1220–1259, 2003. ISSN 0090-5364.

G. Fort, E. Moulines, and P. Priouret. Convergence of adaptive and interacting Markov chain Monte Carlo algorithms. *Ann. Statist.*, 39(6):3262–3289, 2011. ISSN 0090-5364.

G. Fort, E. Moulines, M. Vihola, and A. Schreck. Convergence of Markovian Stochastic Approximation with discontinuous dynamics. Technical report, arXiv math.ST 1403.6803, 2014.

G. Fort, B. Jourdain, E. Kuhn, T. Lelièvre, and G. Stoltz. Convergence of the Wang-Landau algorithm. *Mathematics of Computation*, 84:2297–2327, 2015.

C.J. Geyer. On the convergence of Monte Carlo maximum likelihood calculations. *J. Roy. Statist. Soc. Ser. B*, 56(1):261–274, 1994.

J. Guo, E. Levina, G. Michailidis, and J. Zhu. Joint structure estimation for categorical Markov networks. Technical report, Univ. of Michigan, 2010.

P. Hall and C.C. Heyde. *Martingale Limit Theory and its Application*. Academic Press, 1980.

H. Höfling and R. Tibshirani. Estimation of sparse binary pairwise Markov networks using pseudo-likelihoods. *J. Mach. Learn. Res.*, 10:883–906, 2009.

C. Hu, W. Pan, and J.T. Kwok. Accelerated gradient methods for stochastic optimization and online learning. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, pages 781–789, 2009.

A. Juditsky and A. Nemirovski. First-order methods for nonsmooth convex large-scale optimization, i: General purpose methods. In S. Sra, S. Nowozin, and S. Wright, editors, *Oxford Handbook of Innovation*, pages 121–146. MIT Press, Boston, 2012a.

A. Juditsky and A. Nemirovski. First-order methods for nonsmooth convex large-scale optimization, ii: Utilizing problem's structure. In S. Sra, S. Nowozin, and S. Wright, editors, *Oxford Handbook of Innovation*, pages 149–181. MIT Press, Boston, 2012b.

H. Kamisetty, S. Ovchinnikov, and D. Baker. Assessing the utility of coevolution-based residue-residue contact predictions in a sequence-and structure-rich era. *Proceedings of the National Academy of Sciences*, 2013. doi: 10.1073/pnas.1314045110.

G. Lan. An optimal method for stochastic composite optimization. *Math. Program.*, 133(1-2, Ser. A):365–397, 2012. ISSN 0025-5610.

J. Lin, L. Rosasco, S. Villa, and D.X. Zhou. Modified Fejer Sequences and Applications. Technical report, arXiv:1510:04641v1 math.OC, 2015.

G.J. McLachlan and T. Krishnan. *The EM algorithms and Extensions*. Wiley-Interscience; 2 edition, 2008.

S. Meyn and R.L. Tweedie. *Markov chains and stochastic stability*. Cambridge University Press, Cambridge, second edition, 2009. ISBN 978-0-521-73182-9. With a prologue by Peter W. Glynn.

J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *CR Acad. Sci. Paris Sér. A Math*, 255:2897–2899, 1962.

A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM J. Optim.*, 19(4):1574–1609, 2008. ISSN 1052-6234.

Y.E. Nesterov. *Introductory Lectures on Convex Optimization, A basic course*. Kluwer Academic Publishers, 2004.

A. Nitanda. Stochastic proximal gradient descent with acceleration techniques. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1574–1582. Curran Associates, Inc., 2014.

H.E. Ogden. A sequential reduction method for inference in generalized linear mixed models. *Electron. J. Statist.*, 9(1):135–152, 2015.

N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2013.

N. G. Polson, J. G. Scott, and J. Windle. Bayesian inference for logistic models using Polya-Gamma latent variables. *J. Am. Stat. Assoc.*, 108(504):1339–1349, 2013.

B.T. Polyak. *Introduction to Optimization*. xx, 1987.

P. Ravikumar, M.J. Wainwright, and J.D. Lafferty. High-dimensional Ising model selection using ℓ_1-regularized logistic regression. *Ann. Statist.*, 38(3):1287–1319, 2010.

C.P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer; 2nd edition, 2005.

L. Rosasco, S. Villa, and B.C. Vu. Convergence of a Stochastic Proximal Gradient Algorithm. Technical report, arXiv:1403.5075v3, 2014.

L. Rosasco, S. Villa, and B.C. Vu. A Stochastic Inertial Forward-Backward Splitting Algorithm for multi-variate monotone inclusions. Technical report, arXiv:1507.00848v1, 2015.

E. Saksman and M. Vihola. On the ergodicity of the adaptive Metropolis algorithm on unbounded domains. *Ann. Appl. Probab.*, 20(6):2178–2203, 2010.

J. Schelldorfer, L. Meier, and P. Bühlmann. GLMMLasso: an algorithm for high-dimensional generalized linear mixed models using ℓ_1-penalization. *J. Comput. Graph. Statist.*, 23(2):460–477, 2014.

M.W. Schmidt, N. Le Roux, and F. Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *NIPS*, pages 1458–1466, 2011. see also the technical report INRIA-00618152.

A. Schreck, G. Fort, and E. Moulines. Adaptive Equi-energy sampler : convergence and illustration. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 23(1):Art 5., 2013.

J. Shao. *Mathematical Statistics*. Springer texts in Statistics, 2003.

L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *J. Mach. Learn. Res.*, 11:2543–2596, 2010. ISSN 1532-4435.

L. Xiao and T. Zhang. A Proximal Stochastic Gradient Method with Progressive Variance Reduction. *SIAM J. Optim.*, 24:2057–2075, 2014.

L. Xue, H. Zou, and T. Cai. Non-concave penalized composite likelihood estimation of sparse ising models. *Ann. Statist.*, 40:1403–1429, 2012.

Journal of Machine Learning Research 18 (2017) 1-39 Submitted 5/15; Revised 2/17; Published 3/17

Differential Privacy for Bayesian Inference through Posterior Sampling*

Christos Dimitrakakis CHRISTOS.DIMITRAKAKIS@GMAIL.COM
University of Lille, F-59650 Villeneuve-d'Ascq, France
SEAS, Harvard University, Cambridge MA-02138, USA
DIT, Chalmers University of Technology, SE-412 96, Gothenburg, Sweden

Blaine Nelson BLAINE.NELSON@GOOGLE.COM
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043, USA

Zuhe Zhang ZHANG.ZUHE@GMAIL.COM
School of Mathematics & Statistics
The University of Melbourne
Parkville, VIC 3010, Australia

Aikaterini Mitrokotsa AIKMITR@CHALMERS.SE
Department of Computer Science & Engineering
Chalmers University of Technology
SE-412 96, Gothenburg, Sweden

Benjamin I. P. Rubinstein BRUBINSTEIN@UNIMELB.EDU.AU
School of Computing & Information Systems
The University of Melbourne
Parkville, VIC 3010, Australia

Editor: Charles Elkan

Abstract

Differential privacy formalises privacy-preserving mechanisms that provide access to a database. Can Bayesian inference be used directly to provide private access to data? The answer is yes: under certain conditions on the prior, sampling from the posterior distribution can lead to a desired level of privacy and utility. For a uniform treatment, we define differential privacy over arbitrary data set metrics, outcome spaces and distribution families. This allows us to also deal with non-i.i.d or non-tabular data sets. We then prove bounds on the sensitivity of the posterior to the data, which delivers a measure of robustness. We also show how to use posterior sampling to provide differentially private responses to queries, within a decision-theoretic framework. Finally, we provide bounds on the utility of answers to queries and on the ability of an adversary to distinguish between data sets. The latter are complemented by a novel use of Le Cam's method to obtain lower bounds on distinguishability. Our results hold for arbitrary metrics, including those for the common definition of differential privacy. For specific choices of the metric, we give a number of examples satisfying our assumptions.

∗. A preliminary version of this paper appeared in *Algorithmic Learning Theory 2014* (Dimitrakakis et al., 2014). This version corrects proofs, constant factors in the upper bounds and introduces new material on utility analysis, lower bounds and examples.

©2017 C. Dimitrakakis, B. Nelson, Z. Zhang, A. Mitrokotsa, B. I. P. Rubinstein.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-257.html.

Keywords: Bayesian inference, differential privacy, robustness, adversarial Learning

1. Introduction

The Bayesian framework for statistical decision theory incorporates uncertainty into decision making in a probabilistic manner. This makes the framework attractive, as predictions and modelling can all be made with the machinery of probability. More specifically, a Bayesian statistician begins by assuming that the world is described by a probabilistic model within some family, and he assigns a prior belief to each one of the models. After observing data, this belief is adjusted through Bayes's theorem to the so called posterior belief. This expresses the statistician's conclusion given the data and the prior assumptions. The statistician can then release the posterior to the world, for others to build upon, or use for principled decision making under uncertainty.

Unfortunately, it is frequently the case that the data acquired by the statistician is sensitive. Consequently, there is a fear that any information released by the statistician that depends on the data—be that the posterior distribution itself or any decisions that follow from the calculated posterior—may reveal sensitive information in the original data. Recently, the framework of differential privacy has been proposed to codify this leakage of information. If an algorithm is differentially private, then its output can only leak a bounded amount of information about its input.

We are interested in the question of how one can build differentially-private algorithms within the Bayesian framework. More precisely, we examine when the choice of prior is sufficient to guarantee differential privacy for decisions that are derived from the posterior distribution. Our work develops a unified understanding of privacy and learning in adversarial environments, under a decision-theoretic framework. We show that under suitable assumptions, standard Bayesian inference and posterior sampling can achieve uniformly good utility with a fixed privacy budget in the differential privacy setting. We also indicate strong connections between robustness and privacy. Under the base level of data privacy provided by the posterior distribution, the statistician can safely respond to external queries using samples from the posterior. When estimating a linear model from sensitive data, for example, samples from the posterior correspond to different possible fits. The more samples used, the more privacy is leaked, while query responses may be more accurate.

Our proposed approach complements existing mechanisms rather well, and may be particularly useful in situations where Bayesian inference is already in use. For this reason, we provide illustrative examples in the exponential family. However, our setting is wholly general and not limited to specific distribution families, or i.i.d. observations. Any family could be chosen: so long as it either satisfies our assumptions directly, or can be restricted so that it does. For example, our framework applies to families of discrete Bayesian networks with directed-acyclic topologies (*e.g.*, Markov chains; see Lemma 24 on page 21) and multivariate Gaussians (see Lemma 23), where the observations may not satisfy the i.i.d. assumption.

Summary of setting. A Bayesian statistician ($\mathscr{B}$) wishes to communicate results about data x to a third party ($\mathscr{A}$), but without revealing the data x itself. We make no assumptions on the data x, which could be a single observation, an i.i.d. sample, or a sequence of observations. The protocol of interaction between $\mathscr{B}$ and $\mathscr{A}$ is summarised below.

1. $\mathscr{B}$ selects a model family ($\mathcal{F}_\Theta$) and a prior (ξ).
2. $\mathscr{A}$ is allowed to see $\mathcal{F}_\Theta$ and ξ and is computationally unbounded.
3. $\mathscr{B}$ observes data x and calculates the posterior $\xi(\theta \mid x)$ but does not reveal it. Then, for steps $t = 1, 2, \ldots$, repeat the following:
4. $\mathscr{A}$ sends his utility function u and a query q_t to $\mathscr{B}$.
5. $\mathscr{B}$ responds with the response r_t maximising u, in a manner that depends on the query and the posterior.

Let us now elaborate. In this framework, the choice of the model family $\mathcal{F}_\Theta$ is dictated by the problem. The choice of ξ is normally determined by the prior knowledge of $\mathscr{B}$, but we show that this also affects what level of privacy is achieved. Informally speaking, informative priors achieve better privacy, as the posterior has a weaker dependency on the data. It is natural to assume that the prior itself is public, as it should reflect publicly available information. The statistician's conclusion from the observed data x is then summarised in the posterior distribution $\xi(\theta \mid x)$, which remains private.

The second part of the process is the interaction with $\mathscr{A}$. We adopt a decision-theoretic viewpoint to characterise what the optimal responses to queries should be. More specifically, we assume the existence of a "true" parameter $\theta \in \Theta$, and that $\mathscr{A}$ has a utility function $u_\theta(q_t, r_t)$, which he wishes to maximise. For example, consider the case where $\theta = (\mu, \Sigma)$ are the parameters of a normal distribution. An example query q_t is *"what is the expected value* $\mathbb{E}_\theta\, x_i = \mu$ *of the distribution?"*. The optimal response r_t, would then be a real vector that depends on the utility function. A possible utility function is the negative squared L_2 distance:

$$u_\theta(q_t = \text{"what is the mean?"}, r_t) = -\| \mathbb{E}_\theta\, x_i - r_t \|_2^2.$$

While θ is unknown, $\mathscr{B}$ has information about it in the form of a posterior distribution. Using standard decision-theoretic notions, the optimal response of $\mathscr{B}$ would maximise the expected utility $\mathbb{E}_\xi(u \mid q_t, r_t, x)$, where the expectation is taken over the posterior distribution. However, this deterministic response cannot be differentially private.

In this paper, we promote the use of *posterior sampling* to respond to queries. The posterior sampling mechanism draws a set $\hat{\Theta}$ of i.i.d. samples from the posterior distribution. Then, all the responses only depend on the posterior through $\hat{\Theta}$. Since our algorithm only takes a single sample set $\hat{\Theta}$, further queries by the adversary reveal nothing more about the data than what can be inferred from $\hat{\Theta}$. The empirical distribution induced by $\hat{\Theta}$ serves as a private surrogate for the exact (non-private) posterior. Consequently, we can respond to an arbitrary number of queries with a bounded privacy budget, while guaranteeing good utility for all responses.

We show that if $\mathcal{F}_\Theta$ and ξ are chosen appropriately, this results in differentially-private responses, as well as robustness of the posterior.[1] In addition, we prove upper and lower bounds on how easy it is for an adversary to distinguish two ϵ-close data sets. Finally, we bound the loss in utility incurred due to privacy. The intuition behind our results

1. More specifically, that small changes in the data result in small changes in the posterior in terms of the KL divergence.

is that robustness and privacy are linked via smoothness. Learning algorithms that are smooth mappings—their output (*e.g.*, a spam filter) varies little with perturbations to input (*e.g.*, similar training corpora)—are robust: outliers have reduced influence, and adversaries cannot easily discover unknown information about the data. This suggests that robustness and privacy can be simultaneously achieved and are in fact deeply linked.

We provide a uniform mathematical treatment of the privacy and robustness properties of Bayesian inference based on generalised differential privacy to arbitrary data set distances, outcome spaces, and distribution families. This paper can be summarised as making the following distinct contributions:

- Under certain regularity conditions on the prior distribution ξ or likelihood family $\mathcal{F}_\Theta$, we show that the posterior distribution is *robust*: small changes in the data set result in small posterior changes.
- We introduce a novel *posterior sampling mechanism* that is private.[2] Unlike other common mechanisms in differential privacy, our approach sits squarely in the non-private (Bayesian) learning framework without modification.
- We provide necessary and sufficient conditions for differentially private Bayesian inference.
- We introduce the notion of *data set distinguishability* for which we provide finite-sample bounds for our mechanism: how large would $\hat{\Theta}$ need to be for $\mathscr{A}$ to distinguish two data sets with high probability.
- We provide examples of conjugate-pair distributions where our assumptions hold, including discrete Bayesian networks. We find that even though Bayesian posterior sampling does provide privacy guarantees directly, those appear to be very weak for standard conjugate families. However, with a small modification of the prior, it is easy to obtain good privacy guarantees.

Paper organisation. Section 2 specifies the setting and our assumptions. Section 3 proves results on robustness of Bayesian learning. Section 4 proves our main privacy results. In particular, Section 4.1 shows that the posterior distribution is differentially private, Section 4.2 describes our posterior sampling query response algorithm, Section 4.3 derives bounds on data set indistinguishability, Section 4.5 shows how to obtain matching lower bounds for distinguishability, while Section 4.4 shows how utility and privacy can be traded off within our framework. Examples where our assumptions hold are given in Section 5. We present a discussion of our results, related work and links to the exponential mechanism and robust Bayesian inference in Section 6. Appendix A contains proofs of the main theorems. Finally, Appendix B details proofs of the examples demonstrating our assumptions.

2. Problem Setting

We consider the problem of a Bayesian statistician ($\mathscr{B}$) communicating with an untrusted third party ($\mathscr{A}$). $\mathscr{B}$ wants to convey useful responses to the queries of $\mathscr{A}$ (*e.g.*, how

2. Although previously used *e.g.*, for efficient exploration in reinforcement learning (Thompson, 1933; Osband et al., 2013), posterior sampling has not previously been employed for privacy.

many people suffer from a disease or vote for a particular party) without revealing private information about the original data (*e.g.*, whether a particular person has cancer). This requires communicating information in a way that strikes a balance between utility and privacy. In this paper, we study the inherent privacy and robustness properties of Bayesian inference and explore the question of whether $\mathscr{B}$ can select a prior distribution so that a computationally unbounded $\mathscr{A}$ cannot obtain private information from queries.

2.1 Definitions and Notation

We begin with our notation. Let $\mathcal{S}$ be the set of all possible data sets. For example, if $\mathcal{X}$ is a finite alphabet, then we might have $\mathcal{S} = \bigcup_{n=0}^{\infty} \mathcal{X}^n$, *i.e.*, the set of all possible observation sequences over $\mathcal{X}$. However, $\mathcal{S}$ can have arbitrary structure and so social network or mobility trace data are also handled in this framework. Probability measures on parameters θ are usually denoted by ξ, while measures and densities on data are denoted by P_θ or p_θ respectively. Expectations are denoted by $\mathbb{E}_\xi\, g \triangleq \int_\Theta g(\theta)\,\mathrm{d}\xi(\theta)$, where the subscript denotes the underlying distribution with respect to which we are taking expectations. In case of ambiguity, we explicitly write *e.g.*, $\mathbb{E}_{x\sim P_\theta}\, f(x) = \int_{\mathcal{S}} f(x)\,\mathrm{d}P_\theta(x)$ to denote which variables are drawn from which distributions. Finally, we use $\mathbb{I}\{\pi\}$ to be the identity function, taking the value 1 when the predicate π is true, and 0 otherwise.

2.1.1 Distances Between Data Sets

Central to the notions of privacy and robustness, is the concept of distance between data sets. Firstly, the effect of data set perturbation on learning depends on the amount of noise as quantified by some distance. This is useful for characterising robustness to noise or adversarial manipulation of the data. Secondly, the amount that an attacker can learn from queries can be quantified in terms of the distance of his guesses to the true data set. Finally, it allows for a unified mathematical treatment, as it permits different types of neighbourhoods to be defined. To model these situations, we equip $\mathcal{S}$ with a pseudo-metric[3] $\rho : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_+$. This generalisation has also been used by Chatzikokolakis et al. (2013), which has laid the groundwork for metric-based differential privacy. While this concept has many applications in the context of geographical information systems, we apply this generalisation of differential privacy without necessarily referring to some underlying physical distance.

2.1.2 Bayesian Inference

This paper focuses on the *Bayesian inference* setting, where the statistician $\mathscr{B}$ constructs a posterior distribution from a prior distribution ξ and a training data set x. More precisely, we assume that data $x \in \mathcal{S}$ have been drawn from some distribution $P_{\theta^\star}$ on $\mathcal{S}$, parameterised by $\theta^\star$, from a family of distributions $\mathcal{F}_\Theta$. $\mathscr{B}$ defines a parameter set Θ indexing a family of distributions $\mathcal{F}_\Theta$ on $(\mathcal{S}, \mathfrak{S}_\mathcal{S})$, where $\mathfrak{S}_\mathcal{S}$ is an appropriate σ-algebra on $\mathcal{S}$:

$$\mathcal{F}_\Theta \triangleq \{\, P_\theta : \theta \in \Theta \,\},$$

3. Meaning that $\rho(x, y) = 0$ does not necessarily imply $x = y$.

and where we use p_θ to denote the corresponding densities[4] when necessary. To perform inference in the Bayesian setting, $\mathscr{B}$ selects a prior measure ξ on $(\Theta, \mathfrak{S}_\Theta)$ reflecting $\mathscr{B}$'s subjective beliefs about which θ is more likely to be true, *a priori*; *i.e.*, for any measurable set $B \in \mathfrak{S}_\Theta$, $\xi(B)$ represents $\mathscr{B}$'s prior belief that $\theta^\star \in B$. In general, the posterior distribution after observing $x \in \mathcal{S}$ is:

$$\xi(B \mid x) = \frac{\int_B p_\theta(x)\,\mathrm{d}\xi(\theta)}{\phi(x)}\ , \tag{1}$$

where ϕ is the corresponding marginal density given by:

$$\phi(x) \triangleq \int_\Theta p_\theta(x)\,\mathrm{d}\xi(\theta)\ .$$

While the choice of the prior is generally arbitrary, this paper shows that its careful selection can yield good privacy guarantees. Throughout the paper, we shall use the following simple example to ground our observations and theory. This consists of a finite family of distributions, on a finite alphabet. Consequently, calculation of the posterior distribution is always simple. It is also easy to verify our assumptions on this model.

Example 1 (Finite Bernoulli family.) *Consider a finite family of distributions $\mathcal{F}_\Theta = \{\, P_\theta : \theta \in \Theta \,\}$ on alphabet $\mathcal{X} = \{\, 0, 1 \,\}$, with $\theta \in [0, 1]$, such that for any model in the family and any observation x*

$$P_\theta(x) = \theta\, \mathbb{I}\{x = 1\} + (1 - \theta)\, \mathbb{I}\{x = 0\}\,.$$

For any sequence of observations $x_1, \ldots, x_T$, we have, with some abuse of notation,

$$P_\theta(x_1, \ldots, x_T) = \prod_{t=1}^{T} P_\theta(x_t),$$

i.e., *P_θ defines an i.i.d. distribution on the alphabet. This family corresponds to a set of Bernoulli models. The set of parameters Θ will be chosen to discretise the parameter space of Bernoulli distributions over Δ-sized intervals. For this, the k-th model's parameter will be $\theta_k = \Delta k$, with $\Delta \in (0, 1)$ and $k \leq 1/\Delta$.*

For the above family, we can use a uniform prior distribution $\xi(\theta_k) = \Delta$. The posterior distribution is easily calculated, since we need only sum over a finite number of parameters.

2.1.3 Privacy

We now recall the concept of differential privacy (Dwork, 2006). This states that on *neighbouring* data sets, a randomised query response mechanism yields (pointwise) similar distributions. We adopt the view of mechanisms as conditional distributions under which differential privacy can be seen as a measure of smoothness. In our setting, conditional distributions conveniently correspond to posterior distributions. These can also be interpreted as the distribution of a mechanism that uses posterior sampling, to be introduced in Section 4.2. The precise definition depends on the notion of neighbourhood, with the following choice being common:

4. *I.e.*, the Radon-Nikodym derivative of P_θ relative to some dominating measure ν.

Definition 1 ((ϵ, δ)-differential privacy) *A conditional distribution $P(\cdot \mid x)$ on $(\Theta, \mathfrak{S}_\Theta)$ is* (ϵ, δ)-differentially private *if, for all $B \in \mathfrak{S}_\Theta$ and for any $x \in \mathcal{S} = \mathcal{X}^n$*

$$P(B \mid x) \leq e^\epsilon P(B \mid y) + \delta,$$

for all y in the Hamming-1 neighbourhood *of x. That is, y may differ in at most one entry from x: there is at most one $i \in \{1, \ldots, n\}$ such that $x_i \neq y_i$.*

A typical situation where this definition is employed, is when x, y are matrices and x_i is a single row in the matrix. Then, the data sets are neighbours if a matrix row is changed.[5]

In our setting, it is reasonable to generalise this to arbitrary data set spaces $\mathcal{S}$ that are not necessarily product spaces. To do so, we use the notion of differential privacy under a pseudo-metric ρ on the space of all data sets, which allows for more subtle representations of attacker knowledge and for a more general treatment:

Definition 2 ((ϵ, δ)-differential privacy under ρ.) *A conditional distribution $P(\cdot \mid x)$ on $(\Theta, \mathfrak{S}_\Theta)$ is* (ϵ, δ)-differentially private under *a pseudo-metric $\rho : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_+$ if, for all $B \in \mathfrak{S}_\Theta$ and for any $x, y \in \mathcal{S}$,*

$$P(B \mid x) \leq e^{\epsilon\rho(x,y)} P(B \mid y) + \delta\rho(x, y)\ .$$

In our setting, ρ replaces the notion of neighbourhood. It is of course possible to use ρ that corresponds to the usual meaning of neighbourhood in differential privacy:

Remark 3 *If $\mathcal{S} = \mathcal{X}^n$ and $\rho(x, y) = \sum_{i=1}^n \mathbb{I}\{x_i \neq y_i\}$ is the Hamming distance, this definition is analogous to standard (ϵ, δ)-differential privacy. When considering only $(\epsilon, 0)$-differential privacy or $(0, \delta)$-privacy, it is an equivalent notion.*[6]

Proof For $(\epsilon, 0)$-DP, let $\rho(x, z) = \rho(z, y) = 1$; *i.e.*, the data differ in one element. Then, from standard DP, we have $P(B \mid x) \leq e^\epsilon P(B \mid z)$ and so obtain $P(B \mid x) \leq e^{2\epsilon} P(B \mid y) = e^{\rho(x,y)\epsilon} P(B \mid y)$. By induction, this holds for any x, y pair. Similarly, for $(0, \delta)$-DP, by induction we obtain $P(B \mid x) \leq P(B \mid y) + \delta\rho(x, y)$. ■

Definition 1 allows for privacy against a powerful attacker $\mathscr{A}$, who attempts to match the empirical distribution induced by the true data set, by querying the learned mechanism and comparing its responses to those given by distributions simulated using knowledge of the mechanism and knowledge of all but one datum—narrowing the data set down to a Hamming-1 ball. Indeed the requirement of differential privacy is sometimes *too strong* since it may come at the price of utility. Definition 2 allows for a much broader encoding of the attacker's knowledge via the selected pseudo-metric. It also allows a more fine-grained notion of privacy. This is quite useful for geographical information systems, as proposed by Chatzikokolakis et al. (2013), to which we refer the reader for a broader discussion of the use of metrics in differential privacy.

Finally, we can show that this generalisation of differential privacy satisfies the standard composition property.

5. Another common choice for neighbourhoods is to say that two data sets are neighbours if one results from the other by addition of a row.
6. Making the definition wholly equivalent is possible, but results in an unnecessarily complex definition.

Theorem 4 (Composition) *Let conditional distributions $P(\cdot \mid x)$ on $(\Theta, \mathfrak{S}_\Theta)$ be* (ϵ, δ)-differentially private under *a pseudo-metric $\rho : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_+$ and $P'(\cdot \mid x)$ on $(\Theta', \mathfrak{S}'_{\Theta'})$ be* (ϵ', δ')-differentially private under *the same pseudo-metric. Then the conditional distribution on the product space $(\Theta \times \Theta', \mathfrak{S}_\Theta \otimes \mathfrak{S}'_{\Theta'})$ given by*

$$Q(B \times B' \mid x) = P(B \mid x) P'(B' \mid x), \forall B \times B' \in \mathfrak{S}_\Theta \otimes \mathfrak{S}'_{\Theta'}$$

satisfies $(\epsilon + \epsilon', \delta + \delta')$-differentially private under *the pseudo-metric ρ. Here $\mathfrak{S}_\Theta \otimes \mathfrak{S}'_{\Theta'}$ is the product σ-algebra on $\Theta \times \Theta'$.*

Proof For any $y \in \mathcal{S}$

$$\begin{aligned} Q(B \times B' \mid x) &\leq \left[e^{\epsilon\rho(x,y)} P(B \mid y) + \delta\rho(x, y)\right] P'(B' \mid x) \\ &\leq e^{\epsilon\rho(x,y)} P(B \mid y) \left[e^{\epsilon'\rho(x,y)} P'(B' \mid y) + \delta'\rho(x, y)\right] + \delta\rho(x, y) \\ &\leq e^{(\epsilon+\epsilon')\rho(x,y)} P(B \mid y) P'(B' \mid y) + (\delta + \delta')\rho(x, y) \end{aligned}$$

■

2.2 Our Main Assumptions

In the sequel, we show that if the distribution family $\mathcal{F}_\Theta$ or prior ξ satisfies certain assumptions, then close data sets $x, y \in \mathcal{S}$ result in posterior distributions that are close. In that case, it is difficult for a third party to use such a posterior to distinguish the true data set x from similar data sets.

To formalise these notions, we introduce two possible assumptions one could make on the smoothness of the family $\mathcal{F}_\Theta$ with respect to some metric d on $\mathbb{R}_+$. The first assumption states that the likelihood is smooth for all parameterisations of the family. First, we define our notion of smoothness. Let $f(x, \theta) \triangleq \ln p_\theta(x)$ be the log probability of x under θ. The Lipschitz constant for a parameter value θ is:

$$\ell(\theta) \triangleq \inf \{\, u : |f(x, \theta) - f(y, \theta)| \leq u\rho(x, y) \forall x, y \in \mathcal{S} \,\}. \tag{2}$$

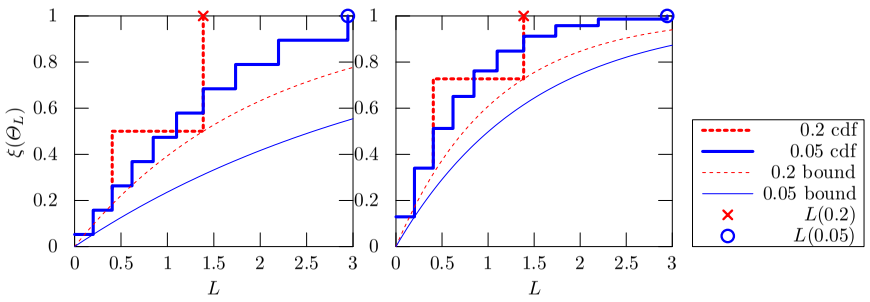
Our first assumption is uniform smoothness for all parameters.

Assumption 1 (Lipschitz continuity) *We assume there exists some $L < \infty$ such that:*

$$\ell(\theta) \leq L, \qquad \theta \in \Theta. \tag{3}$$

In other words, this assumption says that the log probability is Lipschitz with respect to ρ for any parameter value. Consider Example 1 for the Bernoulli model. It is easy to see that a model with Δ-sized intervals satisfies the above assumption with $L = \ln 1/\Delta$.

However, it may be difficult for this assumption to hold uniformly over Θ in general. This can be seen by the following counterexample for the Bernoulli family of distributions: when the parameter is 0, then any sequence $x = 0, 0, \ldots$ has probability 1, while any



(a) Uniform prior (b) Exponential prior

Figure 1: The mass of L-Lipschitz parameters, for two finite familes of Bernoulli distributions with $\Delta \in \{0.2, 0.05\}$ (thick lines) together with their respective stochastic Lipschitz bounds (thin lines) and the corresponding uniform Lipschitz constant L.

sequence containing a 1 has probability 0. The same thing occurs when we take $\Delta \to 0$ in Example 1. To avoid such problems, we relax the assumption by only requiring that $\mathscr{B}$'s *prior* probability ξ is concentrated in the regions of the family for which the likelihood is smoothest:

Assumption 2 (Stochastic Lipschitz continuity; Norkin, 1986) *First, define the subset of parameter values*

$$\Theta_L \triangleq \{\theta \in \Theta : \ell(\theta) \leq L\} \tag{4}$$

to be those parameters for which Lipschitz continuity holds with Lipschitz constant L. Then, there are some constants $c, L_0 > 0$ such that, for all $L \geq L_0$:

$$\xi(\Theta_L) \geq 1 - \exp(-c(L - L_0)) \ . \tag{5}$$

By not requiring uniform smoothness, this weaker assumption is easier to meet but still yields useful guarantees. In fact, in Section 5, we demonstrate that this assumption is satisfied by many important example distribution families. However, it will be illustrative to consider the discrete Bernoulli family example at this point.

Example 2 (Continuation of Example 1) *These conditions can be examined in terms of the finite family of Example 1. Figure 1 demonstrates the assumptions for $\Delta = 0.2$ (red dashed lines) and $\Delta = 0.05$ (blue solid lines).*

In particular, the two thick lines Figure 1a show the probability mass of L-Lipschitz parameters for the two families. They are both step functions, as the families are discrete.[7]

7. The $\Delta = 0.2$ family only has two steps, as the Lipschitz constant is symmetric about $\theta = 0.5$.

The $\times$ and $\circ$ symbols show the corresponding Lipschitz constants for the two families respectively, and we can clearly see $\Delta = 0.2$ has about half the Lipschitz constant of $\Delta = 0.05$. The thinner curves depict the highest lower bound on the probability mass defined in Assumption 2. There we see that the higher curve is achieved by $\Delta = 0.2$.

In order to improve the lower bound, we need to modify our prior distribution on the family members so as to place less mass on the more sensitive parameters. The result of this operation is shown in Figure 1b, which uses the prior $\xi(\theta) \propto \exp(-\ell(\theta))$, i.e., it places exponentially smaller weight in more sensitive parameters. This results in both lower bounds being shifted upwards, corresponding to a higher c constant in Assumption 2. Of course, this has no effect on Assumption 1.

For completeness, we now show that verifying our assumptions for a distribution of a single random variable lifts to a corresponding property for the product distribution on i.i.d. samples.

Lemma 5 *If $\mathcal{F}_\Theta$ satisfies Assumption 1 (resp. Assumption 2) with respect to pseudo-metric ρ and constant L (or c), then, for any fixed $n \in \mathbb{N}$, the product family $\mathcal{F}_\Theta^n$ with densities (sim. measures) $p_\Theta^n(\{x_i\}) = \prod_{i=1}^n p_\Theta(x_i)$ satisfies the same assumption with respect to:*

$$\rho^n(\{x_i\}, \{y_i\}) = \sum_{i=1}^n \rho(x_i, y_i)$$

and constant L (or c).

2.2.1 Necessary Conditions

Finally, let us discuss whether the above conditions are necessary to achieve differential privacy. In fact, either the first condition must be true, or a similar condition must hold on the marginals for every possible data set pair (x, y). Our second condition can be seen as a specific case of the necessary condition for the marginals, as explained below.

Theorem 6 *For a prior ξ to be $2L$-differentially private for a family $\mathcal{F}_\Theta$, either*

$$\sup_{\theta \in \Theta} \ln \frac{P_\theta(x)}{P_\theta(y)} \leq L\rho(x, y), \qquad \text{or} \qquad \ln \frac{\phi(y)}{\phi(x)} \leq L\rho(x, y) \tag{6}$$

for all $x, y \in \mathcal{X}$.

Proof If neither condition holds for some pair (x, y) then there is θ such that $\ln \frac{P_\theta(x)}{P_\theta(y)} > L\rho(x, y)$ and $\ln \frac{\phi(y)}{\phi(x)} > L\rho(x, y)$. Simply adding the two, we obtain $\ln \frac{\xi(\theta|x)}{\xi(\theta|y)} > 2L\rho(x, y)$, and so the resulting posterior is not L-differentially private. ■

In our main results, we show that the first part of the conditions, which is equivalent to our first assumption, is also sufficient. However, the second part is too weak to imply differential privacy on its own.

2.2.2 The Choice of Metric and Sufficient Statistics

The extent to which our assumptions hold for a particular family of distributions $\mathcal{F}_\Theta$ depends mainly on ρ.[8] The choice of metric is also important for achieving differential privacy with

8. Although our results are stated in terms of metrics, it is easy to translate them to neighbourhood-based results, simply by bounding the ρ-distance of any neighbouring data sets. See also the discussion in Section 6.

respect to it. Let us specifically consider metrics defined in terms of a difference in statistics:

$$\rho(x,y) \triangleq \|\tau(x) - \tau(y)\| \ ,$$

where $\tau : \mathcal{S} \to \mathcal{V}$ is a statistic mapping from data sets to a normed vector space.

In that case, our assumptions imply that τ must be a *sufficient* statistic, since if $\tau(x) = \tau(y)$ then $\rho(x,y) = 0$ and it follows that $P_\theta(x) = P_\theta(y)$. More generally, ρ must be such that if the distance between x, y is zero, then their probabilities should be equal. We will see some examples of such statistics for conjugate distributions in the exponential family in Section 5. That means that we cannot use a metric which simply ignores part of the data, for example.

Similarly, the very definition of differential privacy (Definition 2) implies that τ must be a *Bayes-sufficient* statistic. That means that for any x, y, it holds

$$\tau(x) = \tau(y) \quad \Rightarrow \quad \xi(B \mid x) = \xi(B \mid y) \ , \quad \forall B \in \mathfrak{S}_\Theta \ .$$

Note that this is a slightly weaker condition than a sufficient statistic, which is necessary for our assumptions to hold.

2.3 Summary of Results

Given the above assumptions, we show: firstly, that if we choose an informative prior ξ, the resulting posterior is robust in terms of KL-divergence to small changes in the data. Secondly, that the posterior distribution is differentially private. Thirdly, that this implies that sampling from the posterior can be used as part of a differentially-private mechanism. We complement these with results on how easily an adversary can distinguish two similar data sets from posterior samples. Finally, we characterise the trade-off between utility and privacy, stated here informally for ease of exposition:

Claim 1 *If $\mathscr{A}$ prefers to use the prior $\xi^\star$, but $\mathscr{B}$ uses a prior ξ satisfying Assumption 1, and $\mathscr{A}$'s utility is bounded in $[0,1]$, the following is true for the posterior sampling mechanism with N samples:*

- *The mechanism is $2NL$-differentially private.*
- *$\mathscr{A}$'s utility loss is $O\left([1 - \xi^\star(\Theta_L)] + \sqrt{1/N}\right)$ w.h.p., where Θ_L is the support of ξ.*

The following sections discuss our main results in detail. We begin by proving that our assumptions result in robust posteriors, in the sense that the KL divergence between posteriors arising from similar data sets is small. Then we show that they also result in differentially private posterior distributions, and analyse the resulting posterior sampling mechanism. We conclude with some examples and a discussion of related work.

3. Robustness of the Posterior Distribution

We now show that the above assumptions provide guarantees on the robustness of the posterior. That is, if the distance between two data sets x, y is small, then so too is the distance between the two resulting posteriors, $\xi(\cdot \mid x)$ and $\xi(\cdot \mid y)$. We prove this result for

the case where we measure the distance between the posteriors in terms of the well-known KL-divergence:

$$D\left(P \parallel Q\right) = \int_S \ln \frac{\mathrm{d}P}{\mathrm{d}Q} \, \mathrm{d}P \ .$$

The following theorem shows that any distribution family $\mathcal{F}_\Theta$ and prior ξ satisfying one of our assumptions is robust, in the sense that the posterior does not change significantly with small changes to the data set. It is notable that our mechanisms are simply tuned through the choice of prior.

Theorem 7 *When ξ is a prior distribution on Θ and $\xi(\cdot \mid x)$ and $\xi(\cdot \mid y)$ are the respective posterior distributions for data sets $x, y \in \mathcal{S}$, the following results hold:*

1. *Under a pseudo-metric ρ and $L > 0$ satisfying Assumption 1,*

$$D\left(\xi(\cdot \mid x) \parallel \xi(\cdot \mid y)\right) \leq 2L\rho(x,y) \ . \tag{7}$$

2. *Under a pseudo-metric ρ and $c > 1$ satisfying Assumption 2,*

$$D\left(\xi(\cdot \mid x) \parallel \xi(\cdot \mid y)\right) \leq C_\xi^{\mathcal{F}_\Theta} \left(1 + 2L_0 + c^{-1}\right) \rho(x,y) \ , \tag{8}$$

where $C_\xi^{\mathcal{F}_\Theta}$ is the ratio between the maximum and marginal likelihoods (9), and assuming there exists $\chi \in (0,1]$ such that: $\forall x, y \in \mathcal{S}$ there is a sequence $\{z_k\} \subset \mathcal{S}$, with $z_0 = x, z_n = y$, satisfying $\chi\rho(z_k, z_{k+1}) \leq c - 1 \ \forall z_k$.

Note that the second claim bounds the KL divergence in terms of $\mathscr{B}$'s prior belief that L is small, which is expressed via the constant c. The larger c is, the less prior mass is placed in large L and so the more robust inference becomes. Of course, choosing c to be too large may decrease efficiency.

It is important to also discuss the constant $C_\xi^{\mathcal{F}_\Theta}$. To get a better intuition, consider the case where $\Theta, \mathcal{X}$ are finite. Let $\theta^\star_{\mathrm{ML}}(x)$ be the maximum-likelihood estimate for x. Then we have that:

$$C_\xi^{\mathcal{F}_\Theta} \quad = \quad \max_x \frac{P_{\theta^\star_{\mathrm{ML}}(x)}(x)}{\sum_\Theta P_\theta(x)\xi(\theta)} \quad \leq \quad \max_x \frac{1}{\xi(\theta^\star_{\mathrm{ML}}(x))} \ , \tag{9}$$

there is therefore a natural dependency on the prior mass placed on maximum-likelihood estimators.

Finally, χ is going to be 1 for most metric spaces of interest. A notable exception is when the Hamming distance is used, which requires $\chi < 1$ as an additional technical condition for $c < 2$. However, this only affects our results under the second assumption.

4. Privacy and Utility

We next examine the differential privacy of the posterior distribution. We show in Section 4.1 that this can be achieved under either of our assumptions. The result can also be

interpreted as the differential privacy of a *posterior sampling mechanism* for responding to queries (described in Section 4.2), for which we prove a bound on the utility depending on the number of samples taken. Section 4.3 examines an alternative notion of privacy, *data set distinguishability*, similar to Wasserman and Zhou (2010). For this, we prove a bound on privacy, that also depends on the number of samples taken. Together, these exhibit a trade off between utility and privacy controlled by choosing the number of samples appropriately, in a manner described in Section 4.4.

4.1 Differential Privacy of Posterior Distributions

We consider our generalised notion of differential privacy for posterior distributions (Definition 2); and show that the type of differential privacy exhibited by the posterior depends on which assumption holds.

Theorem 8 *1. Under a pseudo-metric ρ and $L > 0$ satisfying Assumption 1, for all $x, y \in \mathcal{S}$, $B \in \mathfrak{S}_\Theta$:*

$$\xi(B \mid x) \leq \exp\{2L\rho(x,y)\}\xi(B \mid y) \; .$$

i.e., *the posterior ξ is $(2L, 0)$-differentially private under pseudo-metric ρ.*

2. Under a pseudo-metric ρ and $c > 1$ satisfying Assumption 2, $C_\xi^{\mathcal{F}_\Theta}$ defined in (9), for all $x, y \in \mathcal{S}$, $B \in \mathfrak{S}_\Theta$:

$$|\xi(B \mid x) - \xi(B \mid y)| \leq \sqrt{\frac{C_\xi^{\mathcal{F}_\Theta}}{2}\left(1 + 2L_0 + c^{-1}\right)\rho(x,y)},$$

i.e., *the posterior ξ is $\left(0, O(\sqrt{C_\xi^{\mathcal{F}_\Theta}(L_0 + 1/c)})\right)$-differentially private*[9] *under pseudo-metric $\sqrt{\rho}$.*

The difference between the two bounds' form is due to the fact that while the first claim has a direct proof, the second claim arises from the KL divergence bound in Theorem 7.

Finally, we show that posterior distributions are also randomly differentially private.

Corollary 9 *Under pseudo-metric ρ, $c > 1$ and $L \geq L_0 > 0$ satisfying Assumption 2:*

$$\mathbb{P}\left[\forall B \in \mathfrak{S}_\Theta : \xi(B \mid x) \leq \exp\left\{2L\rho(x,y)\right\}\xi\left(B \mid y\right), \forall x, y \in \mathcal{S}\right] \geq 1 - \exp(-c(L - L_0)) \; .$$

i.e., *the posterior ξ is $(2L, 0, \exp(-c(L - L_0)))$-randomly differentially private (Hall et al., 2011) under pseudo-metric ρ.*

This is a conceptually different definition from the original RDP, as the measure over which the randomness is defined is not the data distribution, but the prior measure ξ.

This property of the posterior distribution directly leads to the definition of a posterior sampling mechanism which will be differentially private. This is explained in the following section.

9. This holds, for example, for hamming distance as in the Beta-Binomial example presented in Lemma 21.

4.2 Posterior Sampling Mechanism

Given that we have a full posterior distribution which is differentially private, we can use it to define a private mechanism. We may allow the adversary to submit an arbitrary set of queries $\{ q_t \}$ with each $q_t \in \mathcal{Q}$. Each query warrants a response r_t in a set of possible responses $\mathcal{R}$. The adversary is allowed to condition the queries on our previous responses.

We extend our original approach (Dimitrakakis et al., 2014) to take some utility function u into account, which scores preferences of responses given a query. The algorithm requires a prior ξ to be defined on a family $\mathcal{F}_\Theta$ of probability distributions, whose members do not necessarily generate i.i.d. observations. They could be Markov chains for example. The first step is to simply draw a number of samples from the posterior, as in the original approach (Algorithm 2). After the algorithm calculates the posterior distribution $\xi(\cdot \mid x)$, N parameter samples are drawn from it, producing a parameter set $\hat{\Theta}$. Thereafter, responses depend only on the utility function and the sample $\hat{\Theta}$, and we do not draw new samples after every query. This allows us to work with a fixed privacy budget.

Algorithm 1 BAPS: Bayesian Posterior Sampling

input prior ξ, data $x \in \mathcal{S}$
Calculate posterior $\xi(\theta \mid x)$.
for $k = 1, \ldots, N$ **do**
  Sample $\theta^{(k)} \sim \xi(\theta \mid D)$.
end for
return $\hat{\Theta} = \left\{ \theta^{(k)} : k = 1, \ldots, N \right\}$.

Corollary 10 *Algorithm 1 is differentially private under the conditions of Theorem 8, namely:*

1. Under a pseudo-metric ρ and $L > 0$ satisfying Assumption 1, the algorithm is $(2NL, 0)$-differentially private under pseudo-metric ρ; or

2. Under a pseudo-metric ρ and $c > 1$ satisfying Assumption 2, $C_\xi^{\mathcal{F}_\Theta}$ defined in (9), the algorithm is $\left(0, O(N\sqrt{C_\xi^{\mathcal{F}_\Theta}(L_0 + 1/c)})\right)$-differentially private under pseudo-metric $\sqrt{\rho}$.

Proof This follows directly from Theorems 8 and 4 (composition), as the algorithm samples from the posterior distribution, which is differentially private. ■

Utility and optimal responses. We assume the collection of a set of utility functions $\mathcal{U} = \{ u_\theta : \theta \in \Theta \}$, such that the optimal response for a given parameter θ is the one maximising a utility function $u_\theta : \mathcal{Q} \times \mathcal{R} \to [0, 1]$. If we know the true parameter θ, then we should respond to any query q with $r \in \arg\max_r u_\theta(q, r)$. However, since θ is unknown, we must select a method for conveying the required information. In a Bayesian setting, there are three main approaches we could employ. The standard methodology is to maximise

expected utility with respect to the posterior. This corresponds to marginalising out θ, and responding with:

$$r_t \quad \in \quad \arg\max_r \int_\Theta u_\theta(q_t, r) \, \mathrm{d}\xi(\theta \mid x) \ .$$

The second is to use the *maximum a posteriori* value of θ. The final, which we employ here, is to use sampling; *i.e.*, to reply to each query using parameters sampled from the posterior. This allows us to reply to arbitrary queries without compromising privacy, since the most information an adversary could obtain is the set of sampled parameters. By adjusting the number of samples used, we can easily trade off between privacy and utility.

After this we respond to a series of queries. For the t-th received query q_t, the algorithm returns the optimal response over the sampled parameter set $\hat{\Theta}$, in the manner shown in Algorithm 2. Since we allow arbitrary queries, the third party could simply ask for $\hat{\Theta}$ with a suitable choice of the utility function. Then if u is bounded, it is easy to show that the loss due to sampling is bounded.

Algorithm 2 PSAQR: Posterior Sample Query Response

1: **input** Parameter sample $\hat{\Theta}$.
2: **for** $t = 1, \ldots$ **do**
3: Observe query $q_t \in \mathcal{Q}$, perhaps depending on $r_1, \ldots, r_{t-1}$ and $q_1, \ldots, q_{t-1}$.
4: **return** $r_t \in \arg\max_r \sum_{\theta \in \hat{\Theta}} u_\theta(q_t, r)$
5: **end for**

Lemma 11 *The returned responses of the PSAQR mechanism have a utility which is within* $O\left(\sqrt{\ln(1/\delta)/N}\right)$ *of the optimal value with probability at least* $1 - \delta$ *for any* $\delta > 0$.

Now that we have demonstrated bounds on the utility for the algorithm above, we turn to the issue of how utility and privacy can be optimally tuned. First, we try and quantify the amount of samples an adversary needs to distinguish two data sets.

4.3 Distinguishability of Data Sets

In this section, we wish to relate the size of the sample $\hat{\Theta}$ to the amount of information about x that can be obtained by the adversary $\mathscr{A}$. More precisely, we need to bound how well $\mathscr{A}$ can distinguish x from all alternative data sets y. Within the posterior sampling query model, $\mathscr{A}$ has to decide whether $\mathscr{B}$'s posterior is $\xi(\cdot \mid x)$ or $\xi(\cdot \mid y)$. However, he can only do so within some neighbourhood ϵ of the original data. In this section, we bound $\mathscr{A}$'s error in determining the posterior in terms of the number of samples used. This is analogous to the data set-size bounds on queries in interactive models of differential privacy (Dwork et al., 2006), as well as the point of view of privacy as hypothesis testing (Kairouz et al., 2015; Wasserman and Zhou, 2010) where an adversary wishes to distinguish the data set from two alternatives.

For this section, we consider a utility function whose optimal response is $\hat{\Theta}$. This corresponds to the most powerful query possible under the model shown in Algorithm 2.

Then, the adversary needs only to construct the empirical distribution to approximate the posterior up to some sample error. By bounds on the KL divergence between the empirical and actual distributions we can bound his power in terms of how many samples he needs in order to distinguish between x and y.

Due to the sampling model, we first require a finite sample bound on the quality of the empirical distribution. The adversary could attempt to distinguish different posteriors by forming the empirical distribution on any sub-algebra $\mathfrak{S}$.

Lemma 12 *For any* $\delta \in (0, 1)$, *let* $\mathscr{M}$ *be a finite partition of the sample space* $\mathcal{S}$, *of size* $m \leq \log_2 \sqrt{1/\delta}$, *generating the* σ*-algebra* $\mathfrak{S} = \sigma(\mathscr{M})$. *Let* $x_1, \ldots, x_n \sim P$ *be i.i.d. samples from a probability measure* P *on* $\mathcal{S}$, *let* $P_{|\mathfrak{S}}$ *be the restriction of* P *on* $\mathfrak{S}$ *and let* $\hat{P}^n_{|\mathfrak{S}}$ *be the empirical measure on* $\mathfrak{S}$. *Then, with probability at least* $1 - \delta$*:*

$$\left\| \hat{P}^n_{|\mathfrak{S}} - P_{|\mathfrak{S}} \right\|_1 \leq \sqrt{\frac{3}{n} \ln \frac{1}{\delta}} \, . \tag{10}$$

We can combine this bound on the adversary's estimation error with Theorem 7's bound on the KL divergence between posteriors resulting from similar data to obtain a measure of how fine a distinction between data sets the adversary can make after a finite number of draws from the posterior:

Theorem 13 *Under Assumption 1, the adversary can distinguish between data* x, y *with probability* $1 - \delta$ *if:*

$$\rho(x, y) \geq \frac{3}{4Ln} \ln \frac{1}{\delta} \ .$$

Under Assumption 2, this becomes:

$$\rho(x, y) \geq \frac{3}{2n \left(C_\xi^{\mathcal{F}_\Theta} + 2L_0 c^{-1} \right)} \ln \frac{1}{\delta} \ .$$

Consequently, either smoother likelihoods (*i.e.*, decreasing L), or a larger concentration on smoother likelihoods (*i.e.*, increasing c), increases the effort required by the adversary and reduces the sensitivity of the posterior. Note that, unlike the results obtained for differential privacy of the posterior sampling mechanism, these results have the same algebraic form under both assumptions.

4.4 Trading off Utility and Privacy

By construction, in our setting there are three ways with which to tune privacy. The first is the choice of family; the second is the choice of prior; and the third is how many samples N to draw. The choice of family is usually fixed due to other considerations. However, we have the choice of either tuning the prior, so that we can satisfy our assumptions with some suitable constants L or c, or by tuning the number of samples N in the posterior sampling framework.

The following lemma bounds the regret we suffer in terms of utility when the private posterior we use is ξ, in the case where the posterior we would like to use (assuming no privacy constraints) was $\xi^\star$.

Lemma 14 *If our utility is bounded in* $[0,1]$*, the private posterior we use is* ξ*, while the ideal posterior is* $\xi^\star$*, then the regret suffered is bounded by* $2\|\xi - \xi^\star\|_1$*.*

Finally, consider the case where $\mathscr{B}$, being a true Bayesian, is convinced that $\xi^\star$ is the correct prior distribution to use, but needs to use the prior ξ in order to achieve privacy. The following theorem bounds the expected KL divergence between the two resulting posteriors.

Lemma 15 *If* $\forall \theta \in \Theta$*,* $|\ln \xi^\star(\theta)/\xi(\theta)| \leq \eta$ *then the expected KL divergence is*

$$\mathbb{E}_{x\sim\phi^\star} D(\xi^\star(\cdot \mid x) \| \xi(\cdot \mid x)) \leq 2\eta \,,$$

where $\phi^\star$ *is the* $\xi^\star$ *marginal distribution.*

We can now combine Lemmas 11 and 14 with Lemma 15, to obtain the following result:

Corollary 16 *If* $\mathscr{A}$ *has a preferred prior* $\xi^\star$*, while the private prior used by* $\mathscr{B}$ *is* ξ *and it satisfies the conditions of Lemma 15, then the loss of* $\mathscr{A}$ *in terms of the* $\xi^\star$*-expected utility is* $O\left(\eta + \sqrt{\ln(1/\delta)/N}\right)$*, with probability at least* $1-\delta$*.*

Consequently, if $\mathscr{A}$ believes the correct prior should be $\xi^\star$, he can use the private posterior sample to make decisions, incurring a small loss. Finally, we already showed that $\mathscr{A}$ cannot distinguish between data that are closer than $O(1/N)$ with high probability. Hence, in this setting we can tune N to trade off utility and privacy.

The following theorem characterises the link between the choice of prior, the number of samples, privacy and utility directly. This connects several of our results in one place.

Theorem 17 *If, instead of using a non-private prior* $\xi^\star$*, we use a prior* ξ *restricted on* Θ_L *(such that it satisfies Assumption 1 with constant* L*) and generate* N *samples from the posterior, then (a) the sample is* $2LN$*-differentially private and (b) the loss of* $\mathscr{A}$ *in terms of the* $\xi^\star$*-expected utility is* $O\left([1-\xi^\star(\Theta_L)] + \sqrt{\ln(1/\delta)/N}\right)$*, with probability at least* $1-\delta$ *for any* $\delta > 0$*.*

Proof For (a) note that due to composition, N repetitions give $2LN$-differential privacy. For (b), let Θ_L be the support of ξ. Then, because ξ is the restriction of $\xi^\star$ on Θ_L, it holds that:

$$\begin{aligned}\|\xi - \xi^\star\|_1 &= \xi(\Theta_L) - \xi^\star(\Theta_L) + \xi^\star(\Theta \setminus \Theta_L) - \xi(\Theta \setminus \Theta_L) \\ &= 2[1 - \xi^\star(\Theta_L)] \;.\end{aligned}$$

We now just need to couple this with Lemmas 14 and 11 to directly obtain the stated bound on the utility. ∎

In practice, our choice of ξ gives us a base amount of privacy that depends only on L. By keeping ξ fixed and increasing N, we can easily trade off privacy and utility.

Finally, we should note that the adversary could choose any arbitrary estimator ψ to guess x. Section 4.5 below describes how to apply Le Cam's method to obtain matching lower bounds in this case, by defining *data set estimators* as a model for the adversary.

4.5 Lower Bounds

It is possible to apply standard minimax theory to obtain lower bounds on the rate of convergence of the adversary's estimate to the true data. In order to do so, we can for example apply the method due to LeCam (1973), which places lower bounds on the expected distance between an estimator and the true parameter. In order to apply it in our case, we simply replace the parameter space with the data set space.

Le Cam's method assumes the existence of a family of probability measures indexed by some parameter, with the parameter space being equipped with a pseudo-metric. In our setting, we use Le Cam's method in a slightly unorthodox, but very natural manner. Define the family of probability measures on Θ to be:

$$\Xi \triangleq \{\, \xi(\cdot \mid x) : x \in \mathcal{S} \,\},$$

the family of posterior measures in the parameter space, for a specific prior ξ. Consequently, now $\mathcal{S}$ plays the role of the parameter space, while ρ is used as the pseudo-metric. The original family $\mathcal{F}_\Theta$ plays no further role in this construction, other than a way to specify the posterior distributions from the prior.

Now let ψ be an arbitrary estimator of the unknown data x. As in (LeCam, 1973), we extend ρ to subsets of $\mathcal{S}$ via

$$\rho(A,B) \triangleq \inf\{\, \rho(x,y) : x \in A, y \in B \,\} \;, \quad A, B \subset \mathcal{S} \;.$$

Now we can re-state the following well-known lemma for our specific setting.

Lemma 18 (Le Cam's method) *Let* ψ *be an estimator of* x *on* Ξ *taking values in the metric space* $(\mathcal{S}, \rho)$*. Suppose that there are well-separated subsets* $\mathcal{S}_1, \mathcal{S}_2$ *such that* $\rho(\mathcal{S}_1, \mathcal{S}_2) \geq 2\delta$*. Suppose also that* Ξ_1, Ξ_2 *are subsets of* Ξ *such that* $x \in \mathcal{S}_i$ *for* $\xi(\cdot \mid x) \in \Xi_i$*. Then:*

$$\sup_{x\in\mathcal{S}} \mathbb{E}_\xi(\rho(\psi, x) \mid x) \geq \delta \sup_{\xi_i \in co(\Xi_i)} \|\xi_1 \wedge \xi_2\| \;.$$

This lemma has an interesting interpretation in our case. The quantity

$$\mathbb{E}_\xi(\rho(\psi, x) \mid x) = \int_\Theta \rho(\psi(\theta), x) \,\mathrm{d}\xi(\theta \mid x) \;,$$

is the expected distance between the real data x and the guessed data $\psi(\theta)$ when θ is drawn from the posterior distribution. Consequently, it is possible to apply this method directly to obtain results for specific families of posteriors. These would of course be dependent on the family, the prior and the metric. While we shall not engage in this exercise, we point the interested reader to (Yu, 1997), which provides two simple examples with minimax rates of $O(n^{-4/9})$ and $O(n^{-4/5})$.

5. Examples Satisfying our Assumptions

In what follows we study, for different choices of likelihood and corresponding conjugate prior, what constraints can be placed on the prior's concentration to guarantee a desired level of privacy. These case studies closely follow the pattern in differential privacy research

where the main theorem for a new mechanism is a set of sufficient conditions on (*e.g.*, Laplace) noise levels to be introduced to a response in order to guarantee a level ϵ of ϵ-differential privacy.

For exponential families, we have the canonical form $p_\theta(x) = h(x)\exp\left\{\eta_\theta^\top \tau(x) - A(\eta_\theta)\right\}$, where $h(x)$ is the base measure, η_θ is the distribution's natural parameter corresponding to θ, $\tau(x)$ is the distribution's sufficient statistic, and $A(\eta_\theta)$ is its log-partition function. For distributions in this family, under the absolute log-ratio distance, the family of parameters Θ_L of Assumption 2 must satisfy, for all $x, y \in \mathcal{S}$: $\left|\ln\frac{h(x)}{h(y)} + \eta_\theta^\top(\tau(x) - \tau(y))\right| \leq L\rho(x,y)$. If the left-hand side has an amenable form, then we can quantify the set Θ_L for which this requirement holds. Particularly, for distributions where $h(x)$ is constant and $\tau(x)$ is scalar (*e.g.*, Bernoulli, exponential, and Laplace), this requirement simplifies to $\frac{|\tau(x)-\tau(y)|}{\rho(x,y)} \leq \frac{L}{\eta_\theta}$. One can then find the supremum of the left-hand side independent from θ, yielding a simple formula for the feasible L for any θ. For each example, a detailed proof can be found in Appendix B. Note that in the following examples, we are making the conventional assumption in machine learning that data are bounded ($||x|| \leq B$). Also we use $\xi(\theta)\mathbb{1}_{[c_1,c_2]}$ to denote the trimmed density function obtained by setting the density outside $[c_1, c_2]$ to zero and renormalising the density.

We begin with a few simple examples for single observations, that are nevertheless illustrative.

Lemma 19 (Exponential-Exponential conjugate prior) *The exponential distribution $\mathcal{E}xp(x;\theta)$ with a trimmed exponential conjugate prior $\theta \sim \mathcal{E}xp(\theta;\lambda)\mathbb{1}_{[c_1,c_2]}$, $\lambda > 0$, satisfies Assumption 2 with parameter $c = \lambda$, $L_0 = c_1$, $C_\xi^{\mathcal{F}_\Theta} = c_2/\min\left\{c_1 e^{-c_1 B}, c_2 e^{-c_2 B}\right\}$ and metric $\rho(x,y) = |x-y|$.*

Consequently, the trimmed-exponential prior results in a posterior sampling mechanism that is $(0,\delta)$-DP under ρ, with $\delta = \sqrt{\frac{1}{2}C_\xi^{\mathcal{F}_\Theta}(1 + 2c_1 + 1/\lambda)}$. It is also $(0,\delta)$-DP under the classical definition if $x, y \in [0,1]$.

Lemma 20 (Laplace-Exponential conjugate prior) *The distribution Laplace$(x;s,\mu)$ with a trimmed exponential conjugate prior $1/s = \theta \sim Exp(\theta;\lambda)\mathbb{1}_{[c_1,c_2]}$, $\mu \in \mathbb{R}$, $s \geq 1/L$, $\lambda > 0$ satisfies Assumption 2 with parameters $c = \lambda$, $L_0 = c_1$,*

$$C_\xi^{\mathcal{F}_\Theta} = \begin{cases} \frac{c_2}{2\min\left\{\frac{1}{2c_2}, \frac{1}{2c_1}\exp\left(\frac{-B-\mu}{c_1}\right)\right\}} & , \quad x < \mu \\ \frac{c_2}{2\min\left\{\frac{1}{2c_2}, \frac{1}{2c_1}\exp\left(\frac{\mu-B}{c_1}\right)\right\}} & , \quad x \geq \mu \end{cases},$$

and metric $\rho(x,y) = |x-y|$.

It should come as no surprise that the same type of $(0,\delta)$-privacy is achieved for the Laplace distribution with a trimmed exponential prior. Now we move on to an example from which we draw multiple samples.

Lemma 21 (Beta-Binomial conjugate prior) *The Binomial distribution $\mathcal{B}inom(\theta,n)$, with prior $\theta \sim \mathcal{B}eta(\alpha,\beta)$, $\alpha = \beta > 1$ satisfies Assumption 2 for $L_0 = \ln n$, $c = 2^{-2\alpha+1}/B(\alpha)$,*

where $B(\alpha)$ denotes the beta function with parameters $\alpha = \beta$,

$$C_\xi^{\mathcal{F}_\Theta} = B(\alpha)/B\left(\frac{n+2\alpha-1}{2}, \frac{n+2\alpha+1}{2}\right)$$

and metric $\rho(x,y) = \|x-y\|_1$, where $x, y \in \{0,1\}^n$.

This is an example of a conjugate prior pair that is $(0,\delta)$-DP without trimming the prior, with $\delta = \sqrt{\frac{1}{2}C_\xi^{\mathcal{F}_\Theta}(1 + 2\ln n + 2^{2\alpha-1}B(\alpha))}$. Unfortunately, δ is increasing with n, and as Zheng (2015) shows, this result is essentially unimprovable with direct posterior sampling unless the prior is trimmed.

We next present two results on normal distributions.

Lemma 22 (Normal distribution with known mean and unknown variance) *The normal distribution $N(x;\mu,\sigma^2)$ with a trimmed exponential prior $1/\sigma^2 = \theta \sim \mathcal{E}xp(\theta;\lambda)\mathbb{1}_{[c_1,c_2]}$ satisfies Assumption 2 with parameter $c = \frac{2\lambda}{\max\{|\mu|,1\}}$, $L_0 = \frac{c_1 \max\{|\mu|,1\}}{2}$,*

$$C_\xi^{\mathcal{F}_\Theta} = \min\left\{\sqrt{c_2/c_1}\exp\left(\frac{c_1 c_2^2}{2}\right), \exp\left(\frac{c_2^3}{2}\right)\right\}$$

and metric $\rho(x,y) = \left|x^2 - y^2\right| + 2\left|x-y\right|$.

This example is interesting, because privacy is achieved under a rather unusual metric. However, note that the posterior is classically $(0,3\delta)$-DP for data in $[0,1]$.

Unbounded observation spaces are generally a problem for privacy, even for finite parameter spaces, generally because likelihoods become vanishingly small, thus making log likelihood ratios arbitrarily large. However, the following two examples circumvent this problem. In the first example, we consider a general multivariate extension of Lemma 22. In the second we consider the case of discrete Bayesian networks, where privacy depends on the network connectivity and the probability of rare events—we have also considered posterior sampling of networks under complementary conditions, and output perturbation applied to posterior updates, in recent work (Zhang et al., 2016). In these examples, data is usually not i.i.d. (depending on the choice of network or covariance matrix) and the observation space is not a product space.

Lemma 23 (Multivariate normal distribution) *The multivariate normal distribution $N(x;\mu,A^{-1})$ satisfies our Assumption 1 with $L = \frac{1}{2}(\sum_{i=1}^n \lambda_i^2)^{\frac{1}{2}}\max\{1, ||\mu||_2\}$ under metric $\rho(x,y) = ||xx^\top - yy^\top||_F + 2||x-y||_2$. When $\mu = 0$, Assumption 1 is satisfied with $L = \frac{1}{2}(\sum_{i=1}^n \lambda_i^2)^{\frac{1}{2}}$ under metric $\rho(x,y) = ||(xx^\top - yy^\top)||_F$.*

Once more, we achieved $(\epsilon,0)$-DP under our metric, which implies a $(3\epsilon,0)$ classical DP for bounded data.

Lemma 24 (Discrete Bayesian networks) *Consider a family of discrete Bayesian networks on K variables, $\mathcal{F}_\Theta = \{P_\theta : \theta \in \Theta\}$. More specifically, each member P_θ, is a distribution on a finite space $\mathcal{S} = \prod_{k=1}^K \mathcal{S}_k$ and we write $P_\theta(x)$ for the probability of any outcome*

$x = (x_1, \ldots, x_K)$ in $\mathcal{S}$. Let $\varepsilon \triangleq \min_{\theta, x_k, x_{\mathcal{P}(k)}} P_\theta(x_k \mid x_{\mathcal{P}(k)})$, be the smallest conditional probability in the graph, where $\mathcal{P}(k)$ are the parents of node k.

Our observations can be independent samples $\{ x^t : t \in [T] \}$ of dependent variables $x_1^t, \ldots, x_k^t$. Define the connectivity vector $v \in \mathbb{N}^K$ such that $v_k = 1 + \deg(k)$, where $\deg(k)$ is the out-degree of node K. We now define the distance between two data sets x, y to be

$$\rho(x, y) \triangleq v^\top \delta(x, y), \qquad \delta_k(x, y) \triangleq \sum_{t=1}^{T} \mathbb{I}\left\{x_{k,t} \neq y_{k,t}\right\}.$$

Then Assumption 1 is satisfied with $L = \ln 1/\varepsilon$.

Consequently, discrete Bayesian networks, endowed with any prior on the family given in the above example, are $(2 \ln 1/\varepsilon, 0)$-DP under ρ. This also implies that they are $2\|v\|_\infty \ln 1/\varepsilon$-DP under the classical definition.

A simple application of this example is to data drawn from a Markov model on a finite state space. In particular, consider a time-homogeneous family of transition matrices $\theta_{i,j} \triangleq P_\theta(x_{t+1} = i \mid x_t = j)$. Then a prior consisting of product of truncated Dirichlet distributions that bound all multinomial probabilities above ε satisfies our assumptions and results in a $4 \ln 1/\varepsilon$-DP mechanism.

The above examples demonstrate that our assumptions are reasonable. In fact, for several of them we recover standard choices of prior distributions. However, for the privacy guarantees to be reasonable, it is best to restrict the prior to a set of parameters that is not very sensitive.

6. Discussion

We have presented a unifying framework for private and secure inference in a Bayesian setting. Under concentration conditions on the prior, we have shown that Bayesian inference is both robust and private. Firstly, we prove that similar data sets result in posterior distributions with small KL divergence. Secondly, we establish that the posterior is differentially private. This allows us to use a general posterior sampling mechanism for responding to queries, where privacy and utility are easy to trade off by adjusting the number of samples taken.

Owing to the fact that no additional machinery is required, this framework may serve as a fundamental building block for more sophisticated, private Bayesian inference. As an additional step towards this goal, we have demonstrated the application of our framework to deriving analytical expressions for well-known distribution families, and for discrete Bayesian networks. Finally, we bounded the amount of effort required of an attacker to breach privacy when observing samples from the posterior. This serves as a principled guide for how much access can be granted to querying the posterior, while still guaranteeing privacy.

Conversion of our results to the neighbourhood formulation. We state most of our results on specific models using a distance based on a sufficient statistic. Hence, to convert these to standard differential privacy, we only need to bound the ρ-distance of any neighbouring data sets. A good example are DBNs, where the case $\rho(x, y) = 1$ corresponds exactly to that of one record changing in a databse.

Practical application of our results. In general, it is hard to verify whether an existing model family will satisfy DP, because it implies checking whether the log-likelihood function is Lipschitz. Some parametric conjugate families, like the ones we examined in the examples, are amenable to analytic treatment. In practice, though, this might not be possible. It is for this reason that we propose to use rejection sampling in order to sample from the truncated posterior distribution. In particular, it is possible to resample from the posterior distribution, until a sample within the allowed interval of parameters is obtained. This is an approach we recently used in an application paper successfully (Zhang et al., 2016).

6.1 Related Work

In the past, little research in differential privacy focused on the Bayesian paradigm, with Dimitrakakis et al. (2014) being the first to establish conditions for differentially-private Bayesian inference. Nevertheless, our paper has many interesting links with both previous and follow up work, with respect to differential privacy, robustness and Bayesian inference, which we outline below. First, we discuss relations to other mechanisms achieving differential privacy and theoretical works about differential privacy; secondly, we discuss related work on the connection between robustness and privacy; and we conclude the related work section with a discussion of previous versions of this paper and follow-up work.

6.1.1 Differential Privacy

In our paper, we employ a Bayesian framework whereby optimal responses are characterised by the fact that they maximise expected utility. In Bayesian statistical decision theory (Berger, 1985; Bickel and Doksum, 2001; DeGroot, 1970), learning is cast as a statistical inference problem and decision-theoretic criteria are used as a basis for assessing, selecting and designing procedures. In particular, for a given utility function, the Bayes-optimal procedure maximises the expected utility under the posterior distribution.

In our setting, however, decisions using the data are not taken by the statistician $\mathscr{B}$. Instead, $\mathscr{A}$ provides a utility function, and trusts $\mathscr{B}$ to give him responses to queries that maximise expected utility. However $\mathscr{B}$ must also balance the need for privacy of the data provider, which results in some utility loss for $\mathscr{A}$. This is naturally captured by the difference in utility by making the decision private. This idea had already been explored in the exponential mechanism by McSherry and Talwar (2007), which connected differential privacy to mechanism design.

The exponential mechanism can be seen as a generalisation of the *Laplace mechanism*, which adds Laplace noise to released statistics (Dwork et al., 2006). The exponential mechanism releases a response with probability exponential in a utility function describing the usefulness of each response, with the best response having maximal utility. An alternate approach, employed for privatising regularised empirical-risk minimisation (Chaudhuri et al., 2011), is to alter the inferential procedure itself, in that case by adding a random term to the primal objective. We view our posterior sampling mechanism as a Bayesian counterpart. Further results on the accuracy of the exponential mechanism with respect to the Kolmogorov-Smirnov distance are given in (Wasserman and Zhou, 2010), which introduced the concept of privacy as hypothesis testing where an adversary wishes to distinguish two data sets. This is similar to our notion of data set distinguishability.

Learning from private data. In a different direction, Duchi et al. (2013) provided information-theoretic bounds for private learning. This essentially represents the protocol for interacting with an adversary as an arbitrary conditional distribution, rather than restricting it to specific mechanisms or models. In this way, they obtain fundamental bounds on rates of convergence from differentially-private views of data.

Bayesian inference and privacy. Other work at the intersection of privacy and Bayesian inference includes that of Williams and McSherry (2010) who applied Bayesian inference to improve the utility of differentially-private releases by computing posteriors in a noisy measurement model. In a similar vein, Xiao and Xiong (2012) used Bayesian credible intervals to respond to queries with as high utility as possible, subject to a privacy budget. In the PAC-Bayesian setting, Mir (2012) showed that the Gibbs estimator (McSherry and Talwar, 2007) is differentially private. While their algorithm corresponds to a posterior sampling mechanism, it is a posterior found by minimising risk bounds; by contrast, our results are purely Bayesian and come from conditions on the prior. It is also worthwhile noting that our Assumption 1 can in some cases be made equivalent to the definition of Pufferfish privacy (Kifer and Machanavajjhala, 2014), a privacy concept with Bayesian semantics. Thus, our results imply that in some cases Pufferfish privacy also results in differential privacy. Finally, independently to our preliminary work (Dimitrakakis et al., 2014), Wang et al. (2015) later proved differential privacy results for Gaussian processes under similar assumptions.

6.1.2 Robustness and Privacy

Dwork and Lei (2009) made the first connection between (frequentist) robust statistics and differential privacy, developing mechanisms for the interquartile, median and B-robust regression. While robust statistics are designed to operate near an ideal distribution, they can have prohibitively high global, worst-case sensitivity. In this case privacy was still achieved by performing a differentially-private test on local sensitivity before release (Dwork and Smith, 2009). In later work, Dwork et al. (2015) show that differentially-private views of the data result in good generalisation abilities. We discuss this more extensively in Section 6.1.3.

In a similar vein Chaudhuri and Hsu (2012) drew a quantitative connection between robust statistics and differential privacy by providing finite-sample convergence rates for differentially-private plug-in statistical estimators in terms of the *gross error sensitivity*, a common measure of robustness. These bounds can be seen as complementary to ours because our Bayesian estimators do not have private views of the data but use a suitably-defined prior instead.

Smoothness of the learning map, achieved here for Bayesian inference by appropriate concentration of the prior, is related to *algorithmic stability* which is used in statistical learning theory to establish error rates (Bousquet and Elisseeff, 2002). Rubinstein et al. (2012) used γ-uniform stability to calibrate the level of noise when using the Laplace mechanism to achieve differential privacy for the SVM. Hall et al. (2013) extended this technique to adding Gaussian process noise for differentially private release of infinite-dimensional functions lying in an RKHS.

In the Bayesian setting, robustness is typically handled through maximin policies. This is done by assuming that the prior distribution is selected arbitrarily by nature. In the field of robust statistics, the minimax asymptotic bias of a procedure incurred within an ε-contamination neighbourhood is used as a robustness criterion giving rise to the notions of a procedure's *influence function* and *breakdown point* to characterise robustness (Hampel et al., 1986; Huber, 1981). In a Bayesian context, robustness appears in several guises including minimax risk, robustness of the posterior within ε-contamination neighbourhoods, and robust priors (Berger, 1985). In this context Grünwald and Dawid (2004) demonstrated the link between robustness in terms of the minimax expected score of the likelihood function and the (generalised) maximum entropy principle, whereby nature is allowed to select a worst-case prior.

6.1.3 Previous Versions and Follow Up Work

Finally, we note that preliminary versions of this work appeared on arXiv (Dimitrakakis et al., 2013. Latest version 2015.) and ALT (Dimitrakakis et al., 2014). This version corrects technical issues with one proof, which affected the leading constants. We also replaced the original mechanism with one taking a fixed sample, which allows us to maintain a fixed privacy budget for an arbitrary number of queries. We make a novel use of Le Cam's method to prove lower bounds on indistinguishability, and we complement our original bounds with bounds for the utility of the mechanism. Finally, we discuss the relationship between posterior sampling, the exponential mechanism and the *safe Bayesian* generalisation of Bayesian inference. Follow-up work includes: Wang et al. (2015) who, under similar assumptions proved differential privacy results for Gibbs samplers; Zheng (2015) who improved some of our original bounds and also presented new results for other members of the exponential family; and Zhang et al. (2016) who recently initiated the exploration of the posterior sampler in probabilistic graphical models on multiple random variables.

Another important follow up work is that of Dwork et al. (2015). They have shown that *any* differentially private algorithm results in robustness, in the sense that the divergence between posterior distribution arising from similar data is small. This has a direct impact on the generalisation ability of statistical models and inferences drawn, and consequently allows for what they call the "re-usable hold-out". In our work, on the other hand, we have shown that with the right choice of prior, Bayesian inference is both private and robust. We have also shown that if the posterior distribution is robust, then it is also differentially private. In conclusion, robustness and privacy appear to be deeply linked, as our works have jointly shown conditions when one implies the other in three different ways: not only the same sufficient conditions can achieve both privacy and robustness, but privacy can also imply robustness, and robustness implies privacy. Further links between the two concepts are likely, as explained in the next section.

6.2 Future Directions

Although we have shown how Bayesian inference can already be differentially private by appropriately setting the prior, we have not examined how this affects learning. While larger c improves privacy, it also concentrates the prior so much that learning would be

inhibited. Thus, c could be chosen to optimise the trade-off between privacy and learning. However, we believe that the choice of the number of samples is easier to control.

From the theoretical side, we believe that the constant $C_\xi^{\mathcal{F}_\Theta}$ could be substantially improved, since right now it seems to be rather loose. It is also possible that its existence is only an artefact of the analysis, since it only appears for Assumption 2. However, we thought it crucial to include the results from this assumption in the paper, since they are connected to the second necessary condition. Hopefully, future work will uncover improved bounds for Assumption 2, or a similar condition to it.

Other future directions include investigating the links between posterior sampling and the exponential mechanism, as well as with the safe Bayesian approach (Grünwald, 2012) to inference. Consider an exponential mechanism which, given a utility function $u : \Theta \times \mathcal{Q} \to \mathbb{R}$ and a base measure μ on Θ returns $\theta \in \Theta$ sampled from the density

$$f(\theta) \propto e^{\epsilon u(\theta,q)} \frac{\mathrm{d}\mu(\theta)}{\mathrm{d}\lambda} \ .$$

As also noted by Wang et al. (2015), this has a similar form to the posterior distribution, by setting $u(\theta, q) = \ln p_\theta(x)$ and setting $\mu = \xi$ to the prior. This idea was used independently by Zhang et al. (2016) for releasing MAP point estimates. In this framework, privacy is achieved by setting ϵ to a sufficiently small value. However, it is interesting to note that this is how Grünwald (2012) obtains robustness results for modified Bayesian inference. This implies that in some cases we can gain both privacy and efficiency. We note that in our case, we have proven that privacy is attainable by altering the prior, which corresponds to the base measure in the exponential mechanism. Consequently, we believe it is worthwhile examining settings where adjusting both ϵ and the prior measure may be advantageous.

Acknowledgments

We gratefully thank Aaron Roth, Kamalika Chaudhuri, and Matthias Bussas for their discussion and insights as well as the anonymous reviewers for their comments on the paper, which helped to improve it significantly. This work was partially supported by the Marie Curie Project "Efficient Sequential Decision Making Under Uncertainty", Grant Number 237816; the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme (FP7/2007-2013) under REA grant agreement n 608743; the SNSF Project, "SwissSenseSynergia"; and the Australian Research Council (DE160100584).

Appendix A. Proofs of Main Results

Proof of Lemma 5 For Assumption 1, the proof follows directly from the definition of the absolute log-ratio distance; namely,

$$\begin{aligned} |\ln p_\theta^n(\{x_i\}) - \ln p_\theta^n(\{y_i\})| &\leq \textstyle\sum_{i=1}^n |\ln p_\theta(x_i) - \ln p_\theta(y_i)| \\ &\leq L \textstyle\sum_{i=1}^n \rho(x_i, y_i) \ . \end{aligned}$$

For Assumption 2, consider sub-family Θ_L from Eq. (4) for marginal p_θ and pseudo-metric ρ, and define the corresponding sub-family Θ_L^n in terms of product distribution p_θ^n

and pseudo-metric ρ^n. Then the same argument as above shows that $\Theta_L \subseteq \Theta_L^n$. Hence, the same prior and parameter c yield the lower bound of Eq. (5), for Θ_L^n. ■

Proof of Theorem 7 Let us now tackle claim 1. First, we can decompose the KL-divergence into two parts.

$$\begin{aligned} D\left(\xi(\cdot \mid x) \parallel \xi(\cdot \mid y)\right) &= \int_\Theta \ln \frac{\mathrm{d}\xi(\theta \mid x)}{\mathrm{d}\xi(\theta \mid y)} \, \mathrm{d}\xi(\theta \mid x) \\ &= \int_\Theta \ln \frac{p_\theta(x)}{p_\theta(y)} \, \mathrm{d}\xi(\theta \mid x) + \int_\Theta \ln \frac{\phi(y)}{\phi(x)} \, \mathrm{d}\xi(\theta \mid x) \\ &\leq \int_\Theta \left| \ln \frac{p_\theta(x)}{p_\theta(y)} \right| \mathrm{d}\xi(\theta \mid x) + \int_\Theta \ln \frac{\phi(y)}{\phi(x)} \, \mathrm{d}\xi(\theta \mid x) \\ &\leq L\rho(x,y) + \left| \ln \frac{\phi(y)}{\phi(x)} \right| \ . \end{aligned} \tag{11}$$

From Assumption 1, $p_\theta(y) \leq \exp(L\rho(x,y)) p_\theta(x)$ for all θ so:

$$\begin{aligned} \phi(y) &= \int_\Theta p_\theta(y) \, \mathrm{d}\xi(\theta) \\ &\leq \exp(L\rho(x,y)) \int_\Theta p_\theta(x) \, \mathrm{d}\xi(\theta) = \exp(L\rho(x,y))\phi(x) \ . \end{aligned}$$

Combining this with (11) we obtain

$$D\left(\xi(\cdot \mid x) \parallel \xi(\cdot \mid y)\right) \leq 2L\rho(x,y) \ .$$

Claim 2 is dealt with similarly. Once more, we can break down the distance in parts. In more detail, we first write:

$$D\left(\xi(\cdot \mid x) \parallel \xi(\cdot \mid y)\right) \leq \underbrace{\int_\Theta \left| \ln \frac{p_\theta(x)}{p_\theta(y)} \right| \mathrm{d}\xi(\theta \mid x)}_{A} + \underbrace{\int_\Theta \ln \frac{\phi(y)}{\phi(x)} \, \mathrm{d}\xi(\theta \mid x)}_{B} \ ,$$

as before. Now, let us re-write the A term as

$$\int_\Theta \left| \ln \frac{p_\theta(x)}{p_\theta(y)} \right| \frac{p_\theta(x)}{\phi(x)} \, \mathrm{d}\xi(\theta) \leq \sup_{\theta'} \frac{p_{\theta'}(x)}{\phi(x)} \int_\Theta \left| \ln \frac{p_\theta(x)}{p_\theta(y)} \right| \mathrm{d}\xi(\theta) \ ,$$

so that the left-hand side term is the ratio between the maximal likelihood and marginal likelihood. Using the same steps, we can bound B in the same manner.

Now, let us define a data-dependent and a data-independent bound:

$$C_\xi^{\mathcal{F}_\Theta}(x) \triangleq \sup_\theta \frac{p_\theta(x)}{\phi(x)} \ , \qquad C_\xi^{\mathcal{F}_\Theta} \triangleq \sup_x C_\xi^{\mathcal{F}_\Theta}(x) \ .$$

Replacing, we obtain:

$$D\left(\xi(\cdot \mid x) \parallel \xi(\cdot \mid y)\right) \leq C_\xi^{\mathcal{F}_\Theta} \underbrace{\int_\Theta \left| \ln \frac{p_\theta(x)}{p_\theta(y)} \right| \mathrm{d}\xi(\theta)}_{A} + \underbrace{\int_\Theta \ln \frac{\phi(y)}{\phi(x)} \, \mathrm{d}\xi(\theta \mid x)}_{B} \ .$$

Now, to bound the individual terms, we start from A and note that theorem 3 of (Norkin, 1986) on the Lipschitz property of the expectation of stochastic Lipschitz functions applies.

Theorem 25 *(Norkin, 1986) If ξ is a probability measure on Θ and $f : \mathcal{S} \times \Theta \to \mathbb{R}$ is a ξ-measurable function, such that for any $\theta \in \Theta$, $f(\cdot, \theta)$ is $\ell(\theta)$-Lipschitz, then the function $f_\xi(x) \triangleq \mathbb{E}_\xi f(x, \theta)$ is L_ξ-Lipschitz, where $L_\xi = \mathbb{E}_\xi \ell(\theta)$.*

Recall that the expectation of a non-negative random variable can be written in terms of its CDF F as $\int_0^\infty [1 - F(t)]\,\mathrm{d}t$. In our case, $\ell(\theta)$ is a random variable on Θ, and we can write its cumulative distribution function as

$$F(t) \triangleq \xi\left(\{\,\theta \in \Theta : \ell(\theta) \le t\,\}\right) = \xi(\Theta_t) \ ,$$

by the definition of Θ_t. It follows that $\ln p_\theta(x)$ is L_ξ-Lipschitz, where through the formula for the expectation of positive variables:

$$L_\xi = \int_0^\infty [1 - \xi(\Theta_t)]\,\mathrm{d}t \le L_0 \xi(\Theta_{L_0}) + [1 - \xi(\Theta_{L_0})] \int_0^\infty e^{-ct}\,\mathrm{d}t \le L_0 + c^{-1} \ . \tag{12}$$

So, term A becomes $C_\xi^{\mathcal{F}_\Theta}\left(L_0 + c^{-1}\right)\rho(x, y)$.

Now let us move on to term B. For technical reasons, we start by considering a pair x, y such that $\rho(x, y) \le c - 1$. This also implies that $c > 1$, since the distance cannot be negative.

$$\frac{\phi(x)}{\phi(y)} \overset{(a)}{=} \int_\Theta \frac{p_\theta(x)}{\phi(y)}\,\mathrm{d}\xi(\theta) \overset{(b)}{\le} \int_\Theta \frac{p_\theta(y) e^{\ell(\theta)\rho(x,y)}}{\phi(y)}\,\mathrm{d}\xi(\theta) \overset{(c)}{\le} C_\xi^{\mathcal{F}_\Theta} \int_\Theta e^{\ell(\theta)\rho(x,y)}\,\mathrm{d}\xi(\theta) \ . \tag{13}$$

Note that $\left\{\,\theta \in \Theta : e^{\ell(\theta)\rho(x,y)} \le t\,\right\} = \left\{\,\theta \in \Theta : \ell(\theta) \le \rho(x, y)^{-1} \ln t\,\right\} = \Theta_{\rho(x,y)^{-1}\ln t}$. So the CDF of the random variable $e^{\ell(\theta)}$ is $F(t) = \xi(\Theta_{\rho(x,y)^{-1}\ln t})$. Then:

For positive random variables, $\mathbb{E}\, X^\rho = \rho \int_0^\infty t^{\rho-1}[1 - F(t)]dt$. Applying this to our case, we get:

$$\begin{aligned}
\mathbb{E}_\xi\, e^{\ell(\theta)\rho(x,y)} &= \mathbb{E}_\xi[e^{\ell(\theta)\rho(x,y)} \mid \ell \le L_0]\xi(\Theta_{L_0}) + \mathbb{E}_\xi[e^{\ell(\theta)\rho(x,y)} \mid \ell > L_0][1 - \xi(\Theta_{L_0})] \\
&\le e^{L_0\rho(x,y)} + \rho(x, y) \int_{t_0}^\infty t^{\rho(x,y)-1}[1 - \xi(\Theta_{\ln t})]\,\mathrm{d}t \\
&\le e^{L_0\rho(x,y)} + \rho(x, y) \int_{t_0}^\infty e^{\ln t[\rho(x,y)-1]} e^{-c(\ln t - L_0)}\,\mathrm{d}t \qquad \text{(where } t_0 = e^{L_0}\text{)} \\
&= e^{L_0\rho(x,y)} + \rho(x, y) \int_{t_0}^\infty e^{\ln t[\rho(x,y)-c-1]+cL_0}\,\mathrm{d}t \\
&= e^{L_0\rho(x,y)} + \rho(x, y) e^{cL_0} \int_{t_0}^\infty t^{\rho(x,y)-c-1}\,\mathrm{d}t \\
&= e^{L_0\rho(x,y)} + \rho(x, y) e^{cL_0} \frac{t_0^{\rho(x,y)-c}}{c - \rho(x, y)} \\
&= e^{L_0\rho(x,y)} + \rho(x, y) e^{cL_0} \frac{e^{L_0(\rho(x,y)-c)}}{c - \rho(x, y)} \\
&\le e^{L_0\rho(x,y)} + \rho(x, y) e^{cL_0} e^{L_0(\rho(x,y)-c)} \\
&= e^{L_0\rho(x,y)} + \rho(x, y) e^{L_0\rho(x,y)} = (1 + \rho(x, y)) e^{L_0\rho(x,y)} \le e^{(1+L_0)\rho(x,y)}.
\end{aligned}$$

Consequently, $\ln \phi(x)/\phi(y) \le C_\xi^{\mathcal{F}_\Theta}(1 + L_0)\rho(x, y)$.

To handle larger distances ρ, we can simply apply the above result repeatedly between k data sets $z_1, \ldots, z_k$, where $z_1 = x$, $z_k = y$ and such that $\rho(z_i, z_{i+1}) < c - 1$. By chaining logarithmic ratios, *i.e.*, using the fact that $\ln \phi(x)/\phi(y) = \ln \phi(x)/\phi(z) + \ln \phi(z)/\phi(y)$ we can now extend our result to general pairs for term B. Replacing those terms, we obtain:

$$D\left(\xi(\cdot \mid x) \parallel \xi(\cdot \mid y)\right) \le C_\xi^{\mathcal{F}_\Theta}\left(1 + 2L_0 + c^{-1}\right)\rho(x, y) \ .$$

If the intermediate points do not exist under ρ, we can simply scale it by $\chi \le 1$, thus obtaining the final result. ■

Proof of Theorem 8 For part 1, we assumed that there is an $L > 0$ such that $\forall x, y \in \mathcal{S}$, $\left|\log \frac{p_\theta(x)}{p_\theta(y)}\right| \le L\rho(x, y)$, thus implying $\frac{p_\theta(x)}{p_\theta(y)} \le \exp\{L\rho(x, y)\}$. Further, in the proof of Theorem 7, we showed that $\phi(y) \le \exp\{L\rho(x, y)\}\phi(x)$ for all $x, y \in \mathcal{S}$. From Eq. (1), we can then combine these to bound the posterior of any $B \in \mathfrak{S}_\Theta$ as follows for all $x, y \in \mathcal{S}$:

$$\xi(B \mid x) = \frac{\int_B \frac{p_\theta(x)}{p_\theta(y)} p_\theta(y)\,\mathrm{d}\xi(\theta)}{\phi(y)} \cdot \frac{\phi(y)}{\phi(x)} \le \exp\{2L\rho(x, y)\}\xi(B \mid y) \ .$$

For part 2, note that from Theorem 7 part 2 that the KL divergence of the posteriors under assumption is bounded by (8). Now, recall Pinsker's inequality (cf. Fedotov et al., 2003):

$$D\left(Q \| P\right) \ge \frac{1}{2}\|Q - P\|_1^2 = 2\|Q - P\|_{\mathrm{TV}}^2 \triangleq 2 \sup_B |Q(B) - P(B)|^2 \tag{14}$$

This yields: $|\xi(B \mid x) - \xi(B \mid y)| \le \sqrt{\frac{1}{2} D\left(\xi(\cdot \mid x) \parallel \xi(\cdot \mid y)\right)} \le \sqrt{\frac{1}{2} C_\xi^{\mathcal{F}_\Theta}\left(1 + 2L_0 c^{-1}\right)\rho(x, y)}$. ■

Proof of Lemma 11 Sampling N times from the posterior, gives us the following estimate of the utility function $\hat{u_\xi}(q, r) = \frac{1}{N}\sum_{\theta \in \hat{\Theta}} u_\theta(q, r)$, which with probability at least $1 - \delta$ satisfies $|\hat{u_\xi}(q, r) - u(q, r)| < \sqrt{\frac{\ln(2/\delta)}{2N}} = \epsilon$, $\forall r, q$, via Hoeffding's inequality and the boundedness of u. Consequently, we can be at most 2ϵ-away from the optimal. ■

Proof of Lemma 12 (Note that in this proof, ε, δ do not refer to the privacy parameters.) We use the inequality due to Weissman et al. (2003) on the ℓ_1 norm, which states that for any multinomial distribution P with m outcomes, the ℓ_1 deviation of the empirical distribution $\hat{P}_n$ after n draws from the multinomial satisfies:

$$\mathbb{P}\left(\left\|\hat{P}_n - P\right\|_1 \ge \varepsilon\right) \le (2^m - 2)e^{-\frac{1}{2}n\varepsilon^2}, \qquad \forall \varepsilon > 0 \ .$$

The right hand side is bounded by $e^{m\ln 2-\frac{1}{2}n\varepsilon^2}$. Substituting $\varepsilon=\sqrt{\frac{3}{n}\ln\frac{1}{\delta}}$:

$$\begin{aligned}\mathbb{P}\left(\left\|\hat{P}_n-P\right\|_1\geq\sqrt{\frac{3}{n}\ln\frac{1}{\delta}}\right)&\leq e^{m\ln 2-\frac{3}{2}\ln\frac{1}{\delta}}\\&\leq e^{\log_2\sqrt{\frac{1}{\delta}}\ln 2-\frac{3}{2}\ln\frac{1}{\delta}}\\&=e^{\frac{1}{2}\ln\frac{1}{\delta}-\frac{3}{2}\ln\frac{1}{\delta}}\\&=\delta\ .\end{aligned}$$

where the second inequality follows from $m\leq\log_2\sqrt{1/\delta}$. ■

Proof of Theorem 13 Recall that the data processing inequality states that, for any sub-algebra $\mathfrak{S}$:

$$\left\|Q_{|\mathfrak{S}}-P_{|\mathfrak{S}}\right\|_1\leq\|Q-P\|_1\ .$$

Using this and Pinsker's inequality (14) we obtain:

$$\begin{aligned}2L\rho(x,y)&\geq D\left(\xi(\cdot\mid x)\|\xi(\cdot\mid y)\right)\\&\geq\frac{1}{2}\left\|\xi(\cdot\mid x)-\xi(\cdot\mid y)\right\|_1^2\\&\geq\frac{1}{2}\left\|\xi_{|\mathfrak{S}}(\cdot\mid x)-\xi_{|\mathfrak{S}}(\cdot\mid y)\right\|_1^2\ .\end{aligned}$$

On the other hand, due to (10) the adversary's ℓ_1 error in the posterior distribution is bounded by $\sqrt{\frac{3}{n}\ln\frac{1}{\delta}}$ with probability $1-\delta$. In order for him to be able to distinguish the two different posteriors, it must hold that

$$\left\|\xi_{|\mathfrak{S}}(\cdot\mid x)-\xi_{|\mathfrak{S}}(\cdot\mid y)\right\|_1\geq\sqrt{\frac{3}{n}\ln\frac{1}{\delta}}\ .$$

Using the above inequalities, we can bound the error in terms of the distinguishability of the real data set x from an arbitrary set y as:

$$4L\rho(x,y)\geq\frac{3}{n}\ln\frac{1}{\delta}\ .$$

Rearranging, we obtain the required result. The second case is treated similarly to obtain:

$$2C_\xi^{\mathcal{F}_\Theta}\left(1+2L_0+c^{-1}\right)\rho(x,y)\geq\frac{3}{n}\ln\frac{1}{\delta}\ .$$

■

Proof of Lemma 14 Let $r,r^\star$ be the optimal responses under $\xi,\xi^\star$ respectively. For notational convenience, let $u_\xi=\int_\Theta u_\theta\,\mathrm{d}\xi(\theta)$ denote the expected utility under a belief ξ.

Then our regret is

$$\begin{aligned}u_\xi(q,r)-u_\xi(q,r^\star)&=u_\xi(q,r)-u_{\xi^\star}(q,r)\\&\quad+u_{\xi^\star}(q,r)-u_{\xi^\star}(q,r^\star)\\&\quad+u_{\xi^\star}(q,r^\star)-u_\xi(q,r^\star)\\&\leq 2\left\|\xi-\xi^\star\right\|_1\ .\end{aligned}$$

This follows from the fact that

$$\begin{aligned}u_\xi(q,r)-u_{\xi^\star}(q,r)&=\int_\Theta u_\theta(q,r)\,\mathrm{d}[\xi-\xi^\star](\theta)\\&\leq\|u\|_\infty\|\xi-\xi^\star\|_1\end{aligned}$$

and then using the boundedness of u. The third term is dealt with identically. For the second term, note that $u_{\xi^\star}(q,r)-u_{\xi^\star}(q,r^\star)\leq 0$ since $r^\star$ maximises $u_{\xi^\star}$. ■

Proof of Lemma 15 Let $\phi^\star(x)=\int_\Theta p_\theta(x)\,\mathrm{d}\xi^\star(x)$ be the prior marginal distribution. Then the $\xi^\star$-expected KL divergence between the two posteriors is

$$\begin{aligned}&\sum_x\int_\Theta\ln\frac{\mathrm{d}\xi^\star(\theta\mid x)}{\mathrm{d}\xi(\theta\mid x)}\,\mathrm{d}\xi^\star(\theta\mid x)\phi^\star(x)\\&\leq\sum_x\int_\Theta\left(\left|\ln\frac{\mathrm{d}\xi^\star(\theta)}{\mathrm{d}\xi(\theta)}\right|+\left|\ln\frac{\phi(x)}{\phi^\star(x)}\right|\right)\mathrm{d}\xi^\star(\theta\mid x)\phi^\star(x)\\&\leq 2\eta\ .\end{aligned}$$

The first term $\left|\ln\frac{\mathrm{d}\xi^\star(\theta)}{\mathrm{d}\xi(\theta)}\right|$ is bounded by η by assumption. From the same assumption, it follows that $\phi(x)=\int_\Theta p_\theta(x)\,\mathrm{d}\xi(\theta)\leq\int_\Theta p_\theta(x)e^\eta\,\mathrm{d}\xi^\star(\theta)=e^\eta\phi^\star(x)$, and so the second term is also bounded by η. ■

Appendix B. Proofs of Examples

Proof of Lemma 19 Since $Exp(x;\theta)$ is monotonic decreasing in x and concave as a function of θ, we have $\inf_{\{||x||\leq B,\theta\in[c_1,c_2]\}}Exp(x;\theta)=\min\left\{c_1e^{-c_1B},c_2e^{-c_2B}\right\}\leq\phi(x)$. Then we have

$$C_\xi^{\mathcal{F}_\Theta}=c_2/\min\left\{c_1e^{-c_1B},c_2e^{-c_2B}\right\}\ .$$

Next we compute the absolute log-ratio distance for any x_1 and x_2 according to the exponential likelihood function:

$$|\ln p_\theta(x_1)-\ln p_\theta(x_2)|=\theta|x_1-x_2|\ .$$

Thus, for $\theta\in[c_1,c_2]$, under Assumption 2, using $\rho(x,y)=|x-y|$, the set of feasible parameters for any $L>c_1$ is $\Theta_L=(c_1,L)$. Note the density of the renormalized exponential prior on $[c_1,c_2]$ is given by $K\lambda e^{-\lambda\theta}$, where $K=(e^{-\lambda c_1}-e^{-\lambda c_2})^{-1}$. Thus the CDF at L of

this density is $K\left(e^{-\lambda c_1} - e^{-\lambda L}\right)$ for $L \in [c_1, c_2]$ and 1 for $L \geq c_2$. It is natural to choose L_0 to be c_1. Then we need to find c such that

$$\xi(\Theta_L) = \int_{c_1}^{L} K\lambda e^{-\lambda\theta} d\theta = K(e^{-\lambda c_1} - e^{-\lambda L}) \geq 1 - e^{-c(L-c_1)}$$

for $L \in (c_1, c_2)$. By plugging K into the inequality, we have

$$e^{-c(L-c_1)} \geq \frac{e^{-\lambda(L-c_2)} - 1}{e^{-\lambda(c_1-c_2)} - 1} \ .$$

Since $e^{-\lambda(L-c_2)} \leq e^{-\lambda(c_1-c_2)}$, it is sufficiency to find c such that $e^{-c(L-c_1)} \geq e^{-\lambda(L-c_1)}$. Therefore we can have $c = \lambda$. ■

Proof of Lemma 20 Note that $Laplace(x; s, \mu)$ is monotonic decreasing in x if $x < \mu$, and increasing in x if $x \geq \mu$. Since $Laplace(x; s, \mu)$ is concave as a function of s, we have $\phi(t) \geq \min\left\{\frac{1}{2c_2}, \frac{1}{2c_1}\exp\left(\frac{-B-\mu}{c_1}\right)\right\}$ if $x < \mu$ and $\phi(t) \geq \min\left\{\frac{1}{2c_2}, \frac{1}{2c_1}\exp\left(\frac{\mu-B}{c_1}\right)\right\}$ if $x \geq \mu$. Thus, we can take

$$C_\xi^{\mathcal{F}_\Theta} = \begin{cases} \frac{c_2}{2\min\left\{\frac{1}{2c_2}, \frac{1}{2c_1}\exp\left(\frac{-B-\mu}{c_1}\right)\right\}} \ , & x < \mu \\ \frac{c_2}{2\min\left\{\frac{1}{2c_2}, \frac{1}{2c_1}\exp\left(\frac{\mu-B}{c_1}\right)\right\}} \ , & x \geq \mu \end{cases} .$$

For any x_1 and x_2, the absolute log-ratio distance for this distribution can be bounded as

$$\begin{aligned} &|\ln p_{\mu,s}(x_1) - \ln p_{\mu,s}(x_2)| \\ =&\tfrac{1}{s}\left|\|x_1 - \mu\| - \|x_2 - \mu\|\right| \leq \tfrac{1}{s}\|x_1 - x_2\| \ , \end{aligned}$$

where the inequality follows from the triangle inequality on $\|\cdot\|$. Thus, if we use $\rho(x, y) = \|x - y\|$, the set of feasible parameters for Assumption 2 is $\mu \in \mathbb{R}$ and $\frac{1}{s} = \theta \leq L$. Again we can use the trimmed exponential prior with rate parameter $\lambda > 0$ for the inverse scale, $\frac{1}{s}$, and similar to the previous example, Assumption 2 is satisfied with $c = \lambda$ and $L_0 = c_1$. ■

Proof of Lemma 21 Here, we consider data drawn from a Binomial distribution with a beta prior on its proportion parameter, θ. Thus, the likelihood and prior functions are

$$\begin{aligned} p_{\theta,n}(X = k) &= \tbinom{n}{k}\theta^k(1-\theta)^{n-k} \\ \xi_0(\theta) &= \tfrac{1}{B(a,b)}\theta^{a-1}(1-\theta)^{b-1} \ , \end{aligned}$$

where $k \in \{0, 1, 2, \ldots, n\}$, $a, b \in \mathbb{R}_+$ and $B(a, b)$ is the beta function. The resulting posterior is a Beta-Binomial distribution. Again we consider the application of Assumption 2 to this Beta-Binomial distribution. For this purpose, we must quantify the parameter sets Θ_L for a given $L > 0$ according to a distance function. The absolute log-ratio distance between the Binomial likelihood function for any pair of arguments, k_1 and k_2, is

$$|\ln p_{\theta,n}(k_1) - \ln p_{\theta,n}(k_2)| = \left|\Delta_n(k_1, k_2) + (k_1 - k_2)\ln\tfrac{\theta}{1-\theta}\right|$$

where $\Delta_n(k_1, k_2) \triangleq \ln\binom{n}{k_1} - \ln\binom{n}{k_2}$. By substituting this distance into the supremum of Eq. (4), we seek feasible values of $L > 0$ for which the supremum is non-negative; here, we explore the case where $\rho((n, k_1), (n, k_2)) \triangleq |k_1 - k_2|$. Without loss of generality, we assume $k_1 > k_2$, and thus require that

$$\sup_{k_1 > k_2} \left|\tfrac{\Delta_n(k_1,k_2)}{k_1-k_2} + \ln\tfrac{\theta}{1-\theta}\right| \leq L \ . \tag{15}$$

However, by the definition of $\Delta_n(k_1, k_2)$, the ratio $\frac{\Delta_n(k_1,k_2)}{k_1-k_2}$ is in fact the slope of the chord from k_2 to k_1 on the function $\ln\binom{n}{k}$. Since the function $\ln\binom{n}{k}$ is concave in k, this slope achieves its maximum and minimum at its boundary values; *i.e.*, it is maximised for $k_1 = 1$ and $k_2 = 0$ and minimised for $k_1 = n$ and $k_2 = n - 1$. Thus, the ratio attains a maximum value of $\ln n$ and a minimum of $-\ln n$ for which the above supremum is simply $\ln n + \left|\ln\frac{\theta}{1-\theta}\right|$. From Eq. (15), we therefore have, for all $L \geq \ln n$:

$$\Theta_L = \left[\left(1 + \tfrac{e^L}{n}\right)^{-1}, \left(1 + \tfrac{n}{e^L}\right)^{-1}\right] \ .$$

We want to bound $\xi(\Theta_L)$. We know that: $\xi(\Theta_L) = 1 - \xi\left(\Theta_L^\complement\right)$ where $\Theta_L^\complement$ is the complement of Θ_L. so $\xi(\Theta_L^\complement)$ is composed of two symmetric intervals: $\left[0, \left(1 + \frac{e^L}{n}\right)^{-1}\right)$ and $\left(\left(1 + \frac{n}{e^L}\right)^{-1}, 1\right]$. We selected $\alpha = \beta$, therefore the mass must concentrate at $\frac{1}{2}$, as we have $\alpha > 1$.

Due to symmetry, the mass outside of Θ_L is two times that is the first interval. This is:

$$\frac{2}{B(\alpha,\alpha)}\int_0^{\frac{p}{1+p}} x^{\alpha-1}(1-x)^{\alpha-1}\,\mathrm{d}x \ .$$

where p denotes $ne^{-L} \in [0, 1]$, Therefore c is upper bounded by

$$\ln\left(\frac{2A(p)}{B(\alpha,\alpha)}\right)/(L_0 - L) = \ln\left(\frac{2A(p)}{B(\alpha,\alpha)}\right)/\ln p,$$

where $A(p)$ denotes the incomplete Beta function $\int_0^{\frac{p}{1+p}} x^{\alpha-1}(1-x)^{\alpha-1}dx$. Note that we have

$$A'(p) = \frac{p^{\alpha-1}}{(1+p)^{2\alpha}} \ ,$$

$$A''(p) = \frac{p^{\alpha-2}[(\alpha-1)(1+p) - 2\alpha p]}{(1+p)^{2\alpha+1}} \ .$$

Claim 2 *$H(p) = \alpha A(p) - \frac{p^\alpha}{(1-p)(1+p)^{2\alpha-1}} \leq 0$ for all $p \in (0, 1)$.*

Proof Calculating derivatives and simplifying

$$\begin{aligned}
& H'(p) \\
= \; & \alpha A'(p) - \frac{\alpha p^{\alpha-1}(1-p)(1+p)^{2\alpha-1} - p^\alpha\left[(2\alpha-1)(1-p)(1+p)^{2\alpha-2} - (1+p)^{2\alpha-1}\right]}{[(1-p)(1+p)^{2\alpha-1}]^2} \\
= \; & \frac{\alpha p^{\alpha-1}}{(1+p)^{2\alpha}} - \frac{\alpha p^{\alpha-1}(1-p)(1+p) - p^\alpha[(2\alpha-1)(1-p) - (1+p)]}{(1-p)^2(1+p)^{2\alpha}} \\
= \; & \frac{p^{\alpha-1}}{(1+p)^{2\alpha}}\left(\alpha - \frac{\alpha(1-p^2) - 2p(\alpha-1-p\alpha)}{(1-p)^2}\right) \\
= \; & \frac{p^{\alpha-1}}{(1+p)^{2\alpha}(1-p)^2}\left(\alpha(1-2p+p^2) - \alpha(1-p^2) + 2p(\alpha-1-\alpha p)\right) \\
= \; & \frac{-2p^\alpha}{(1+p)^{2\alpha}(1-p)^2} < 0 \; .
\end{aligned}$$

Therefore $H(p)$ is strictly decreasing. Then combined with $H(0) = 0$, we claim follows. ■

Claim 3 *$G(p) = p\frac{A'(p)}{A(p)}\ln p - \ln\frac{2A(p)}{B(\alpha,\alpha)} < 0$ for all $p \in (0,1)$.*

Proof Again taking derivatives

$$\begin{aligned}
G'(p) &= \frac{A'(p)}{A(p)}(1+\ln p) + p\ln p\frac{A''(p)A(p) - A'(p)^2}{A(p)^2} - \frac{A'(p)}{A(p)} \\
&= \frac{\ln p}{A(p)^2}(A(p)A'(p) + pA''(p)A(p) - pA'(p)^2) \\
&= \frac{\ln p}{A(p)^2}\left[\frac{p^{\alpha-1}}{(1+p)^{2\alpha}}A(p)\left(1 + \frac{(\alpha-1)(1+p) - 2\alpha p}{1+p}\right) - \frac{p^{2\alpha-1}}{(1+p)^{4\alpha}}\right] \\
&= \frac{\ln p}{A(p)^2}\frac{p^{\alpha-1}}{(1+p)^{2\alpha+1}}\left[\alpha(1-p)A(p) - \frac{p^\alpha}{(1+p)^{2\alpha-1}}\right] \\
&= \frac{p^{\alpha-1}}{(p+1)^{2\alpha+1}A(p)^2}H(p)\ln p(1-p) > 0 \; .
\end{aligned}$$

So $G(p)$ is strictly increasing. Combined with $\lim_{p\to 1} G(p) = 0$, the claim follows. ■

Claim 4 *$F(p) = \ln\left(2I_{\frac{p}{1+p}}(\alpha)\right)/\ln p$ is decreasing in $p \in (0,1)$, where the incomplete Beta function $I_{\frac{p}{1+p}}(\alpha) = A(p)/B(\alpha,\alpha)$.*

Proof Taking derivatives

$$\begin{aligned}
F'(p) &= \frac{1}{\ln^2 p}\left(\frac{A'(p)}{A(p)}\ln p - \frac{1}{p}\ln\frac{2A(p)}{B(\alpha,\alpha)}\right) \\
&= \frac{1}{p\ln^2 p}\left(\frac{A'(p)}{A(p)}p\ln p - \ln\frac{2A(p)}{B(\alpha,\alpha)}\right) \\
&= \frac{1}{p\ln^2 p}G(p) < 0 \; .
\end{aligned}$$

■

Therefore $\ln\left(2I_{\frac{p}{1+p}}(\alpha)\right)/\ln p$ is monotonic decreasing in p. Thus the minimum value of $F(p)$ is $\frac{1}{B(\alpha)2^{2\alpha-1}}$ as $p \to 1$, which we can take as our c in this example.

Let us consider $C_\xi^{\mathcal{F}_\Theta}$ for this example. We have

$$\frac{p_\theta(x)}{\phi(x)} = \frac{B(\alpha,\beta)\theta^x(1-\theta)^{n-x}}{B(\alpha+x, n+\beta-x)} \; ,$$

where $\theta \in [0,1]$ and $x \in [0,1,\ldots,n]$. Note that

$$\frac{B(\alpha+x+1, n+\beta-x-1)}{B(\alpha+x, n+\beta-x)} = \frac{\Gamma(\alpha+x+1)\Gamma(n+\beta-x-1)}{\Gamma(\alpha+x)\Gamma(n+\beta+1)} = \frac{\alpha+x}{n+\beta-x-1} \; .$$

So $B(\alpha+x+1, n+\beta-x-1) \le B(\alpha+x, n+\beta-x)$ if $x \le \frac{n+\beta-\alpha-1}{2}$; $B(\alpha+x+1, n+\beta-x-1) > B(\alpha+x, n+\beta-x)$ otherwise. Thus

$$B(\alpha+x, n+\beta-x) \ge B\left(\frac{n+\alpha+\beta-1}{2}, \frac{n+\alpha+\beta+1}{2}\right) \; .$$

Hence we can take $C_\xi^{\mathcal{F}_\Theta} = B(\alpha,\beta)/B\left(\frac{n+\alpha+\beta-1}{2}, \frac{n+\alpha+\beta+1}{2}\right)$. ■

Proof of Lemma 22 Since $N(x;\mu,\theta)$ is decreasing in x^2 and concave as a function of θ. We have $\phi(t) \ge \inf_{\{x : \|x\| \le B\}, \theta \in [c_1,c_2]} N(x;\mu,\theta) = \min\left\{\sqrt{\frac{c_1}{2\pi}}e^{\frac{-c_1c_2^2}{2}}, \sqrt{\frac{c_2}{2\pi}}e^{\frac{-c_2^3}{2}}\right\}$. Then we can take

$$C_\xi^{\mathcal{F}_\Theta} = \min\left\{\sqrt{c_2/c_1}e^{\frac{c_1c_2^2}{2}}, e^{\frac{c_2^3}{2}}\right\}$$

For the normal distribution, (4) requires: $2L\rho(x,y)\sigma^2 \ge |2\mu - x - y|\,|x-y|$. Taking the absolute log ratio of the Gaussian densities we have

$$\begin{aligned}
& \frac{1}{2\sigma^2}\left|\left((x-\mu)^2 - (y-\mu)^2\right)\right| \\
\le & \frac{\max\{|\mu|, 1\}}{2\sigma^2}\left(\left|x^2 - y^2\right| + 2|x-y|\right) .
\end{aligned}$$

Consequently, we can set $\rho(x,y) = \left|x^2 - y^2\right| + 2|x-y|$ and $L(\mu,\sigma) = \frac{\max\{|\mu|,1\}}{2\sigma^2}$. Again, the trimmed exponential prior is given by $K\lambda e^{-\lambda\theta}$, where $K = (e^{-\lambda c_1} - e^{-\lambda c_2})^{-1}$. Thus the CDF at L of this density is $K\left(e^{-\lambda c_1} - e^{-\lambda L}\right)$ for $L \in [\frac{c_1\max\{|\mu|,1\}}{2}, \frac{c_2\max\{|\mu|,1\}}{2}]$ and 1 for $L \ge \frac{c_2\max\{|\mu|,1\}}{2}$. Thus the CDF at L of this density is $K\left(e^{-\lambda c_1} - e^{\frac{-2\lambda L}{\max\{|\mu|,1\}}}\right)$. We choose L_0 to be $\frac{c_1\max\{|\mu|,1\}}{2}$. Then we need to find c such that

$$\xi(\Theta_L) = \int_{c_1}^{L} K\lambda e^{-\lambda\theta}d\theta = K(e^{-\lambda c_1} - e^{-\lambda L}) \ge 1 - e^{-c\left(L - \frac{c_1\max\{|\mu|,1\}}{2}\right)}.$$

By plugging K into the inequality, we have

$$e^{-c\left(L-\frac{c_1\max\{|\mu|,1\}}{2}\right)} \geq \frac{e^{\frac{-2\lambda L}{\max\{|\mu|,1\}}+\lambda c_2}-1}{e^{-\lambda(c_1-c_2)}-1}.$$

Since $e^{-\lambda\left(\frac{2\lambda L}{\max\{|\mu|,1\}}-c_2\right)} \leq e^{-\lambda(c_1-c_2)}$, it is sufficiency to find c such that

$$e^{-c\left(L-\frac{c_1\max\{|\mu|,1\}}{2}\right)} \geq e^{-\lambda\left(\frac{2L}{\max\{|\mu|,1\}}-c_1\right)}.$$

This is equivalent to have c satisfying

$$c\left(L-\frac{c_1\max\{|\mu|,1\}}{2}\right) \leq \lambda\left(\frac{2L}{\max\{|\mu|,1\}}-c_1\right).$$

Then we can take $c=\frac{2\lambda}{\max\{|\mu|,1\}}$ to satisfy the above inequality. ■

Proof of Lemma 23 Consider the likelihood log-ratio distance of multivariate normal distributions with precision matrix A:

$$\frac{1}{2}|x^\top Ax - y^\top Ay| \ ,$$

where A is positive definite with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_n > 0$). For simplicity, assume the mean to be a zero vector then

$$\begin{aligned}
|x^\top Ax - y^\top Ay| &= \left|\sum_{i,j} x_i x_j A_{i,j} - \sum_{i,j} y_i y_j A_{i,j}\right| \\
&= \left|\sum_{i,j} A_{i,j}(x_i x_j - y_i y_j)\right| \\
&= |Tr(A(xx^\top - yy^\top)')| \\
&\leq [Tr(A^2)Tr((xx^\top - yy^\top)(xx^\top - yy^\top)')]^{\frac{1}{2}} \\
&= \left(\sum_{i=1}^n \lambda_i^2\right)^{\frac{1}{2}} ||(xx^\top - yy^\top)||_F \ .
\end{aligned}$$

For mean equal to μ, we have

$$\frac{1}{2}|(x^\top-\mu)A(x-\mu) - (y^\top-\mu)A(y-\mu)| \ .$$

By the above analysis we have the difference being bounded by

$$\frac{1}{2}\left(\sum_{i=1}^n \lambda_i^2\right)^{\frac{1}{2}} ||(x-\mu)(x-\mu)' - (y-\mu)(y-\mu)')||_F \ .$$

Note that

$$\begin{aligned}
||(x-\mu)(x-\mu)' - (y-\mu)(y-\mu)')||_F =& ||xx^\top - \mu(x^\top - y^\top) - (x-y)\mu' - yy^\top||_F \\
\leq& ||xx^\top - yy^\top||_F + 2||\mu(x-y)'||_F \\
=& ||xx^\top - yy^\top||_F + 2||\mu||_2||(x-y)'||_2 \\
\leq& \max\{1, ||\mu||_2\}(||xx^\top - yy^\top||_F + 2||x-y||_2) \ .
\end{aligned}$$

■

Proof of Lemma 24 It is instructive to first examine the case where all variables are independent and we have a single draw from P_θ. Then $P_\theta(x) = \prod_{k=1}^K \theta_{k,x_k}$ and

$$\left|\ln\frac{P_\theta(x)}{P_\theta(y)}\right| = \left|\ln\prod_{k=1}^K \frac{\theta_{k,x_k}}{\theta_{k,y_k}}\right| \leq \sum_{k=1}^K \left|\ln\frac{\theta_{k,x_k}}{\theta_{k,y_k}}\right| \mathbb{I}\{x_k \neq y_k\} \leq \max_{i,j,k}\left|\ln\frac{\theta_{k,i}}{\theta_{k,j}}\right| \rho(x,y) \ . \tag{16}$$

Consequently, if $\varepsilon \triangleq \min_{k,j}\theta_{k,j}$ is the smallest probability assigned to any one sub-event, then $L > \ln 1/\varepsilon$, since $\theta_{k,j} \leq 1$.

In the general case, we have independent draws x^t, y^t, where $x^t \sim P_\theta(x)$ and the variables x_k^t have dependences defined through a graphical model, such that $P_\theta(x) = \prod_k P_\theta(x_k \mid x_{\mathcal{P}(k)})$, where $\mathcal{P}(k)$ are the parents of node k. Similarly to (16), we write

$$\begin{aligned}
\left|\ln\frac{P_\theta(x)}{P_\theta(y)}\right| &= \left|\ln\prod_t \frac{P_\theta(x^t)}{P_\theta(y^t)}\right| = \left|\ln\prod_t\prod_k \frac{P_\theta(x_k^t \mid x_{\mathcal{P}(k)}^t)}{P_\theta(y_k^t \mid y_{\mathcal{P}(k)}^t)}\right| \\
&\leq \sum_{t.k}\left|\ln\frac{P_\theta(x_k^t \mid x_{\mathcal{P}(k)}^t)}{P_\theta(y_k^t \mid y_{\mathcal{P}(k)}^t)}\right| \leq \ln\frac{1}{\epsilon}\sum_{t.k}\mathbb{I}\left\{x_k^t \neq y_k^t \vee x_{\mathcal{P}(k)}^t \neq y_{\mathcal{P}(k)}^t\right\}.
\end{aligned} \tag{17}$$

The last term is the number of times a value is different in x and y times one plus the number of variables it affects. To model this, let $v \in \mathbb{N}^K$ be such that $v_k = 1 + \deg(k)$ and define: $\rho(x,y) \triangleq v^\top \delta(x,y)$ and $\delta_k(x,y) \triangleq \sum_t \mathbb{I}\{x_{k,t} \neq y_{k,t}\}$. Rewriting (17) in terms of ρ, we obtain $\left|\ln\frac{P_\theta(x)}{P_\theta(y)}\right| \leq \ln\frac{1}{\varepsilon}\cdot\rho(x,y)$ as desired. ■

References

James O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, 1985.

Peter J. Bickel and Kjell A. Doksum. *Mathematical Statistics: Basic Ideas and Selected Topics*, volume 1. Holden-Day Company, 2001.

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2(Mar):499–526, 2002.

Konstantinos Chatzikokolakis, Miguel E. Andres, Nicolas Emilio Bordenabe, and Catuscia Palamidessi. Broadening the scope of differential privacy using metrics. In *Privacy Enhancing Technologies*, pages 82–102, 2013.

Kamalika Chaudhuri and Daniel Hsu. Convergence rates for differentially private statistical estimation. In *Proceedings of the 29th International Conference on Machine Learning*, ICML, pages 1327–1334, 2012.

Kamalika Chaudhuri, Claire Monteleoni, and Anand D. Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.

Morris H. DeGroot. *Optimal Statistical Decisions*. John Wiley & Sons, 1970.

Christos Dimitrakakis, Blaine Nelson, Aikaterini Mitrokotsa, and Benjamin Rubinstein. Robust and private Bayesian inference. Technical Report 1306.1066, arXiv, 2013. Latest version 2015.

Christos Dimitrakakis, Blaine Nelson, Aikaterini Mitrokotsa, and Benjamin Rubinstein. Robust and private Bayesian inference. In *25th Conference on Algorithmic Learning Theory (ALT)*, volume 8776 of *Lecture Notes in Computer Science*, pages 291–305. Springer, 2014.

John C Duchi, Michael Jordan, and Martin J Wainwright. Local privacy and statistical minimax rates. In *54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 429–438. IEEE, 2013.

Cynthia Dwork. Differential privacy. In *Automata, Languages and Programming (ICALP)*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2006.

Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC, pages 371–380, 2009.

Cynthia Dwork and Adam Smith. Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2):135–154, 2009.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Theory of Cryptography Conference*, TCC, pages 265–284, 2006.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. The reusable holdout: Preserving validity in adaptive data analysis. *Science*, 349 (6248):636–638, 2015.

Alexei A. Fedotov, Peter Harremoës, and Flemming Topsoe. Refinements of Pinsker's inequality. *IEEE Transactions on Information Theory*, 49(6):1491–1498, 2003.

Peter Grünwald. The safe Bayesian: Learning the learning rate via the mixability gap. In *Proceedings of the 23rd International Conference on Algorithmic Learning Theory*, (ALT), pages 169–183, 2012.

Peter D. Grünwald and A. Philip Dawid. Game theory, maximum entropy, minimum discrepancy, and robust Bayesian decision theory. *The Annals of Statistics*, 32(4):1367–1433, 2004.

Rob Hall, Alessandro Rinaldo, and Larry Wasserman. Random differential privacy. *Journal of Privacy and Confidentiality*, 4(2), 2011.

Rob Hall, Alessandro Rinaldo, and Larry Wasserman. Differential privacy for functions and functional data. *Journal of Machine Learning Research*, 14(Feb):703–727, 2013.

Frank R. Hampel, Elvezio M. Ronchetti, Peter J. Rousseeuw, and Werner A. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. John Wiley and Sons, 1986.

Peter J. Huber. *Robust Statistics*. John Wiley and Sons, 1981.

Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. In *Proceedings of The 32nd International Conference on Machine Learning*, ICML, pages 1376–1385, 2015.

Daniel Kifer and Ashwin Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Transactions on Database Systems (TODS)*, 39(1):3, 2014.

Lucien LeCam. Convergence of estimates under dimensionality restrictions. *The Annals of Statistics*, pages 38–53, 1973.

Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS, pages 94–103, 2007.

Darakhshan Mir. Differentially-private learning and information theory. In *Proceedings of the 2012 Joint EDBT/ICDT Workshops*, pages 206–210. ACM, 2012.

VI Norkin. Stochastic Lipschitz functions. *Cybernetics and Systems Analysis*, 22(2):226–233, 1986.

Ian Osband, Daniel Russo, and Benjamin Van Roy. (More) efficient reinforcement learning via posterior sampling. In *Advances in Neural Information Processing Systems*, NIPS, pages 3003–3011, 2013.

Benjamin I. P. Rubinstein, Peter L. Bartlett, Ling Huang, and Nina Taft. Learning in a large function space: Privacy-preserving mechanisms for SVM learning. *Journal of Privacy and Confidentiality*, 4(1), 2012.

W.R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.

Yu-Xiang Wang, Stephen E. Fienberg, and Alex Smola. Privacy for free: Posterior sampling and stochastic gradient monte carlo. In *Proceedings of The 32nd International Conference on Machine Learning*, ICML, pages 2493–2502, 2015.

Larry Wasserman and Shuheng Zhou. A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105(489):375–389, 2010.

Tsachy Weissman, Erik Ordentlich, Gadiel Seroussi, Sergio Verdu, and Marcelo J Weinberger. Inequalities for the L_1 deviation of the empirical distribution. Technical report, Hewlett-Packard Labs, 2003.

Oliver Williams and Frank McSherry. Probabilistic inference and differential privacy. In *Advances in Neural Information Processing Systems*, NIPS, pages 2451–2459, 2010.

Yonghui Xiao and Li Xiong. Bayesian inference under differential privacy. arXiv preprint arXiv:1203.0617, 2012.

Bin Yu. Assouad, Fano, and Le Cam. In *Festschrift for Lucien Le Cam*, pages 423–435. Springer, 1997.

Zuhe Zhang, Benjamin I. P. Rubinstein, and Christos Dimitrakakis. On the differential privacy of Bayesian inference. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, AAAI, pages 2365–2371, 2016.

Shijie Zheng. The differential privacy of Bayesian inference, 2015. Bachelor's thesis, Harvard College `http://nrs.harvard.edu/urn-3:HUL.InstRepos:14398533`.

Journal of Machine Learning Research 18 (2017) 1-5 Submitted 8/15; Revised 9/16; Published 3/17

Refinery: An Open Source Topic Modeling Web Platform

Daeil Kim DAEIL@CS.BROWN.EDU
Benjamin F. Swanson DUJIAOZHU@GMAIL.COM
Michael C. Hughes MHUGHES@CS.BROWN.EDU
Erik B. Sudderth SUDDERTH@CS.BROWN.EDU
Department of Computer Science, Brown University, Providence, RI 02192, USA

Editor: Geoff Holmes

Abstract

We introduce Refinery, an open source platform for exploring large text document collections with topic models. Refinery is a standalone web application driven by a graphical interface, so it is usable by those without machine learning or programming expertise. Users can interactively organize articles by topic and also refine this organization with phrase-level analysis. Under the hood, we train Bayesian nonparametric topic models that can adapt model complexity to the provided data with scalable learning algorithms. The project website `http://daeilkim.github.io/refinery/` contains Python code and further documentation.

Keywords: topic models, visualization, software

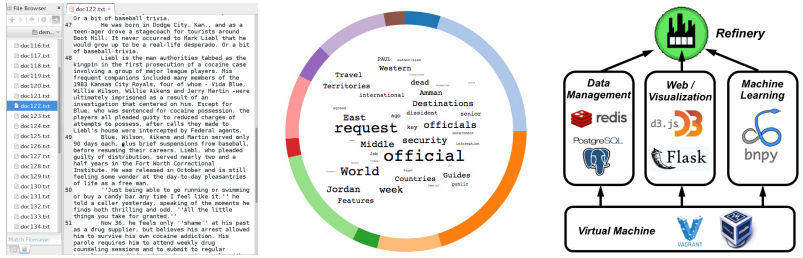
1. Introduction

Topic models have become a ubiquitous class of tools for the analysis of large document collections (Blei, 2012). However, there is still a significant adoption barrier for individuals who have little to no background in coding or computer science. For example, librarians might be interested in understanding ways of organizing their vast archives of information while journalists could potentially use learned topics to extract insights from Freedom of Information Act (FOIA) requests. However, few have the expertise to implement necessary code nor an intuition for how to fine tune parameters associated with these models. Beyond the challenge of running statistical inference algorithms, the learned topics and word associations must be visualized in a way that is accurate but intuitive.

To make topic modeling algorithms and results more accessible, we built an open source web application called Refinery. Refinery allows users to explore a large corpus of documents and further identify a small set of relevant documents for careful study. The simple drag and drop operation of this toolbox allows researchers to focus on the process of quickly understanding their data without being encumbered by the complexities of inference.

2. Running Refinery

Refinery is an in-browser web application that builds on many existing open-source projects related to visualization, data management, and machine learning, as shown in Fig. 1. To make installation simple, it has only three dependencies: the Git version-control system,



(a) *Input:* Plain-text files, one per document (b) *Output:* Interactive visualization of learned topics (c) Technology stack

Figure 1: Refinery is a web-application that allows users to upload plain-text documents (a) and visualize the semantic themes learned by a topic-model (b). Under the hood, Refinery uses many existing technologies (c) for browser interaction, data management, machine learning, and visualization. A video demo is available online: `http://youtu.be/7yRQ1J9Z_LI`.

Virtualbox (Oracle, 2013), and Vagrant (Hashimoto, 2013). At a Unix-like command line with these dependencies installed, running Refinery requires just a few lines of code:

```
> git clone https://github.com/daeilkim/refinery.git
> vagrant up # launch machine and install required Python packages
```

After executing these lines, Refinery can be accessed from a web browser via the local address `http://11.11.11.11:8080`.

3. Topic Modeling

Uploading documents. Users submit documents for analysis in the form of a `.zip` or `.tar.gz` file, which uncompressed contains a folder of plain-text files, one for each individual document. Users do *not* need to preprocess documents or identify a vocabulary in advance. This is done automatically by Refinery, which tokenizes every unigram from the input documents. The final vocabulary excludes very rare and very common words; we discard terms appearing in more than 80% of all documents, or in less than 2 documents.

Training a topic model. The topic modeling algorithm used for Refinery comes from the BNPy toolbox (Hughes et al., 2015b). Specifically, we train a *hierarchical Dirichlet process* (HDP) topic model (Teh et al., 2006) using a memoized variational inference algorithm (Hughes et al., 2015a). While the underlying algorithm has many free parameters, we set most of these to smart defaults and only ask the user to specify an upper bound on the number of inferred topics. This bound provides a truncation for infinite-dimensional HDP model.

Browseable visualization. After training a model, users can explore the resulting topics interactively via a multi-colored ring, as shown in Fig. 2. Each segment of the ring represents one topic, with its size indicative of the topic's frequency in the overall corpus. Hovering

©2017 Kim, Swanson, Hughes and Sudderth.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v18/15-441.html`.

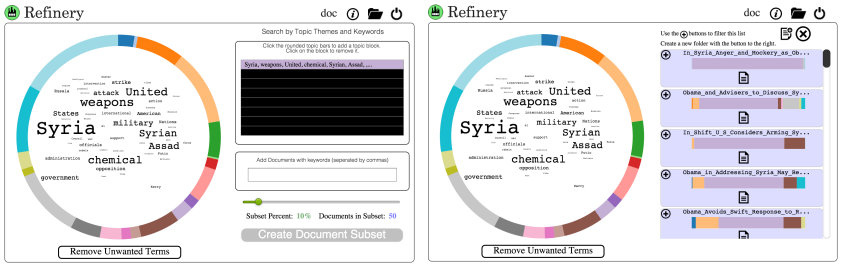


Figure 2: The results of an analysis on 500 New York Times articles that contained the keyword "obama" during the year 2013. In the figure above, the "Syria" topic has been selected along with a subset of 50 documents related to this topic.

the cursor over a topic's segment shows the top 50 words associated with that topic. In this word cloud visualization, text size is scaled according to the topic-specific word frequency.

Focused exploration of document subsets. Often, analysis of large collections requires identifying subsets of relevant documents and performing more detailed clustering of that subset. We support this by allowing the main corpus of documents to be filtered by keyword and topic presence. Refinery provides two methods for selecting relevant documents given a trained topic model with K topics. The first allows the user to directly specify a query distribution over topics which recovered documents should emulate. The second takes in a list of search terms, and averages the probability of topic given word for each term to obtain a (normalized) distribution over K topics. Given this K-dimensional query vector, documents are ranked based on the KL divergence between their MAP topic distribution and the query.

4. Phrase Extraction and Refinement

While topic modeling enables scalable first-pass analysis, often the underlying bag-of-words assumptions are too limiting. After filtering a large corpus to find only documents relevant to a small set of topics, Refinery further allows users to explore documents by phrase similarity and create a summary of the corpus composed of sentences extracted from the input documents.

We use the open-source implementation of the Splitta algorithm (Gillick, 2010), which Gillick (2009) introduced for sentence boundary detection. First, the Splitta algorithm is applied for sentence boundary detection. This algorithm processes raw text and does not make use of the topic model output. Next, to help the user efficiently select a subset of relevant sentences from their corpus, Refinery offers two exploration modes. The first, triggered by the button marked "Variety", implements the KLSum algorithm for multidocument summarization (Haghighi and Vanderwende, 2009). This algorithm ranks candidate sentences higher if their addition to the current set would bring the unigram distribution closer to that of the corpus as a whole, intuitively preferring sentences that contain globally

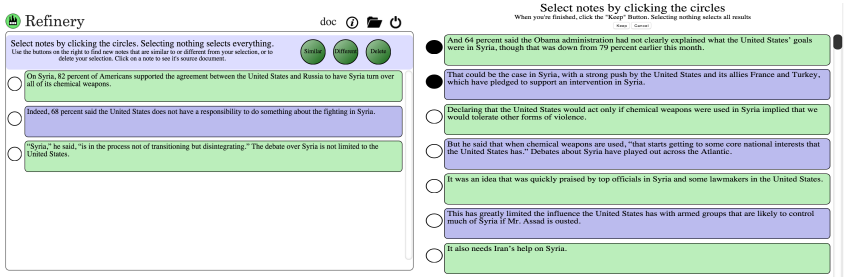


Figure 3: Refinery's phrase extraction feature. *Left:* User has selected some initial phrases related to the war in Syria from news articles. *Right:* Hitting the "similar" button brings up another set of phrases similar to the initial set.

relevant information that does not yet appear in the set of selected sentences. The second exploration method, labeled "Similar", simply ranks candidates by cosine similarity to the current summary's unigram distribution. Each sentence in the summary is linked to and highlighted in its original source document, facilitating the creation of a comprehensive set of notes with fast access to their provenance.

5. Discussion and Related Work

Refinery provides a first step towards allowing non-technical professionals to explore large document collections with modern topic models. Several others groups have strived to simplify and democratize the use of topic models. Notably, Chaney and Blei (2012) created a web-based navigator to help users understand the relationship between topics and documents, while Chuang et al. (2012) developed a visualization package for assessing the goodness of topics. Topicnets (Gretarsson et al., 2012) focuses on a graph-based exploration and visualization of topics, while other packages such as Gephi (Bastian et al., 2009) and Tethne (Peirson et al., 2015) are often used to create similar visualizations of an already-learned set of topics. Most prior work has focused on the parametric *latent Dirichlet allocation* (LDA) topic model (Blei et al., 2003).

Refinery differs from these packages in the way it simplifies the entire process of analyzing text with topic models for non-expert users. Furthermore, Refinery's use of scalable Bayesian nonparametric topic models offers the benefit of automatic model selection while parametric models like LDA require a prior knowledge of the number of topics. Refinery also supports phrase extraction, which allows for a more refined search across documents. Building on the BNPy toolbox allows potential future extensions to more advanced models that cluster documents organized as a time-series or network. Ultimately, we plan to support a larger variety of potential document file types including structured word processor files, spreadsheets, and presentations.

Acknowledgments

Refinery was a recipient of the Knight Prototype Fund in 2014.

References

Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi: An open source software for exploring and manipulating networks. In *International AAAI Conference on Weblogs and Social Media*, 2009. URL `http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154`.

David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

Allison June-Barlow Chaney and David M Blei. Visualizing topic models. In *ICWSM*, 2012.

Jason Chuang, Christopher D. Manning, and Jeffrey Heer. Termite: Visualization techniques for assessing textual topic models. In *Advanced Visual Interfaces*, 2012. URL `http://vis.stanford.edu/papers/termite`.

Dan Gillick. Sentence boundary detection and the problem with the U.S. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 241–244. Association for Computational Linguistics, 2009.

Daniel Gillick. splitta: statistical sentence boundary detection, 2010. URL `https://code.google.com/p/splitta/`.

Brynjar Gretarsson, John Odonovan, Svetlin Bostandjiev, Tobias Höllerer, Arthur Asuncion, David Newman, and Padhraic Smyth. Topicnets: Visual analysis of large text corpora with topic modeling. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(2):23, 2012.

Aria Haghighi and Lucy Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370. Association for Computational Linguistics, 2009.

Mitchell Hashimoto. *Vagrant: Up and Running*. O'Reilly Media, Inc., 2013. URL `http://www.vagrantup.com`.

Michael C. Hughes, Daeil Kim, and Erik B. Sudderth. Reliable and scalable variational inference for the hierarchical dirichlet process. In *AISTATS*, 2015a.

Michael C. Hughes et al. BNPy: Bayesian nonparametric machine learning for Python, 2015b. URL `https://github.com/bnpy/bnpy/`.

Oracle. Virtualbox, 2013. URL `http://www.virtualbox.org`. Version 4.2.18.

Erick Peirson, Aaron Baker, and Ramki Subramanian. Tethne: Bibliographic network analysis in Python, 2015. URL `https://github.com/diging/tethne`.

Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *JASA*, 101 (476):1566–1581, 2006.

Journal of Machine Learning Research 18 (2017) 1-43 Submitted 8/15; Revised 9/16; Published 3/17

Using Conceptors to Manage Neural Long-Term Memories for Temporal Patterns

Herbert Jaeger H.JAEGER@JACOBS-UNIVERSITY.DE
Dpt. of Computer Science and Electrical Engineering
Jacobs University Bremen
28759 Bremen, Germany

Editor: Nando de Freitas

Abstract

Biological brains can learn, recognize, organize, and re-generate large repertoires of temporal patterns. Here I propose a mechanism of neurodynamical pattern learning and representation, called *conceptors*, which offers an integrated account of a number of such phenomena and functionalities. It becomes possible to store a large number of temporal patterns in a single recurrent neural network. In the recall process, stored patterns can be morphed and "focussed". Parametric families of patterns can be learnt from a very small number of examples. Stored temporal patterns can be content-addressed in ways that are analog to recalling static patterns in Hopfield networks.

Keywords: Recurrent neural network, temporal pattern learning, neural long-term memory, neural dynamics

1. Introduction

In the cognitive and neurosciences as well as in neural computation it is customary to distinguish between various sorts of short-term memory and long-term memory (Fusi and Wang, 2016). This article is concerned with neural long-term memory (LTM) in the sense that some information becomes permanently coded in synaptic weights at learning time, to be somehow recalled at exploitation time without changing those weights.

The paradigmatic model of a neural LTM is the *associative memory* pioneered, among others, by Willshaw et al. (1969), Cooper (1973), Kohonen (1974), Palm (1980) and Hopfield (1982). This classical family of models explains how a number of different patterns can be stored in a neural network by an explicit calculation of the network weight matrix or by iterative learning processes. A rich mathematical theory affords insights into capacity bounds and noise robustness. But, these classical associative memories are mainly devised for storing *static* patterns, for instance bit vectors or images. Biological neural systems obviously can learn and recall rich repertoires of *temporal* patterns, for instance gestures, melodies, words. It would be desirable to have a neural network model which extends the capabilities of associative networks to the domain of temporal patterns. This challenge is still largely open. I am aware of only a few proposals for neural memories of temporal patterns:

1. The classical associative memory model for static patterns has been extended on several occasions to learn sequences of patterns. The key mechanism is to replace the learning of auto-associations of stored patterns (which results in fixed-point attractors representing these) by a learning of hetero-associations of a temporal pattern frame with its successor frame. This leads to recall episodes which step through a discrete sequence of patterns (Amari, 1972). The basic discrete hetero-associative memory mechanism has been elaborated in various ways to accomodate analog values, smoother temporal development, or identical patterns at different time slices (examples: Sompolinsky and Kanter (1986); Rinkus (1996); Billard and Hayes (1999); Huang and Hagiwara (2002)). A recent extension of hetero-associative symbol sequence coding networks (Jiang et al., 2016) integrates methods from graph-based coding theory, which leads to improved memory capacities and temporal pattern completion robustness, and enables an incremental storing of patterns.

2. In the field of reservoir computing, a temporal pattern is trained into a recurrent neural network (RNN) by adapting only the weights from the ("reservoir") network to the output neuron. Thus an arbitrary number of temporal patterns can be learnt on the basis of a single reservoir network if individual patterns are represented by separate output neurons, as for example in Hinaut and Dominey (2011).

3. Since the advent of RNN models in cognitive modeling it has been recognized that RNNs can be trained to generate several temporal patterns when they are paired in training with individual pattern-addressing input signals (Kolen and Pollack, 1991; Jordan, 1997; Krause et al., 2010).

4. Another possibility to train and select several temporal patterns in a single RNN is to associate different patterns with different initial network states. An example is Paine and Tani (2005) where an RNN-based mobile robot controller was trained to steer the robot into different action sequences depending on the state initialization.

5. In machine learning, RNNs trained on sequence prediction/generation tasks are typically considered as models of a stochastic process, for instance the process whose realizations are Wikepedia texts, as in Sutskever et al. (2011), rather than as cases of neural memory. But it also makes good sense to view them as neural memories. One may say that such a network after training has memorized typical continuations of initial cue sequences. Even more generally, any neural system trained on some training data set D for whatever purpose (classification, control, ...) could be regarded as having "stored" the probabilistic structure of D. In this general sense, any network-internal, trained representation can be interpreted as a LTM.

The diversity of these examples shows that there is no unique definition of what makes a neural LTM. Here is a list of system properties that are variously attributed (or not) to a neural memory system:

Multiplicity. An neural memory system typically can store more than only one memory pattern. The number of storable pattern is a standard performance criterion for memory systems.

Addressability. When several patterns have been stored, they should be efficiently addressable at use-time, whether by content-addressing or pointer-addressing.

©2017 Herbert Jaeger.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-449.html.

Generativity. Stored patterns can be re-generated at retrieval time with some degree of accuracy.

Organization and searchability. Stored items are organized in a way that brings relationships between them to the fore, for instance, abstraction or part-of relationships (as in semantic network models in AI). This organization enables a systematic navigation and search in memory space. None of the above examples of dynamical pattern memories has this characteristic, but a diversity of neural learning systems for static patterns have been proposed that have an exploitable internal organization of some sort, for instance self-organizing semantic maps (Ritter and Kohonen, 1989), connectionist-localist networks for semantic concept modeling (Shastri, 1999), or recursive auto-associative memories (Pollack, 1990).

Incremental extensibility. New memory items can be stored incrementally on top of already stored ones. Such an incremental extensibility is difficult or impossible for networks trained with backpropagation due to the "catastrophic forgetting" problem (appreciation in Douglas and Sejnowski (2008), partial solutions and further references in French (2003); Grossberg (2005); Goodrich and Arel (2014)) and for most associative memories (exception: Jiang et al. (2016)).

Differential extension. When adding a new item, store only that information about the new item that is not already present in the memory system. None of the examples has this property but arguably human LTM has it.

Forgetting. Allow the memory system to fade out memory items that are not retrieved for a long time. This characteristic is typical for biological neural networks but is absent in all examples above.

Erasibility. Selectively delete arbitrary items from the memory, freeing storage space. This is typical for digital computer memories but is not usually considered in neural network models.

In this article I describe a neuro-computational mechanism, *conceptors*, by which the dynamics of an RNN can be governed in a variety of ways. Conceptors are a general-purpose mechanism that can be used in a diversity of neural information processing tasks. In a very long techreport (Jaeger, 2014) I present elementary demonstrations of conceptors used for temporal pattern classification, one-shot learning, human motion pattern generation, de-noising and signal separation. The present article focusses on exploits of conceptors for neural LTM management. From the above list of typical LTM properties, conceptors enable or facilitate all except the last two, forgetting and erasability. To keep the length of this article within reasonable bounds, I have omitted a sizable portion of LTM-relevant material from Jaeger (2014), namely the topic of Boolean operations on conceptors and the options that such Boolean operations grant for incremental and differential memory extensibility without catastrophic forgetting.

Most of the technical content is adopted from Jaeger (2014), but assembled in a more compact and intuitive exposition, and backed up by improved simulations. For space economy I omit proofs and some mathematical detail and refer the reader to the full treatment

given in Jaeger (2014). The Matlab code for all simulations is available at the JMLR online paper repository.

The article is organized in three main sections. Section 2 introduces the basic model in a detailed step-by-step fashion. Basic pattern morphing and pattern generalization demos, a "focussing" mechanism and a real-world data demo (learn from human motion capture data to animate a body model with 61 degrees of freedom) illustrate the basic functionalities of conceptors for storing, re-generating and modulating temporal patterns in an RNN. Section 3 introduces a conceptor-based model of content-addressable RNN memories for dynamical patterns. The stored patterns can be recovered from short and possibly distorted cues by a process of conceptor auto-adaptation. In several respects this provides a temporal analog of Hopfield networks.

2. Storing a Multitude of Temporal Patterns in an RNN

This section gives a concise introduction to the basic ideas and formalism of conceptors, starting from a toy example (Subsection 2.1) and then moving on to phenomena of pattern morphing, generalization, and "focussing" in the remaining subsections.

2.1 Basic Idea and Formalism

In this subsection I explain the basic ideas of conceptors by stepping through an elementary demonstration. The task is to store two dissimilar periodic temporal patterns in a tiny RNN and subsequently retrieve them.

2.1.1 Preparations

Before patterns can be stored and recalled, an experimental set-up is installed as follows.

Procuring training patterns. In this article a "pattern" means a discrete-time signal $p(n)$, where $p(n) \in \mathbb{R}^M$ and $n \geq 0$. For this elementary demo I use two discrete-time scalar patterns $p^1(n)$ and $p^2(n)$. The first is a sinewave signal with an irrational period length sampled at integer time points. The irrational period length makes the sampled version quasi-periodic. The second is a periodic signal with integer period 2. Figure 1 (left) depicts the two patterns. I remark that later I will also be considering non-periodic and non-stationary patterns.

Network creation. An RNN with N neurons is installed by randomly creating an $N \times N$ matrix W^* of internal connection weights, an $N \times M$ input weight vector W^{in}, and a random $N \times 1$ bias vector b. Here I use an unrealistically small network with only $N = 3$ neurons, which allows me to illustrate relevant effects in 3-D graphics.

This network can be driven by a scalar pattern $p(n)$ by the following state update equation:

$$\mathbf{x}(n+1) = \tanh(W^* \mathbf{x}(n) + W^{\text{in}} p(n) + \mathbf{b}). \quad (1)$$

All parameters of W^*, W and $\mathbf{b}$ are sampled from a normal distribution and scaled such that an overall system dynamics is obtained that works well for this didactic demonstration. Details of the scaling procedure are not important here, but I remark that the internal weights W^* must be scaled small enough to ensure that the RNN has the *echo state property* with respect to the input signals that are fed to it. In intuitive terms, an RNN has the echo

state property with respect to an input signal $p(n)$ if any initial network state is "forgotten" (washed out) when the network is driven by $p(n)$ for a long enough time. The echo state property is a core concept in the field of reservoir computing (Jaeger, 2001; Lukosevicius, 2012; Manjunath and Jaeger, 2013). Because the conceptor approach borrows some ideas from reservoir computing, I refer to the RNN as the *reservoir*.

Training a readout. The conceptor-based procedures explained below lead to a memory functionality in the following sense:

- At storing time, the reservoir is driven by a to-be-stored pattern $p^j(n)$ through (1). This results in a pattern-driven network dynamics $\mathbf{x}^j(n)$ (I use upper indices to refer to patterns).
- At recall time, the reservoir is run without input (under the control of a conceptor C^j, to be explained below), resulting in a free-running network dynamics $\tilde{\mathbf{x}}^j(n)$.
- The network has successfully "memorized" $p^j(n)$ to the degree that the free-running dynamics $\tilde{\mathbf{x}}^j(n)$ is approximately the same as the original pattern-driven dynamics $\mathbf{x}^j(n)$.

Thus, the network does not in fact memorize the original pattern $p^j(n)$, but instead its own, high-dimensional dynamical response to the pattern input. I will refer to such input-driven reservoir state dynamics as *state patterns*. But for practical exploits one wishes to recall the original input pattern, not the induced neural state pattern. In order to transform state patterns $\tilde{\mathbf{x}}^j(n)$ back to the original input patterns $p^j(n)$, a network state observer $y(n)$ is trained. This is a linear output neuron which reads from the network state through output weights W^{out}. It should display the following behavior

$$\begin{aligned} \textbf{if} \quad & \mathbf{x}(n+1) \quad = \tanh(W^* \mathbf{x}(n) + W^{\text{in}} p(n) + \mathbf{b}) \\ \textbf{then} \quad & y(n) \quad := W^{\text{out}} \mathbf{x}(n) \approx p(n) \end{aligned} \tag{2}$$

for *any* driving pattern $p(n)$. That is, the output signal read through W^{out} should simply re-generate any driving input from the excited reservoir state.

The output weights W^{out} for such a generic input-redisplayer neuron can be trained by first driving the reservoir with an M-dimensional white-noise input signal $\nu(n)$ via $\mathbf{x}^\nu(n+1) = \tanh(W^* \mathbf{x}^\nu(n) + W^{\text{in}} \nu(n) + \mathbf{b})$, then compute W^{out} by linear regression to minimize the quadratic loss $\sum_n (W^{\text{out}} \mathbf{x}^\nu(n) - \nu(n))^2$, following the rationale of reservoir computing. Since this readout neuron is trained on white-noise input $\nu(n)$, it will also be able to re-generate other input signals $p(n)$ in agreement with (2).

I point out that this pattern-generic output neuron is trained prior to, and independent of, the subsequent memory learning. It is also possible to train the output weights W^{out} on network states induced by the to-be-stored input patterns, not by a generic white-noise input. That procedure gives more accurate results at recall time and therefore will usually be the preferred method. However, it obscures the essential independence of the storing procedure from the observer training. Therefore in this didactic demo I used white noise input to train W^{out}.

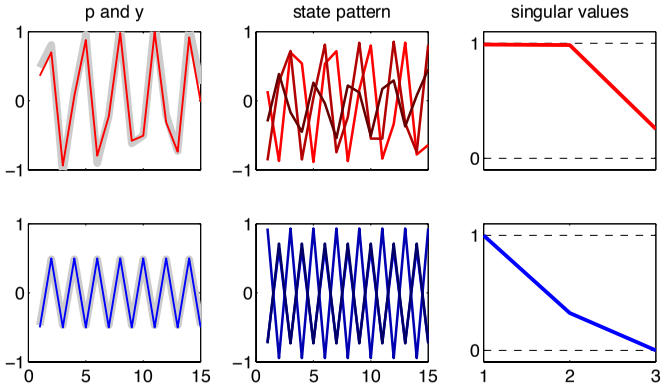


Figure 1: Left: the original patterns $p^1(n), p^2(n)$ (red/blue thin lines, 15 time steps are shown) and their recalled versions (gray thick lines; original and recalled patterns were phase-aligned for plotting). The patterns are discrete-time signals; connecting lines are drawn for better visual appearance. Center: traces of the three neurons' activations when the reservoir is driven with the respective pattern input. Right: singular value spectra of C^1 and C^2.

2.1.2 Storing Patterns

Loading the reservoir with the training patterns. I proceed to describe how the two patterns $p^1(n), p^2(n)$ of our demo are stored in the reservoir. In intuitive terms, storing patterns p^j amounts to re-compute the initial random reservoir weights W^*, giving a new set of network weights W, such that the new reservoir can mimic the impact of drivers p^j in the absence of them. I refer to the new weights W as "input internalization weights" on the grounds that they incorporate the impact of an external input into the autonomous network update dynamics.

W is computed as follows. In two separate runs, the two patterns are fed into the initial reservoir via

$$\mathbf{x}^j(n+1) = \tanh(W^* \mathbf{x}^j(n) + W^{\text{in}} p^j(n) + \mathbf{b}), \quad j = 1, 2;\ n = 0, \ldots, L. \tag{3}$$

Figure 1 (middle panels) shows the activation traces of the three neurons when the reservoir is driven with either pattern.

Then W is computed to minimize the quadratic loss

$$\sum_{j=1,2} \sum_{n=n_0+1,\ldots,L} \|W^* \mathbf{x}^j(n) + W^{\text{in}} p^j(n) - W \mathbf{x}^j(n)\|^2, \tag{4}$$

where only network states for times after n_0 are used (in order to allow for washing out the arbitrary intial state $\mathbf{x}(0)$ according to the echo state property). This again amounts

to a linear regression. If this is achieved with a small training error, one will obtain similar network updates from states $\mathbf{x}^1(n)$ or $\mathbf{x}^2(n)$ with either the original input-driven update rule, or with an input-free update rule that employs W instead of W^*:

$$\tanh(W^* \mathbf{x}^j(n) + W^{\text{in}} p^j(n) + \mathbf{b}) \approx \tanh(W \mathbf{x}^j(n) + \mathbf{b}). \quad (5)$$

I call this procedure to transform the initial random weights W^* to W *loading* the patterns into the reservoir.

Linear regressions can be computed with various algorithms. In all simulations reported below I use ridge regression. Given a collection of L argument-target vector pairs $(\mathbf{a}_i, \mathbf{t}_i) \in \mathbb{R}^\nu \times \mathbb{R}^\mu$, ridge regression computes a transformation matrix $M \in \mathbb{R}^{\mu \times \nu}$ which minimizes the regularized mean square error $1/L \sum_i \|\mathbf{t}_i - M \mathbf{a}_i\|^2 + \|\varrho M\|^2_{\text{fro}}$, where $\| \cdot \|^2_{\text{fro}}$ is the squared Frobenius matrix norm and ϱ the Tychonov regularization coefficient.

In the case of (4), $\nu = \mu = N$ and the arguments are all $\mathbf{x}^j(n)$ and the targets are the corresponding $W^* \mathbf{x}^j(n) + W^{\text{in}} p^j(n)$.

The re-computation of initial weights of a RNN to obtain what I called here input internalization weights is a procedure that has been independently proposed several times in recent years under different names and for different purposes: as *self-predicting networks* for augmenting the performance of reservoir computing techniques (Mayer and Browne, 2004), as *equilibration* for enabling external controllability of RNN dynamics (Jaeger, 2010), as *reservoir regularization* for improved stability of neural motor controllers (Reinhart and Steil, 2011), as *self-sensing networks* for making RNN training methods more flexible (Sussillo and Abbott, 2012), and as *innate training* for reliable, noise-resistant reproducible chaotic patterns in short-term memory RNNs (Laje and Buonomano, 2013).

Lower cost variants of the loading procedure. In a variant of the loading procedure, only the input term $W^{\text{in}} p^j(n)$ is replaced, leaving W^* unchanged. That is, the input-driven network state $\tanh(W^* \mathbf{x}^j(n) + W^{\text{in}} p^j(n) + \mathbf{b})$ becomes emulated by the autonomous states $\tanh(W^* \mathbf{x}^j(n) + D \mathbf{x}^j(n) + \mathbf{b})$, where D is computed by linear regression to minimize the following variant of the quadratic loss (4):

$$\sum_{j=1,\ldots,K} \sum_{n=n_0+1,\ldots,L} \|W^{\text{in}} p^j(n) - D \mathbf{x}^j(n)\|^2. \quad (6)$$

I call the $N \times N$ matrix D *input simulation weights*. Another variant of the loading procedure is even more minimalistic and aims at replacing only the very input $p^j(n)$, by finding *input replacing weights* R (size $N \times M$) that minimize

$$\sum_{j=1,\ldots,K} \sum_{n=n_0,\ldots,L} \|p^j(n) - R \mathbf{x}^j(n)\|^2,$$

leading to autonomous states $\tanh(W^* \mathbf{x}^j(n) + W^{\text{in}} R \mathbf{x}^j(n) + \mathbf{b})$.

Computing conceptors. Considering (5), the loaded reservoir should be able to generate approximate versions of the two original state patterns. However, if the network were run just by iterating $\mathbf{x}(n+1) = \tanh(W \mathbf{x}(n) + \mathbf{b})$, the resulting input-free reservoir dynamics is entirely unpredictable because the reservoir can't "decide" which of the loaded pattern dynamics it should engage in.

Here conceptors enter the stage. Each loaded pattern p^j is associated with an $N \times N$ sized *conceptor* matrix C^j which at recall time is inserted into the state update loop via

$$\mathbf{x}(n+1) = C^j \tanh(W \mathbf{x}(n) + \mathbf{b}).$$

The matrix C^j acts as a filter that leaves states $\mathbf{x}^j(n)$ from the state pattern associated with pattern p^j essentially unchanged, but suppresses state components of states $\mathbf{x}^{j'}(n)$ associated with other patterns $p^{j'}$. Stated differently, C^j should act like the identity matrix for states $\mathbf{x}^j(n)$, but like the null matrix for state components that are not typical for states $\mathbf{x}^j(n)$.

This consideration leads to a quadratic loss function

$$L(C^j) = E[\|C^j \mathbf{x}^j(n) - \mathbf{x}^j(n)\|^2] + (\alpha^j)^{-2} \|C^j\|^2_{\text{fro}}, \quad (7)$$

where the expectation E is taken over all states $\mathbf{x}^j(n)$ that arise in p^j-driven runs according to (3). Explanations:

- The first component $E[\|C^j \mathbf{x}^j(n) - \mathbf{x}^j(n)\|^2]$ of this loss function is minimal when C^j is the identity. This component reflects the objective that C^j should leave states $\mathbf{x}^j(n)$ unchanged.
- The second component $\|C^j\|^2_{\text{fro}}$ becomes minimial for $C^j = \mathbf{0}$. This takes care of the objective that C^j should suppress state components which are untypical of states $\mathbf{x}^j(n)$, i.e. such state components which do not enter the expectation of the first component.
- The machine learning view on the loss (7) is to consider it as the loss for a *regularized identity function*. It is mathematically closely related to computing a *denoising autoencoder* for state patterns corrupted by Gaussian noise with variance $(\alpha^j)^{-2}$.
- The parameter $\alpha^j \geq 0$, called *aperture* for reasons that will soon become clear, negotiates between the two objectives. When the aperture is large, C^j will be close to the identity matrix I. Conversely, for small apertures C^j will shrink toward the null matrix $\mathbf{0}$. In our example I use $\alpha^1 = 12$ and $\alpha^2 = 20$.

Minimizing the loss $L(C^j)$ leads to the solution

$$C^j = R^j (R^j + (\alpha^j)^{-2} I)^{-1}, \quad (8)$$

where $R^j = E[\mathbf{x}^j(n) \mathbf{x}^j(n)']$ is the $N \times N$ correlation matrix of states $\mathbf{x}^j(n)$ obtained in the state dynamics driven by pattern p^j. C^j has the following properties:

- C^j is positive semi-definite, with eigenvalues (= singular values) $0 \leq \sigma^j_i \leq 1$ ($i = 1, \ldots, N$).
- The N eigenvectors $\mathbf{u}^j_i$ of C^j are the same as the eigenvectors of R^j, and can be arranged column-wise in an orthonormal matrix $U^j = (\mathbf{u}^j_1 \cdots \mathbf{u}^j_N)$. These eigenvectors are the same as the principal component vectors obtained from a principal component analysis of state sets $\mathbf{x}^j(n)$.

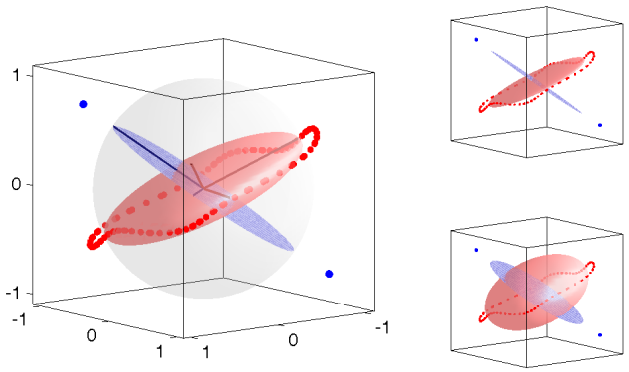


Figure 2: Conceptor geometry. Left: 3-dimensional state pattern $\mathbf{x}^1(n)$ (red dots) and $\mathbf{x}^2(n)$ (blue dots). Ellipsoids corresponding to conceptors C^1 and C^2 are rendered in red and blue respectively, with principal axes shown. They lie inside the unit sphere, shown in light gray. Right: effects of halving (top) and doubling (bottom) apertures.

I will often use the singular value decomposition (SVD) of $C^j = U^j S^j U^{j\prime}$, where S^j is the diagonal matrix containing the singular values σ_i^j on its diagonal, by convention arranged in descending order.

The derivation of (8) and the properties of C^j is elementary and can be found in Jaeger (2014). In practice, the correlation matrices R^j are estimated from states collected in training runs.

It is easily verified that $R\,(R+\alpha^{-2}I)^{-1} = \alpha^2 R\,(\alpha^2 R+I)^{-1}$. Because $\alpha^2 R = E[\alpha\mathbf{x}\,(\alpha\mathbf{x})']$, the aperture α can also be understood as a virtual scaling factor of reservoir states.

2.1.3 Geometry of Conceptors

It is instructive to contemplate the geometry of conceptors C^j and how it relates to the state dynamics $\mathbf{x}^j(n)$. The main panel in Figure 2 shows 100 instances of $\mathbf{x}^1(n)$ (red dots), which densely fill a cyclic path because of the irrational ratio between the driving sinewave period and the sampling interval. The resulting conceptor C^1 can be visualized by an ellipsoid centered at the origin. Its principal axes are the eigenvectors of C^1 scaled by their singular values (they are indicated in Figure 1, right panels). Since the singular values of conceptor matrices range in $[0, 1]$, such ellipsoids lie inside the unit sphere. The singular values of C^1 are all nonzero, hence the ellipsoid is non-degenerate and extends in three directions. The state pattern $\mathbf{x}^2(n)$ induced by the 2-periodic driver p^2 alternates between two states

(blue dots). These two states span a 2-dimensional subspace of $\mathbb{R}^3$: only two of the singular values of C^2 are nonzero, and the resulting ellipsoid is a degenerate (2-dimensional only).

The two small panels illustrate the geometrical effects of adjusting aperture. Increasing the aperture "widens" conceptors, while decreasing aperture lets conceptors contract.

2.1.4 Pattern Re-generation

The loaded reservoir will behave unpredictably when run via $\mathbf{x}(n+1) = \tanh(W\,\mathbf{x}(n) + \mathbf{b})$, but it will engage in the j-th state pattern $\mathbf{x}^j(n)$ when run with the corresponding conceptor in the loop via $\mathbf{x}(n+1) = C^j \tanh(W\,\mathbf{x}(n) + \mathbf{b})$. The conceptor C^j acts as a state filter which leaves states belonging to pattern j mostly unaffected, while other states that would belong to other patterns are projected into the ellipsoidal state volume typical for pattern j.

The original input pattern can be read from this dynamics by $W^{\text{out}}\,\mathbf{x}(n) \approx p^j(n)$. Figure 1 (left panels) shows the re-generated patterns under conceptor control overlaid with the original drivers.

2.1.5 Intuitive Summary, and Relationship to Reservoir Computing

Three kinds of matrices are trained in the conceptor-based pattern learning setup: the readout weights, the recurrent reservoir weights, and the conceptor matrices. How can the respective functional roles of these three components be intuitively understood?

In the loading process, when the random initial reservoir weights W^* are recomputed into W, information about the detailed dynamics of each training pattern is imprinted on the reservoir. A helpful metaphor is to liken the loading process to driving a vehicle over Sahara sands. Each pattern digs its individual track into the surface of the native sandscape. The original reservoir with weights W^* is like a virgin sandscape. After loading K patterns, K tracks have been engraved into the surface. These tracks are coded in the weight matrix W.

Still following that metaphor, the pattern tracks imprinted on the sand will have crossings. Sending a vehicle on a track-following mission, it wouldn't know where to turn at those crossings. If one would just run a network simulation with loaded weights W without further ado, the resulting network dynamics is unpredictable: the network doesn't know where to turn at the crossings; typically one observes a trajectory that appears like high-dimensional chaos or a trajectory that gets stuck in a point attractor.

This is where conceptors come into play. If conceptor C^j is inserted into the network update loop, it informs the network how to follow track number j. Figuratively speaking, the conceptor flattens out competing tracks but leaves track j mostly as it is. More technically speaking, this "flattening out" is effected by the large number of zero or close-to-zero singular values of C^j. In memory terminology, the conceptor acts as a *selector* for one of the loaded patterns. As we will see later on, as a side effect this "flattening out" endows the network dynamics with strong noise-suppression and dynamical stabilization characteristics.

Finally, the readout weights are not a crucial component in the conceptor approach to RNNs. This is different from the usual perspective of reservoir computing where the training of readout weights is the defining part of the entire paradigm. The focus of conceptor theory is *representation learning*, not output generation. It is all about creating representations

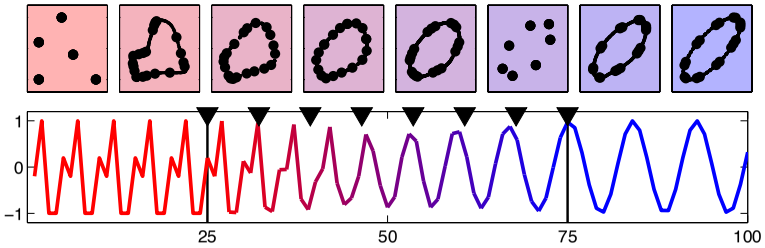


Figure 3: Morphing from an integer-5-periodic pattern p^1 to an irrational-period sine p^2 from time $n = 25$ to $n = 75$. Top: delay-embedding plots of signals $y(n)$ obtained with fixed conceptor mixes (indicated by triangle markers in center panel). Bottom: Output signal $y(n)$.

of dynamical patterns inside an RNN, where these representations (the tracks in the sand) become serviceable through conceptors. The readout weights merely provide an externally interpretable *observer* for the learnt representations, and a means to quantify the quality of the internal representations by measuring the mismatch between original patterns and re-generated output patterns.

2.2 Morphing and Generalization

When K "prototype" patterns $p^1, \ldots, p^K$ have been loaded, they can be *morphed* in recall by using linear blends of their conceptors via

$$\mathbf{x}(n+1) = (\sum_{j=1,\ldots,K} a^j\, C^j)\, \tanh(W\,\mathbf{x}(n) + \mathbf{b}), \quad y(n) = W^{\text{out}}\,\mathbf{x}(n), \tag{9}$$

where $\sum_{j=1,\ldots,K} a^j = 1$. For an elementary demonstration, an integer-5-periodic pattern p^1 and a sine pattern p^2 with an irrational period length of about 8.8 (sampled at integer time points) were loaded into a 100-neuron reservoir. Figure 3 (bottom) displays the output $y(n)$ of a recall run where for the first 25 steps, the pure conceptor C^1 was used, then for a morphing time of 50 steps C^1 was linearly blended into C^2, concluding the run with another 25 steps under the control of the pure C^2. The output signal exhibits a gradual blending from p^1 to p^2 during the morphing period from step 25 to step 75. In order to get more insight into the morphing dynamics, for eight mixing ratios (marked by triangle markers in the bottom panel), the system (9) was run separately with mixing coefficients frozen at these ratios. The panels in the top row show "fingerprints" of the resulting output signals $y_i(n)$, by plotting delay embedding state points $(y_i(n+1), y_i(n))$. Each panel shows 20 successive points as thick dots, and further 100 successive points as thin dots. The first panel, which corresponds to the pure 5-periodic pattern p^1, shows 5 cyclically repeated points, and the last panel a quasi-periodic oscillation, as expected. Intermediate dynamics are mostly quasi-periodic, but there are also mixture settings where there is a 6-, 7-, and

8-periodic behavior (of which only the 7-periodic instance happens to be captured by one of the displayed plots).

In a slightly more involved demonstration three representative patterns p^1, p^2, p^3 are loaded, which are taken from a two-parametric family of modulated sinewave patterns. One parameter changes the frequency, the other modulates the shape from "pointed upwards" to "pointed downwards" (details in the appendix). Figure 4 shows a close-up of the three loaded patterns in the separate top panels (red: slightly pointed downwards, high frequency; green: slightly pointed downwards, low frequency; blue: slightly pointed upwards, intermediate frequency). Note that three is the minimal number of patterns required to span a 2-parametric pattern space. To visualize the morphs, pattern samples generated with morphed conceptors $a^1\,C^1 + a^2\,C^2 + a^3\,C^3$ are arranged in a tesselated drawing plane. The "pure" recalled patterns (corresponding to mixtures $(a^1, a^2, a^3) = (1,0,0); (0,1,0); (0,0,1)$ respectively) are placed at the corners of an equilateral triangle (saturated red/green/blue tiles in the main figure panel). Tiles within the triangle spanned by the three loaded patterns correspond to interpolated conceptors, whereas patterns outside the triangle were obtained by *extrapolating* conceptor mixtures with coefficents outside [0 1].

By visual inspection of the signals shown in Figure 4 one finds that the system has acquired command over a sizeable portion of the parametrized pattern space by generalizing from three training instances. This generalization is effective not only in the "convex hull" of the three training examples (interpolation area within the central pattern triangle in the figure), but extends far into the extrapolation range. Going right/left from the three training patterns one sees that the mild frequency modulation indicated by the three training patterns reaches out into a wider frequency span; and going up/down one finds the "pointed up/down" characteristics, which is only feebly present in the training patterns, becomes fully expressed.

But one should not expect miracles. Upon closer inspection one will find that many re-generated patterns are not precise members of the family. Deviations grow as the mixture coefficients extend further away from the interpolation region. Also, there is no linear relationship between variations in mixing coefficients and geometric pattern characteristics. Far outside the interpolation region similiarities with correct patterns from the target family dissolve. If one wishes to obtain a wider-range faithful re-generation of patterns from the family, a larger number of reference patterns must be loaded. In Section 3.2 we will meet an altogether different way to employ conceptors to learn a pattern family from a small number of training examples.

2.3 A Real-World Data Example

In order to illustrate that the approach scales to more complex patterns, fifteen diverse human motion patterns were loaded: *slow walk*; *fast walk*; *walk with exaggerated stride*; *jog*; *sit*; *get up from stool*; *sit down on stool*; *kneel down*; *crawl*; *get up from crawl*; *waltz*; *cartwheeling*; and three *boxing punches*. These patterns are 61-dimensional signals (body pose and joint angles). Some are periodic (like *jog*), others are transient (like *get up from crawl*), yet others are irregular-stochastic (the *boxing* patterns); some patterns have multiple timescales. Training data consisted of short sequences clipped from publicly available human motion capture traces (CMU Graphics Lab)), one such clip per motion type. The length

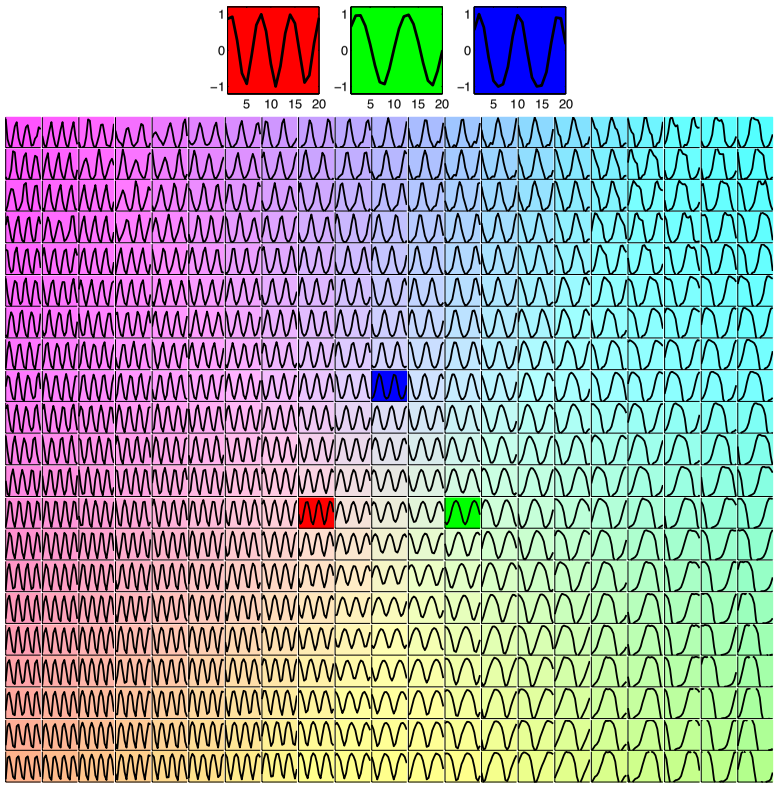


Figure 4: Morphing between and beyond three patterns from a 2-parametric family. For explanation see text.

of training patterns varied from 150 to 900 sampling points (sampling rate 120 / second). Data preprocessing and visualization was done with the help of a public Matlab motion capture toolbox (Burger and Toiviainen, 2013). Figure 5 gives an impression of the training data after preprocessing.

A 600-unit reservoir made from leaky-integrator neurons was loaded with the 15 training samples (details in the appendix). In a recall run, the 15 patterns were re-generated in a shuffled sequence according to a hand-designed choreography. In this sequence, a pattern p^j is reproduced by inserting the corresponding conceptor C^j into the reservoir update loop for some period of time. In order to achieve a smooth transition into the next motion $p^{j'}$, C^j was morphed into the following conceptor $C^{j'}$ for one simulated second before $C^{j'}$ takes over full

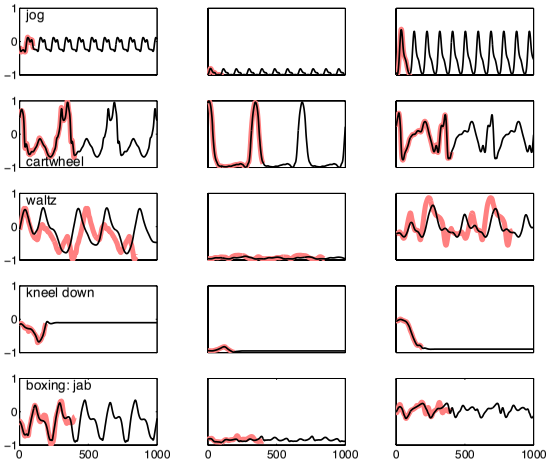


Figure 5: Exemplary human motion training data. Plots show the range-normalized versions used as input for the reservoir. The four rows show four out of the 15 loaded patterns. The three columns show three out of the 61 data dimensions. Thick red lines: training data; black lines: 1000-step free-running pattern recall under conceptor control. For explanation see text.

control. The resulting choreographed recall sequence of one minute duration was rendered into a video. The video is on YouTube (`www.youtube.com/watch?v=DkS_Yw1ldD4`). Figure 6 shows a few snapshots.

Morphing and generalizing dynamical patterns is a common but nontrivial task for training motor patterns in robots. It typically requires training demonstrations of numerous interpolating patterns (Reinhart and Steil, 2008; Coates et al., 2008; Lukic et al., 2012). Conceptor-based pattern learning and morphing appears promising for flexible robot motor pattern learning from a very small number of training patterns. I want to emphasize that this is only a method for "playback" of learnt trajectories with flexible sequence scheduling and motion blending with potential applications in computer game character animation; it is not a method for motor *control*. The procedure demonstrated here is visually superior to some existing neural-network based approaches (wyffels and Schrauwen, 2009; Sutskever et al., 2009; Boström et al., 2013; Wright and Jordanov, 2012) with respect to number of patterns, body model complexity, and smoothness of transitions. The mocap-trained human motion generation system presented by Taylor et al. (2011), which is based on restricted Boltzmann machines, comes close to the conceptor-based demo with respect to number of patterns and body model complexity, and has visually more appealing transitions between gaits.

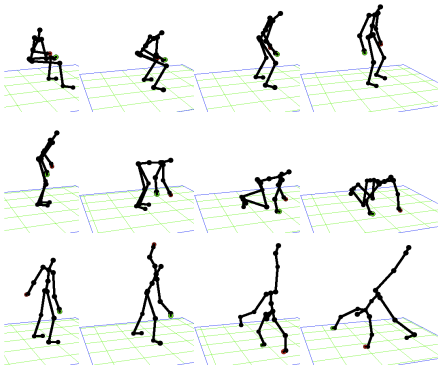


Figure 6: Snapshots from a human motion re-generation video created from a conceptor-controlled RNN. First row: standing up from a stool and starting a slow walk, second row: kneeling down and starting to crawl, third row: cartwheel.

I emphasize that this case study is solely meant as a visually appealing demo that conceptor-controlled pattern generation can scale to sizable numbers of high-dimensional, non-stationary patterns. In this it serves the same purpose as the other motion capture learning neural networks that I referenced above. It is neither a method for motor *control* in physical robots, nor does it aspire to compete with state-of-the-art game character animation engines, which each combine a diversity of sophisticated animation techniques, often including physics simulations (Gillies and Spanlang, 2010).

2.4 Aperture Adaptation

For good performance in recall, the aperture must be chosen appropriately. To illustrate this, I loaded four patterns into a $N = 500$ reservoir. The patterns were obtained from four classical chaotic attractors, p^1: Lorenz attractor; p^2: Rössler attractor; p^3: Mackey-Glass attractor; p^4: Hénon attractor (details in the appendix). Green plots in Fig. 7**A**,**B** visualize the clean training signals. In the reproduction stage, for each pattern p^j a number of different conceptors C^j with varied apertures were tried. Fig. 7**A** shows the effects for the Lorenz attractor. When α is too small, the reservoir-conceptor feedback loop becomes too constrained and the generated patterns de-differentiate. With α too large the feedback loop becomes over-excited. In this demonstration, a good aperture could be automatically found by minimizing a measurable that I call *attenuation*. This is the fraction of the reservoir signal energy which is suppressed by applying the conceptor. Formally, the attenuation $a_{C,\alpha}$ induced by a conceptor C at aperture α is

$$a_{C,\alpha} = E[\|\mathbf{r}(n) - \mathbf{z}(n)\|^2]/E[\|\mathbf{r}(n)\|^2],$$

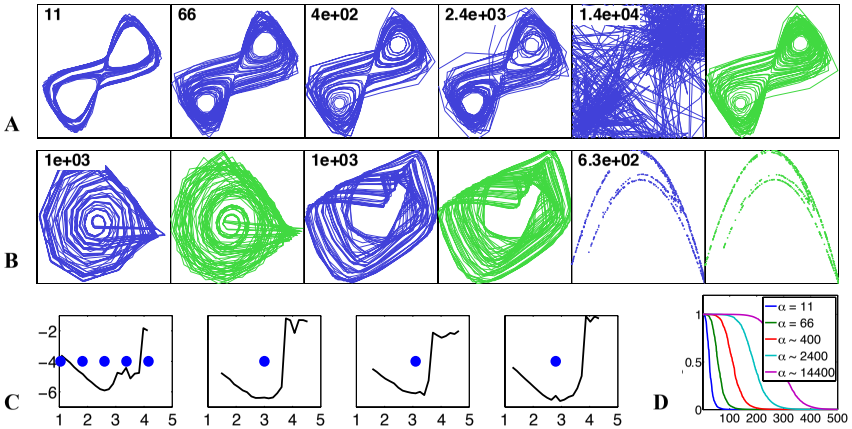


Figure 7: Reproducing four chaotic attractor patterns in a single network. Green: original training signals, blue: network-generated reproductions. The 2-dimensional plots are obtained from 1-dimensional timeseries by delay embedding, plotting $y(n)$ against $y(n-d)$ for a delay d chosen to yield instructive graphics. (**A**) The Lorenz attractor re-generated with five different aperture settings (α values inserted in panels). (**B**) From left to right: re-generations of the the Rössler, Mackey-Glass, and Hénon attractors. (**C**) Log10 of attenuation plotted against log10 of aperture. Dots mark the apertures used in the reproductions in (**A**,**B**). (**D**) Singular value spectra of conceptors used for the Lorenz attractor (compare panel **A**).

where $\mathbf{r}(n) = \tanh(W\mathbf{z}(n-1)+\mathbf{b})$, $\mathbf{z}(n) = C\,\mathbf{r}(n)$. This quantity can be cheaply estimated online and can be used to automatically adapt the aperture at recall time. Fig. 7**C** plots attenuation against aperture. The minimum of this curve marks a good aperture value, where "good" means "good visual agreement of recalled attractor with original attractor" (I did not implement an attractor comparison metric). The apertures used for the reproductions shown in Fig. 7**B** were obtained via this minimum-attenuation criterion.

While placing the aperture roughly in the right range is important, fine-tuning is unnecessary. The Lorenz attractor shown in Fig. 7**A** is quite well reproduced with apertures ranging over two orders of magnitude. This is also true for the other chaotic attractor signals, as well as for all other simulation studies that I undertook so far. While this 4-pattern demo is an academic exercise, it underlines the functionality of conceptors to stabilize neural pattern generation.

3. Autoconceptors

The definition and usage of conceptors, as described so far, invites an obvious critique: conceptor matrices have the same size as the reservoir weight matrix. Thus, for representing a pattern p^j by a conceptor matrix C^j one has to create and store an object that is as heavyweight as the hosting reservoir—one might say that each new pattern adds another brain. This is clearly inappropriate for computational neuroscience modeling, and it may become too expensive in machine learning applications. The storing cost of adding a conceptor matrix can be reduced by storing only economical SVDs of conceptor matrices, which often have low numerical rank. This however only mitigates but does not solve the principal problem.

I can offer two answers to this important critique:

Conceptor auto-adaptation and content-addressable memories. At loading time, when pattern internalization or pattern simulation weights are computed, no conceptors are created. Pattern recall functions by content-addressing: a short / incomplete cue of the target pattern is presented and an appropriate conceptor is generated on the fly by an auto-adaptation process. This leads to a memory model which is reminiscent of Hopfield networks. The present section is devoted to such auto-adapting conceptors.

Diagonal conceptors with single-neuron representations. When conceptor matrices are constrained to diagonal form, filtering a reservoir state by a conceptor reduces to unit-wise scalar multiplications, and a conceptor can be mathematically represented by a vector, which in turn could possibly be biologically implemented by a single neuron. This venue is explored in some detail in Jaeger (2014), where also proof-of-principle demos are presented of multi-reservoir hierarchical memory systems that are based on such diagonal conceptors. I am however not satisfied with the robustness properties of diagonal conceptors (they are too parameter-sensitive), and I am currently working on improved versions. Therefore I will not report on the available material from Jaeger (2014) this article.

3.1 Conceptor Auto-Adaptation

In the preceding sections I have defined conceptors as transforms $C^j = R^j\,(R^j + \alpha^{-2} I)^{-1}$ of reservoir state correlation matrices R^j (Equation 8) obtained from driving the reservoir with pattern p^j. The conceptor C^j could then be exploited for pattern re-generation via $\mathbf{x}(n+1) = C^j\,\tanh(W\mathbf{x}(n)+\mathbf{b})$ (using input internalization weights) or its variant $\mathbf{x}(n+1) = C^j\,\tanh(W^*\,\mathbf{x}(n) + D\,\mathbf{x}(n) + \mathbf{b})$ (using input simulation weights).

This way of using conceptors, however, requires that the conceptor matrices C^j are computed at loading time, and they have to be stored in some way for usage at exploitation time. Such a procedure may be useful in some practical engineering respects (for instance, for dynamical pattern morphing and pattern stabilization), but otherwise invites the critique pointed out at the beginning of this section.

This motivates to look for ways of how conceptors can be used for constraining reservoir dynamics without the necessity to create and store conceptor matrices beforehand. The network would have to create the requisite conceptors on the fly by some auto-adaptation process while it is performing some relevant task. In this section I investigate auto-adapted

conceptors for the task of realizing a content-addressable memory. They are also useful for other tasks—in Jaeger (2014) (Section 3.15) I demonstrate the use of such auto-adapted conceptors for the task of simultaneous signal denoising and classification.

For ease of discussion, I call conceptors created by auto-adaptation at use-time *autoconceptors*, as opposed to the conceptors which are created at pattern loading time and which are externally inserted into the reservoir update loop at use-time—I will refer to those as *alloconceptors*. Autoconceptors, like alloconceptors, are positive semidefinite matrices with singular values in the unit interval. Aperture operations are identical for allo- and autoconceptors. However, the way how autoconceptors are generated by auto-adaptation leads to additional constraints on their algebraic characteristics. The set of autoconceptor matrices is a proper subset of the set of alloconceptor matrices.

3.1.1 Basic Equations

The basic system equation for autoconceptor systems is

$$\mathbf{x}(n+1) = C(n)\,\tanh(W^*\,\mathbf{x}(n) + W^{\text{in}} p(n+1) + \mathbf{b}) \tag{10}$$

or variants thereof, like

$$\mathbf{x}(n+1) = C(n)\,\tanh(W\,\mathbf{x}(n) + \mathbf{b}) \tag{11}$$

or

$$\mathbf{x}(n+1) = C(n)\,\tanh(W^*\mathbf{x}(n) + D\mathbf{x}(n) + \mathbf{b}), \tag{12}$$

the latter two capturing the situation after having patterns loaded by input internalization or input simulation weights. The novel element in these equations is that $C(n)$ is time-dependent. Its evolution is governed by an adaptation rule that I will describe presently. $C(n)$ need not be positive semidefinite at all times; only after convergence the $C(n)$ matrices will have the algebraic properties of conceptors.

It is convenient to formally split the network state $\mathbf{x}$ in (10) into a "reservoir state" $\mathbf{r}$ measured directly after the neuron nonlinearity, and a "filtered" state $\mathbf{z}$ obtained after passing the reservoir state through the conceptor. This turns (10) into

$$\begin{aligned} \mathbf{r}(n+1) &= \tanh(W^*\mathbf{z}(n) + W^{\text{in}} p(n+1) + \mathbf{b}) &(13)\\ \mathbf{z}(n+1) &= C(n)\,\mathbf{r}(n+1). &(14)\end{aligned}$$

The auto-adaptation of $C(n)$ aims at minimizing a loss that at first sight looks the same as the loss (7):

$$L(C) = E[\|C\,\mathbf{z}(n) - \mathbf{z}(n)\|^2] + \alpha^{-2}\,\|C\|^2_{\text{fro}}, \tag{15}$$

which for alloconceptors was solved by $C = R\,(R + \alpha^{-2}\,I)^{-1}$ with $R = E[\mathbf{x}\,\mathbf{x}']$ being the autocorrelation matrix of reservoir states. Similarly, (15) is minimized by

$$C = R\,(R + \alpha^{-2}\,I)^{-1},\ \text{with}\ R = E[\mathbf{z}\,\mathbf{z}']. \tag{16}$$

The crucial difference is that now the state correlation matrix R depends on C:

$$R = E[\mathbf{z}\,\mathbf{z}'] = E[C\mathbf{r}\,(C\mathbf{r})'] = C\,E[\mathbf{r}\,\mathbf{r}']\,C =: CQC,$$

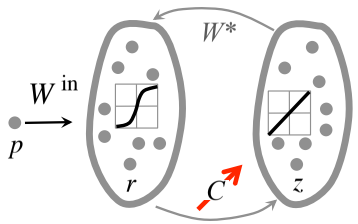


Figure 8: Network representation of a basic autoconceptor system. Bias b and readout mechanisms are omitted. The red arrow indicates that C is adapted online. For explanation see text.

where we introduce $Q = E[\mathbf{r}\,\mathbf{r}']$. This transforms the explicit solution formula (8) into a fixed-point equation:

$$C = CQC\,(CQC + \alpha^{-2}\,I)^{-1}. \tag{17}$$

Since Q depends on $\mathbf{r}$ states, which in turn depend on $\mathbf{z}$ states, which in turn depend on C again, Q depends on C and should be more appropriately be written as Q_C. A formal analysis of solutions to the fixed-point equation $C = CQ_CC\,(CQ_CC + \alpha^{-2}\,I)^{-1}$ is involved (carried out in some detail in Jaeger (2014)). Explicit solution formulas are unlikely to exist because the nonlinear reservoir is involved in Q_C. When one uses autoconceptors, however, one does not need to solve (17) explicitly. Instead, one can invoke stochastic gradient descent to minimize the loss (15). It is easily derived (Jaeger, 2014) that

$$C(n+1) = C(n) + \lambda\,\left((\mathbf{z}(n) - C(n)\,\mathbf{z}(n))\,\mathbf{z}'(n) - \alpha^{-2}\,C(n)\right) \tag{18}$$

implements stochastic gradient descent with respect to the loss (15). Here λ is an adaptation rate. Equations (13), (14) and (18) taken together—or versions where (13) is replaced by (11) or (12)—define the joint state update and conceptor adaptation working cycle.

In Jaeger (2014) I show that, if the driver $p(n)$ is a stationary process and if $C(n)$ converges under this auto-adaptation rule, the limit C is positive semidefinite with singular values in the set $(1/2, 1) \cup \{0\}$. Singular values of C asymptotically obtained under the evolution (18) are either greater than $1/2$ or they are zero. If the aperture α is fixed at increasingly smaller values, increasingly many singular values converge to zero. Furthermore, the analysis in Jaeger (2014) reveals that among the nonzero singular values, the majority will be close to 1. Both effects together endow autoconceptors with singular value spectra that are typically approximately rectangular.

3.1.2 Basic Demonstrations

In order to display how autoconcepters can be used to realize a content-addressable memory, I ran simulations according to the following scheme:

1. *Loading.* A collection of 10 scalar patterns p^j was loaded in an N-dimensional reservoir, yielding an input simulation matrix D and readout weights W^{out}. More specifically, D was computed to minimize the loss (6), using states $\mathbf{x}^j(n)$ obtained from driven runs $\mathbf{x}^j(n+1) = \tanh(W^*\,\mathbf{x}^j(n) + W^{\text{in}}\,p^j(n) + \mathbf{b})$. No conceptors were involved or computed in the loading procedure.

2. *Recall.* For each pattern p^j, a recall run was executed which consisted of three stages:

   (a) *Initial washout.* Starting from a zero network state, the reservoir was driven with p^j for n_{washout} steps, in order to obtain a task-related reservoir state.

   (b) *Cueing.* The reservoir was continued to be driven with p^j for another n_{cue} steps. During this cueing period, C^j was adapted through $\mathbf{z}(n+1) = \tanh(W^*\mathbf{z}(n) + W^{\text{in}}p(n) + \mathbf{b})$, $C^j(n+1) = C^j(n) + \lambda^{\text{cue}}\,((\mathbf{z}(n) - C^j(n)\,\mathbf{z}(n))\,\mathbf{z}'(n) - \alpha^{-2}\,C^j(n))$. At the beginning of this cueing period, $C^j(0)$ was initialized to the zero matrix. Notice that during cueing, the nascent conceptor $C^j(n)$ was not inserted in the reservoir state update loop. At the end of this period, a conceptor $C^{j\,\text{cue}}$ was obtained.

   (c) *Autonomous auto-adaptation and pattern recall.* The network run was continued with the external input switched off, using $\mathbf{z}(n+1) = C^j(n)\,\tanh(W^*\mathbf{z}(n) + D\mathbf{z}(n) + \mathbf{b})$, while continuing to auto-adapt the conceptor through $C^j(n+1) = C^j(n) + \lambda^{\text{recall}}\,((\mathbf{z}(n) - C^j(n)\,\mathbf{z}(n))\,\mathbf{z}'(n) - \alpha^{-2}\,C^j(n))$. At three test time points t_1, t_2, t_3 the conceptor $C^j(t_i)$ in its current adaptation stage was recorded for an offline quality assessment.

3. *Measuring the quality of conceptors.* The quality of the conceptors $C^{j\,\text{cue}}$ and $C^j(t_i)$ (where $i = 1, 2, 3$) was measured in separate offline runs without conceptor adaptation using $\mathbf{z}(n+1) = C\,\tanh(W^*\mathbf{z}(n) + D\mathbf{z}(n) + \mathbf{b})$, where C was one of $C^{j\,\text{cue}}, C^j(t_1), C^j(t_2), C^j(t_3)$. A reconstructed pattern $y(n) = W^{\text{out}}\,\mathbf{z}(n)$ was obtained and its similarity with the original pattern p^j was quantified in terms of a NRMSE.

I carried out two instances of this experiment, using two kinds of patterns:

5-periodic pattern. The patterns were random integer-periodic patterns of period 5. Experiment parameters: Reservoir size $N = 100$, spectral radius of W^* set to 1.5, input weight scaling 1.5, bias weight scaling 0.5, ridge regularization coefficients $\varrho^2_D, \varrho^2_{W^{\text{out}}}$ both set to 0.0001, aperture $\alpha = 1000$, $n_{\text{washout}} = 20$, $n_{\text{cue}} = 10$, $t_1, t_2, t_3 = 10, 50, 500$, auto-adaptation rates $\gamma^{\text{cue}} = 0.02$, $\gamma^{\text{recall}} = 0.01$. To make the task more challenging, during the 10-step cueing phase the cue input pattern was corrupted by additive uniform noise sampled from $[-0.05, 0.05]$, and during the autonomous recall time the reservoir states were *very strongly* perturbed by additive Gaussian noise (fed inside the tanh) whose variance equalled the average reservoir state component variance (a signal-to-noise ratio of 1).

2-parametric mix of 2 irrational-period sines. The 10 patterns were were taken from the 2-parametric family of patterns governed by

$$p(n) = a\,\sin(2\,\pi\,n/P) + (1-a)\,\sin(4\,\pi\,(b + n/P)).$$

These signals are weighted sums of two sines, the first with period length P and the second with period length $P/2$. The weights of these two components are a and $(1-a)$, and the second component is phase-shifted relative to the first by a fraction b of its period length $P/2$. The reference period length P was fixed to $P = \sqrt{30}$. The parameters a, b were freshly sampled from the uniform distribution on $[0, 1]$ for each p^j. Network parameters: $N = 100$, reservoir spectral radius 1.5, input weight scaling 1.5, bias scaling 0.5, $\varrho_D^2 = \varrho_{W^{\text{out}}}^2 = 0.0001$, $\alpha = 200, n_{\text{washout}} = 100, n_{\text{cue}} = 12$, $t_1, t_2, t_3 = 20, 1000, 10000$, $\gamma^{\text{cue}} = \gamma^{\text{recall}} = 0.01$. Cue patterns were corrupted by the same amount of additive noise as the 5-periodic patterns, and reservoir states were again perturbed with a signal-to-noise ratio of 1 during the autonomous recall.

Figure 9 illustrates the outcomes of these two experiments. For brevity I refer to the two experiments as the "IP5" and "PF" conditions. Observations and interpretations:

- The quality of the preliminary conceptor $C^{j\,\text{cue}}$ was always much improved by the subsequent auto-adaptation (panels **B**, **D**).
- The effects of autoconceptive adaptation are reflected in the singular value profiles of $C^{j\,\text{cue}}$ versus $C^j(t_3)$ (**A**, **C**). During the short cueing time, the online adaptation of the conceptor from a zero matrix to $C^{j\,\text{cue}}$ only manages to build a preliminary "nascent" profile, which then "matures" by auto-adaptation.
- The conceptors $C^j(t_3)$ have an almost rectangular singular value profile, in agreement with mathematical analyses (Jaeger, 2014).
- In PF the averge re-generation NRMSE with the final conceptor $C(t_3)$ was 0.11. For comparison I loaded and recalled the 10 patterns following the basic alloconceptor procedure with a manually optimized aperture. This gave an NRMSE of 0.087, that is, content-addressed autoconceptor recall from short and noisy cues here worked almost as well as re-generation with previously stored conceptors.
- For the 5-periodic patterns, a comparison simulation with the basic alloconceptor procedure yielded a mean NRMSE of .0033, much better than the mean NRMSE of 0.070 obtained after $t_3 = 500$ steps in the IP5 experiment. When perturbations to the cue signal and the reservoir states were switched off (not shown), the final NRMSE improved to 0.0037, almost as good as in the alloconceptor comparison. In summary, with cue and state noise the IP5 experiment worked out satisfactorily, and in a noise-free version essentially as well as with alloconceptors.
- In the FP experiment, unlike in IP5, state noise not only did not harm, but was conducive for fast adaptation. As a comparison I repeated the simulation with noiseless cues and without state noise insertion during autoconception (not shown). The outcome: in some cases the auto-adaptation failed altogether, and when it did function, it needed much longer runtimes (1,000,000 steps) to reach reconstruction qualities similar to what is reported in Figure 9 **C** **D**. State noise in auto-adaptation helps to drive the singular values of relevant state components toward 1 and considerably speeds up the adaptation.

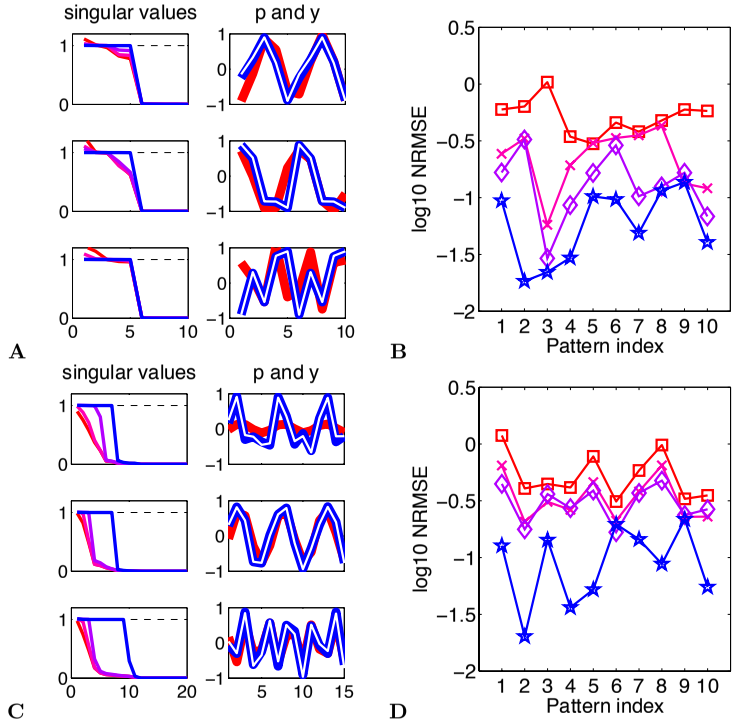


Figure 9: Basic content-addressable memory demos. **A**, **B**: 5-periodic pattern, **C**, **D**: mix of sines pattern. Panels **A**, **C** show the first three patterns, giving the first few singular values of the conceptors C^{cue}, $C(t_1)$, $C(t_2)$, $C(t_3)$. The "p and y" panels show the original patterns (white), the pattern re-generated with C^{cue} (thick red) and with the final $C(t_3)$ (thick blue) after optimal phase alignment with the original. **B**, **D**: recall log10 NRMSEs obtained from $C^{\text{cue}}, C(t_1), C(t_2), C(t_3)$.

- The robustness of auto-adaptation against very strong state noise may seem surprising at first, but is easily explained. Autoconceptor adaptation leads to singular value spectra with many zero values (Jaeger, 2014). Reservoir state noise components in directions of the nulled eigenvectors are entirely suppressed in the conceptor-reservoir loop, and state noise components within the nonzero conceptor eigenspace do not impede the development of a "clean" rectangular profile but in fact, as seen in FP, may even speed up the auto-adaptation process.

- In the simulations reported above, zero singular values were present from the start because the conceptor was initialized as the zero matrix. If it had been initialized differently (for instance, as the identity matrix), the auto-adaptation would only asymptotically pull (the majority of) singular values to zero, with noise robustness only gradually emerging to the degree that many singular values decrease toward zero. If noise robustness is desired, it can be reached by additional adaptation mechanisms for C. In particular, it is helpful to include thresholding: all singular values of $C(n)$ exceeding a suitable threshold are set to 1, all singular values dropping below a certain cutoff are zeroed (not shown).

- The simulations reported here were done with unfavorable settings on purpose, using short cueing times and strong noise. With larger reservoir networks, longer cueing times, less noise etc. much higher recall accuracies can be obtained (not shown).

An inquisitive reader may ask why I don't install auto-adaptation with respect to the reservoir output $\mathbf{r}$ instead of $\mathbf{z}$, that is, why not use the loss

$$L(C) = E[\|C\,\mathbf{r}(n) - \mathbf{r}(n)\|^2] + \alpha^{-2}\,\|C\|^2_{\mathrm{fro}}, \tag{19}$$

instead of the loss (15). This alternative loss (19) appears as natural as the loss (15) (and its solutions would be much easier to analyze mathematically). But in simulation experiments (not reported) I found that the performance of auto-adaptation based on (19) was far inferior to the version used above. Specifically, (i) all noise robustness was lost, (ii) during auto-adaptation the conceptors grew very heavy tails in their singular value spectra even for 5-periodic patterns where one would desire a spectrum with exactly 5 nonzero singular values, and (iii) singular values greater than 1 emerged under auto-adaptation, likewise undesirable within the conceptor philosophy.

The conceptor auto-adaptation dynamics in content-addressable memories is quite involved, with intriguing analogies and differences to Hopfield network adaptation dynamics. I discuss this in Section 3.3.

3.2 From Rote Learning to "Understanding"

When the cue represents a pattern that was not loaded, autoconceptors can be invoked to re-generate even such patterns. Specifically, if the reservoir was loaded with a few patterns $p^1, \ldots, p^k$ from a parametric family and then is cued with another pattern p^* from that family, an attempt can be made to re-generate this new pattern, too. This will work out provided that (i) enough patterns $p^1, \ldots, p^k$ had been loaded to instruct the reservoir about the characteristics of the pattern family, and (ii) the reservoir size is large enough to code the characteristics of the pattern family. For a demonstration I chose a 100-neuron reservoir and the mix-of-sines patterns from the 2-parametric family used before. A cue time of 12 steps and an auto-adaption time of 10,000 was used. The simulation was carried out as follows (detail in the appendix):

1. Create a random reservoir.

2. In separate trials, load this reservoir with an increasing number k of patterns (ranging from $k = 1$ to $k = 1000$), randomly taken from the parametric family.

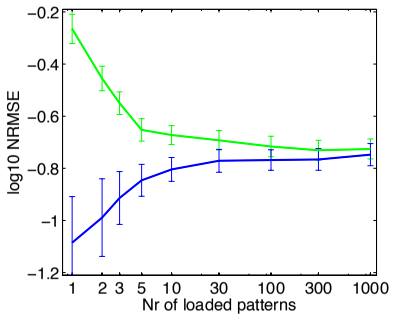


Figure 10: Class learning effect. Error bars indicate 95 % confidence intervals for mean NRMSEs. The curves are means over the 10 recall targets and the 10 experiment repetitions (blue: recall of patterns that were loaded; green: recall of patterns that were not loaded). Both axes are in logarithmic scale. For explanation see text.

3. After loading, cue (with noiseless cues) and re-generate the loaded patterns by autoconceptors. Measure the final recall accuracy for the first 10 of the loaded patterns (when fewer than 10 were loaded, do it only for these).

4. In addition, per trial, also try to cue and "recall" 10 novel patterns that were drawn randomly from the family. Monitor the "recall" accuracy of these novel patterns as well.

5. Repeat this entire scheme 10 times, with freshly sampled reservoirs and patterns. Report averaged diagnostics.

Figure 10 shows the results. Observations:

- The recall NRMSE for patterns that have been expressedly loaded is lowest when only few patterns have been loaded. As the number of loaded patterns grows, the NRMSE first rises, then levels out for very large numbers of loaded patterns.

- The recall NRMSE for non-loaded patterns is high in conditions when only few patterns have been loaded. As the number of loaded patterns grows, this NRMSE decreases and levels out at about the same goodness as the NRMSE for loaded patterns.

Similar findings were obtained in other simulations (Jaeger, 2014). Pending a formal analysis, the following interpretation suggests itself:

- For small numbers of loaded patterns the system stores and recalls individual patterns. The input simulation matrix D represents **individual** patterns.

- For large numbers of loaded patterns, the system learns a representation of the parametrized pattern family and can re-generate any pattern from that family from a cue. The input simulation matrix D represents the **class** of patterns.

In suggestive wording, as more and more patterns are loaded this memory system changes its nature from "rote learning" to "understanding" the nature of the pattern family. With the aid of autoconceptors, any member of the family can be "recognized" from a cue and become re-generated. This family learning effect becomes effective already when only a single pattern has been loaded, and when as few as about 5 patterns are loaded, the recall log10 NRMSE for novel patterns is within 20% of the accuracy for loaded patterns. This is another display of the generalization powers of conceptor-based systems, which we encountered before in connection with conceptor morphing (Section 2.2).

3.3 Analysis of Auto-Adaptation Dynamics

The dynamics of autoadaptation—that is, the evolution of $C(n)$ under the simultaneous reservoir state update (13), (14) (or variants) and conceptor matrix adaptation rule (18)—is highly nonlinear and involves at least the two timescales of reservoir updates (fast, N variables) and C adaptation (slow, N^2 variables). A full analysis is out of reach, but some core aspects are analytically accessible. In Jaeger (2014) I investigate properties of fixed points of C auto-adaptation. Here I review the most important findings and attempt an interpretation of some phenomena found in simulation studies in the light of these analytical insights. This subsection assumes some familiarity with methods from dynamical systems theory. Readers not interested in technical detail may jump to the intuitive summary at the end of this subsection.

My analysis was based on assuming an adaptation rate λ that is small enough to admit temporal averaging over the fast $\mathbf{z}(n)$ state variables. The discrete update equation (18) can then be transformed into a continuous-time ODE which is easier to analyze. This ODE is

$$\tau \dot{C} = (I - C)\, C\, Q_C\, C' - \alpha^{-2}\, C, \tag{20}$$

where $Q_C = E_{\mathbf{z}}[\tanh(W\,\mathbf{z} + \mathbf{b})\tanh(W\,\mathbf{z} + \mathbf{b})'] = E_{\mathbf{r}}[\mathbf{r}\,\mathbf{r}']$ is the autocorrelation matrix of reservoir states (here in a version with input internalization weights, versions with input simulation weights are analog) when these states evolve under the control of C. The time constant τ is of no concern for the qualitative analysis. Specifically I investigated two objects:

Fixed points F of (20). These are candidates for conceptors that are asymptotically obtained for convergent autoadaptation runs.

Jacobians J_F of fixed points F. J_F is an $N^2 \times N^2$ matrix whose (i,j)-th element is the derivative $\partial \dot{F}_i / \partial F_j$, under the dynamics (20) linearized in F, and where F_i is the i-th element of F (enumerated row-wise). The eigenvalues of J_F characterize stability properties of F.

Here is a summary of insights from this analysis:

- Fixed points F are positive-semidefinite matrices with eigenvalues (= singular values) ranging in $[0, 1]$, and thus qualify as conceptors.

- When F has singular values that are smaller than $1/2$ and greater than 0, J_F has positive eigenvalues and hence F is not a stable fixed point. Such F cannot be obtained in empirical auto-adaptation runs. Conversely, this means that if auto-adaptation converges, the resulting conceptors C have singular values that are either exactly zero or larger than $1/2$.

- The rank of F is bounded from above by the number of singular values of Q_F which are greater than $4\alpha^{-2}$. This implies that reducing the aperture α forces an increasing number of singular values of F to become zero.

- Assume F has rank k. Then J_F has $k(N-k)$ zero eigenvalues, that is, a center manifold $\mathcal{K}$ of dimension $k(N-k)$. On this center manifold the stability properties of the dynamics (20) cannot be elucidated by linearization techniques; it may be stable, instable, or neutral.

- If F is a rank-k solution of the fixed-point equation (17) which is generic in the sense that the nonzero singular values of F are pairwise different, and if Q in (17) is fixed (that is, its dependency on C is ignored), then differential changes of F along a $k(N-k)$-dimensional subspace will also lead to rank-k solutions of (17) (Jaeger (2014), Section 3.13.4). The identity of the center manifold dimension and the dimension of this local solution space implies that the center manifold coincides with the local solution space. Hence the adaptation dynamics on the center manifold $\mathcal{K}$ is neutrally stable: $\mathcal{K}$ is a plane attractor. Note however that this interpretation is based on fixing Q at Q_F in the vicinity of F. Plane attractors are non-generic in randomly parametrized ODEs, and fixed point solutions F are generically isolated (not embedded in a plane attractor). When the dependence of Q on C is included in the picture, and F is a fixed-point solution of (17), we would generically expect a separation of timescales in the autoadaptation dynamics in the vicinity of F: fast convergence toward F in directions orthogonal to $\mathcal{K}$ and (very) slow drift within $\mathcal{K}$. Below I present a numerical simulation that corroborates this intuition.

- In order to obtain a better understanding of convergence properties in directions orthogonal to $\mathcal{K}$ one needs to analyse the negative eigenvalues of the Jacobian J_F. Assume again that F has rank k, and furthermore the k nonzero singular values $s_1, \ldots, s_k$ of F are greater than $1/2$. Then the $N^2 - k(N-k)$ nonzero eigenvalues of J_F are all negative. They come in three sorts (Proposition 16 in Jaeger (2014)):

  1. $N(N-k)$ eigenvalues are equal to $-\alpha^{-2}$ ("sort 1"),
  2. k eigenvalues are associated with the k singular values s_i and have the form $-\alpha^{-2}\,(2s_i - 1)/(1 - s_i)$ ("sort 2"),
  3. the remaining $k(k-1)$ eigenvalues ("sort 3") come in pairs

$$\lambda_{1,2} = \frac{-\alpha^{-2}}{2}\left(\frac{s_l}{1-s_m} + \frac{s_m}{1-s_l} \pm \sqrt{\left(\frac{s_l}{1-s_m} - \frac{s_m}{1-s_l}\right)^2 + 4}\right),$$

  where $m < l \leq k$.

When $k \ll N$, which is often the case, the majority of all eigenvalues of F is of sort 1, revealing that in the majority of directions orthogonal to the center manifold the convergence rate toward F is inversely proportional to α^2. A relevant message here is that increasing the aperture substantially slows down convergence. An eigenvalue of sort 2 is also inversely proportional to α^2, but furthermore the term $(2s_i - 1)/(1 - s_i)$ indicates that as $s_i \downarrow 1/2$, the convergence rate in this direction slows down arbitrarily much, while conversely for $s_i \uparrow 1$, convergence becomes arbitrarily fast. Finally, eigenvalues of sort 3 are products of two factors too, the first factor being again inversely proportional to α^2. The second factor, which depends on a pair s_l, s_m, may become arbitrarily large or arbitrarily small depending on s_l, s_m and the $\pm$ choice. Specifically, this second factor approaches zero if both s_l, s_m approach $1/2$ from above and the "$-$" option is taken. In summary, outside the center manifold convergence rates are always co-determined by a factor proportional to α^2, and by factors that depend on the singular values s_i of F, where these latter factors may shrink toward zero if associated singular values s_i approach $1/2$.

- As we have just seen, singular values s_i of F which are close to $1/2$ lead to slow convergence (which implies weak stability) in some directions of C. It is therefore relevant to investigate into typical values of s_i. Unfortunately I can only provide a plausibility argument to the effect that most s_i will be close to 1 and only very few (if any at all) close to $1/2$. This argument goes as follows. Recall from (16) that F, because it is a minimizer of (15), can be written as $F = R(R + \alpha^{-2} I)^{-1}$, where $R = E[\mathbf{z}\,\mathbf{z}']$. This implies that the singular values s_i of F relate to the singular values s_i^R of R by $s_i = s_i^R / (s_i^R + \alpha^{-2})$. Furthermore, nonzero s_i must be greater than $1/2$. Assume the "worst case" that the smallest nonzero singular value of the (rank k) matrix F is $s_k = 1/2$, which translates to $s_k^R = \alpha^{-2}$. Let $s_{k-1}^R = s_k^R + \varepsilon$ be the next greater singular value of R. A simple argument based on the derivative $d\,s_i / d\,s_i^R$ evaluated at s_k^R yields that for small increments ε we get $s_{k-1} \approx 1/2 + 1/4\,\alpha^2\,\varepsilon$. In words, the singular values s_i of F grow away from $1/2$ by increments $1/4\,\alpha^2\,\varepsilon$. Since apertures are often in the order of 10 (or much larger), this means that we may expect that most if not all nonzero singular values of F will be close to 1. This is in fact what I consistently observe in simulations, like in the demos above (Figure 9). It is a desirable effect because according to the considerations made in this subsection before, singular values $s_i \approx 1$ or $s_i = 0$ foster noise robustness of auto-adaptation.

- The analyses available so far can explain the robustness of conceptor auto-adaptation to state noise. When at some stage of auto-adaptation the matrix $C(n)$ has k nonzero singular values and $N - k$ singular values equal to zero, the latter will be preserved under auto-adaptation because the states $\mathbf{z}(n) = C(n)\,\mathbf{r}(n)$ that enter the adaptation of $C(n)$ will have zero components in the $N - k$ directions associated with the zero singular values. The effect of state noise on the nonzero $s_i(n)$ is to drive them closer to 1 than they would go without state noise. Conceptors eventually converged to with state noise are not identical, but similar to variants that would be obtained without noise: the difference being that the former are (even) more rectangular than the latter. This favourable situation however hinges on the condition that the post-cue C^{cue} has (many) zero singular values. In the simulations in previous subsections

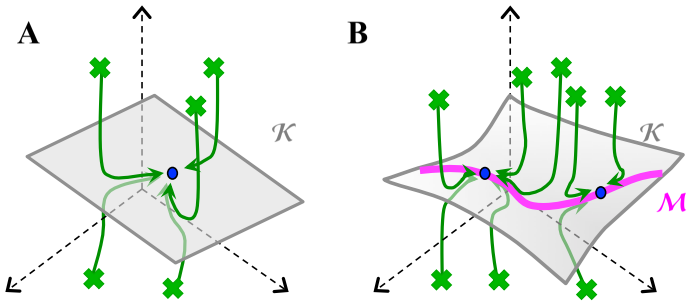


Figure 11: Hypothetical phase portraits of C autoadaptation in the parameter space of C (schematic). Blue points show stable fixed point solutions C. The gray plane in **A** represents the center manifold $\mathcal{K}$ and in **B** the merged center manifolds of neighboring fixed points C. Green arrows show sample trajectories of late stages of C adaptation. Green crosses mark the starting points of the adaptation trajectories set by the cueing procedure. **A**: When a small number of patterns has been loaded, individual stable fixed point conceptors C are created. **B**: In the case of learning a d-parametric pattern class, fixed point solutions C_i become located within a d-dimensional pattern manifold $\mathcal{M}$ (magenta line). For explanation see text.

this was achieved by (i) initializing $C(0) = \mathbf{0}$ and (ii) not inserting state noise during the cueing—state noise insertion in that sensitive period would have led to a large number of nonzero singular values in C^{cue}. If state noise or other perturbations are unavoidable during cueing, it will be a practical idea to simply zero small singular values of the developing $C(n)$ through a thresholding mechanism.

Based on these insights I tentatively offer the following qualitative account of conceptor auto-adaptation dynamics.

This dynamics evolves in the N^2-dimensional parameter space of conceptor (or conceptor candidate) matrices $C(n)$ under the assumption of a small enough adaptation rate λ, such that we may use the ODE (20). I distinguish two cases depending on whether few or many patterns have been loaded. Figure 11 gives a graphical impression.

Representing individual patterns. This corresponds to the situation called "rote learning" in Section 3.2. Here I assume that a small number of patterns p^j have been loaded, and that they become represented by corresponding isolated attracting fixed points F^j of (20). The dynamics in a neighborhood of F^j then should be characterized by a (very) slow timescale of attraction in the directions of the center manifold $\mathcal{K}$ and faster attraction in orthogonal directions (Figure 11 **A**). The attraction orthogonal to

$\mathcal{K}$ may in turn be governed by a wide range of different timescales due to what we have learnt about the dependency of eigenvalues of J_{F^j} on the singular values of F^j.

Representing a pattern family. When many patterns p^j from a family characterized by d parameters have been loaded, we have seen in Section 3.2 that any pattern from that family will be re-generated upon cue equally well. I hypothesize that during the loading procedure a d-dimensional manifold $\mathcal{M}$ is created which represents the pattern family and which superficially appears like a line/plane attractor: when by virtue of the cueing procedure the auto-adaptation is initialized into the vicinity of $\mathcal{M}$, one will observe an auto-adaptation toward $\mathcal{M}$ (Figure 11 **B**). The dimension d of $\mathcal{M}$ will typically be much smaller than the dimension $k(N-k)$ of the center manifold of conceptors in $\mathcal{M}$. I hypothesize that $\mathcal{M}$ is embedded as a submanifold in a $k(N-k)$-dimensional manifold $\mathcal{K}$ which can be understood as a merge of the (locally defined) center manifolds of conceptors sitting in $\mathcal{M}$. The convergence toward $\mathcal{M}$ leads to a good reconstruction of the cue pattern after a "reasonable" adaptation time. It is this convergence which was exploited for 10,000 step runtimes in the experiment shown in Figure 10. However, as remarked earlier, line/plane attractors are non-generic objects and unlikely to exist in empirical or randomly created systems. Therefore, for very long runtimes I would predict a very slow drift within $\mathcal{M}$ that lets auto-adaptation ultimately end up in some isolated attracting fixed point within $\mathcal{M}$. According to this view, line/plane attractors only appear phenomenally on some intermediate timescale, and hence the class learning effect described in Section 3.2 should degrade for very long adaptation times.

In order to substantiate this interpretation of an only "ephemeral" appearance of a line/plane attractor, I carried out a long-duration auto-adaptation simulation. Ten 5-periodic patterns were loaded into a small ($N = 50$) reservoir. These patterns represented ten stages of a linear morph between two similar patterns p^1 and p^{10}, resulting in a morph sequence $p^1, p^2, \ldots, p^{10}$ where $p^i = (1-(i-1)/9)\,p^1 + ((i-1)/9)\,p^{10}$, thus representing instances from a 1-parametric family. Considering what was found in Section 3.2, loading these ten patterns should enable the system to re-generate by auto-adaptation any linear morph p_{test} between p^1 and p^{10} after being cued with p_{test}.

After loading, the system was cued for 20 steps with 20 different cues. In each of these $j = 1, \ldots, 20$ conditions, the cueing pattern p^j_{test} was the j-th linear interpolation between the loaded p^1 and p^{10}. At the end of the cueing, the system will be securely driven into a state $\mathbf{z}$ that is very accurately connected to re-generating the pattern p^j_{test}, and the conceptor matrix that has developed by the end of the cueing would enable the system to re-generate a close simile of p^j_{test} (a post-cue log10 NRMSE of about -2.7 was obtained in this simulation).

After cueing, the system was left running in conceptor auto-adaptation mode for 1 Mio timesteps, with an adaptation rate of $\lambda = 0.01$. At times $n = 1, 1000, 10000, 1e^6$ the stage of convergence was assessed as follows. The pairwise distances between the twenty autoconceptors $C^j(n)$ were compared, resulting in a 20×20 distance matrix $D(n) = (\|C^k(n) - C^l(n)\|_{\text{fro}})_{k,l=1,\ldots,20}$. Figure 12 shows color plots of these distance matrices. The outcome: at the beginning of autoadaptation ($n = 1$), the 20 autoconceptors are spaced from each other by distances proportional to their morphing distances. In terms of the

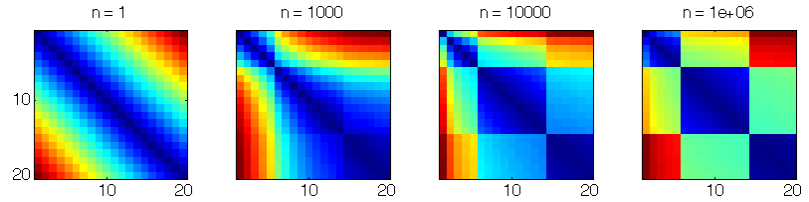


Figure 12: Numerical exploration of fixed point solutions under C auto-adaptation. Each panel shows pairwise distances of 20 conceptor matrices obtained after n auto-adaptation steps, after being cued along a 20-step morph sequence of cue signals. Color coding: blue: zero distance; red: maximum distance. For explanation see text.

schematic in Figure 11 **B**, they would all be almost equi-distantly lined up on the manifold $\mathcal{M}$, which is a line in this case of a 1-parametric pattern family. Then, as the adaptation time n grows, they contract toward three point attractors within $\mathcal{M}$ (which would correspond to a version of 11 **B** with three isolated fixed point attractors within $\mathcal{M}$). These three point attractors correspond to the three dark blue squares on the diagonal of the last distance matrix shown in Figure 12.

This singular simulation cannot, of course, provide conclusive evidence that the qualitative picture proposed in Figure 11 is correct. A rigorous mathematical characterization of the hypothetical manifold $\mathcal{M}$ and its relation to the center manifolds of fixed point solutions of the adaptation dynamics remains to be worked out.

Plane attractors have been proposed as models for a number of biological neural adaptation processes (summarized in Eliasmith (2005)). Specifically, the fact that animals can fix their gaze in arbitrary (continuously many) directions has been modelled by plane attractors in the oculomotoric neural control system. Each gaze direction corresponds to a (controlled) constant neural activation profile. It would be interesting to carry out in-vivo experiments of gaze direction fixation where the animal (or human) is cued with a direction target, which is then removed, under the instruction to preserve the gaze direction after target removal for an extended timespan. The mathematical model outlined above would predict that the gaze direction would start drifting, possibly becoming ultimately arrested in one of a number of isolated, "preferred" gaze directions.

3.4 Single-Step Content-Addressing by Thresholding

We observed above that approximately rectangular conceptors evolved in auto-adaptation and that directions in $\mathbf{z}$ states which are nulled by C^{cue} remain nulled during auto-adaptation. This suggests a drastic shortcut of the recall procedure: take the preliminary C^{cue} which is available immediately after the cue period, then transform its singular value profile to a binary 0-1-rectangular shape by thresholding to give $\overline{C^{\text{cue}}}$, and use this conceptor for pattern re-generation. Accordingly, I re-ran the 5-periodic and mix-of-sines demos from

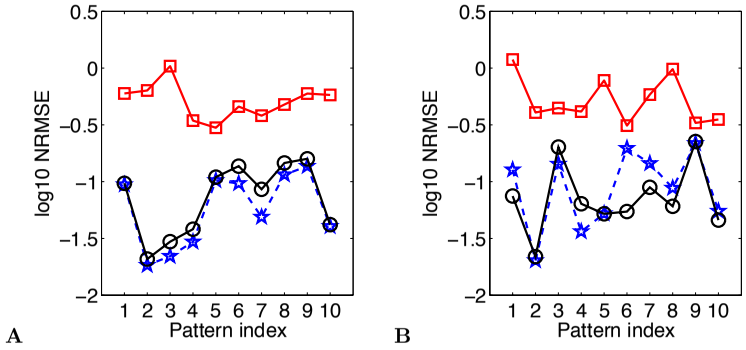


Figure 13: Using single-step adapted conceptors $\overline{C^{\text{cue}}}$ for cued pattern recall. **A**: 5-periodic patterns, **B**: mix-of-sines. Red and blue plots are identical to Figure 9 and represent post-cue NRMSE from C^{cue} and $C(t_3)$. The black plot gives the NRMSE obtained from $\overline{C^{\text{cue}}}$.

Section 3.1.2 with the same reservoirs and cueing conditions, and directly at the end of the cueing phase I computed $\overline{C^{\text{cue}}}$ from C^{cue} as follows:

1. compute the SVD $USU' = C^{\text{cue}}$,
2. recompute the entries s_i on the diagonal of S to new, binary values by thresholding, obtaining $\overline{S}$ with diagonal values

$$\overline{s_i} = \begin{cases} 1, & \text{if } s_i > \tau \\ 0, & \text{else} \end{cases}$$

   for some threshold τ,
3. recombine into $\overline{C^{\text{cue}}} = U\overline{S}U'$,
4. immediately use $\overline{C^{\text{cue}}}$ for re-generating a pattern and report the matching NRMSE with the original pattern that was used for the cue.

Figure 13 displays the outcome. I used thresholds τ of 0.5 and 0.01 respectively for the 5-periodic and mix-of-sines experiments. For both kinds of patterns the re-generation quality obtained from $\overline{C^{\text{cue}}}$ conceptors is essentially the same as the quality obtained from the auto-adapted $C(t_3)$ after 500 resp. 10,000 steps for the two kinds of patterns.

Considering that thresholded $\overline{C^{\text{cue}}}$ are much faster and cheaper to create than online-adapted conceptors, yet (in these simulations) are of the same quality, it is natural to ask why one should bother about online-adapted autoconceptors at all instead of focussing all attention on thresholded conceptors. The reasons why I spent much effort on the former (and used much of the reader's patience) are manifold:

- There are other applications of online-adapted autoconceptors besides pattern recall from a memory. In Jaeger (2014) I employ conceptor auto-adaptation in hierarchical RNN demonstrator systems for tasks of signal denoising, pattern classification, and pattern mixture recognition, and in ongoing work I am using them for single-shot training of a nonlinear channel equalizer and blind signal separation. In all of these cases, an adaptive RNN-based system has to be able to smoothly adjust its reactions to a non-stationary input stream, requiring a continuously ongoing gradual adaptation.

- A serious drawback of the above auto-adaptation mechanism is that it is based on stochastic gradient descent in a high-dimensional space, incurring the well-known stability issues which mandate small adaptation rates, hence yield slow convergence. In Jaeger (2014) I introduce a simplified version of conceptors which basically constrains conceptor matrices C to diagonal form, leading to decoupled adaptation rules for the diagonal parameters which admits individual and very large adaptation rates. While systems of this kind work well in some specific scenarios (demos in Jaeger (2014), Section 3.15), I am not satisfied with them in general, mainly because the value range for well-working apertures becomes very narrow. My current research concentrates on new types of conceptors which combine fast online adaptivity with the robustness properties found in matrix conceptors.

- In computational neuroscience, attractor-like phenomena of all sorts are widely investigated as neural mechanisms for representing and processing information in neural systems (partial overviews in Durstewitz et al. (2000); Fusi and Wang (2016); Jaeger (2012)). The multiple-timescale attractor phenomenology of conceptor auto-adaptation discussed in the previous section sheds some further light on this field, especially with respect to the role of line/plane attractors in the representation and addressing of continuous memory items.

- Biological implausibility. The fast thresholding mechanism invokes a matrix SVD computation. Much research has been devoted to determine biologically plausible neural algorithms for the closely related task of computing a PCA or of finding some cost-optimal projection subspaces (Oja, 1982; Pehlevan et al., 2015). However, these neural algorithms share with my stochastic gradient conceptor adaptation the shortcoming of slow convergence, and biological plausibility remains debatable in my view.

3.5 Comparison with Hopfield Networks

As I mentioned in the Introduction, Hopfield networks and its relatives (Willshaw et al., 1969; Cooper, 1973; Kohonen, 1974; Palm, 1980; Hopfield, 1982) are the paradigmatic model of content-addressable neural long-term memories. For brevity I will use the term *Hopfield networks* (HNs) as an umbrella term for this family of models, and refer to the networks introduced in this section as autoconceptor networks (ACNs). ACNs are analogous to HNs in some ways and different in others:

Nature of memory items. Patterns stored in (auto-associative) HNs are static (often images), though hetero-associative HN variants can be trained to re-generate sequences of static patterns—I briefly discussed this in the Introduction. ACNs intrinsically

host temporal patterns. The pattern sequences of static patterns learnt by sequence-representing HNs differ in kind from the temporal patterns in ACNs. The former are "jump" transitions between a finite number of explicitly trained "waypoint" patterns, and in principle any transition order can be learnt. Temporal patterns in ACNs have what one might call intrinsic temporality; the reservoir state sequence cannot in general be re-trained in other orders. Specifically, ACNs can host non-periodic patterns sampled from ODE evolutions which is not possible with hetero-associative HNs. One might word this as, "hetero-associative HNs can store pattern sequences, and ACNs can store sequence patterns".

States and patterns. In HNs patterns are identified with certain network states. In ACNs patterns are state evolution processes.

Symmetric vs. directed synaptic connections. Standard auto-associative HNs have symmetric connections, which admits an analysis of HN dynamics in terms of descent in an energy landscape. Reservoir networks in ACNs with their asymmetric weight matrices do not admit an energy-based interpretation.

Pattern restauration. A hallmark of HNs is their ability to restore patterns perfectly and quickly from highly corrupted cues. This striking capability results from the fact that HN training creates well-defined local minima in the HN energy landscape which correspond 1-1 to target patterns; from any arbitrary cue the HN *must* settle in one of these perfect target patterns (setting aside the issue of spurious attractors in HNs). Furthermore, perfection of restauration is aided by the circumstance that HNs often are installed with binary neurons, representing binary patterns. In contrast, the continuous-valued states in ACNs and the vastly more complex multi-timescale dynamics of auto-adaptation limit the perfection and speed of pattern re-generation.

Class learning. ACNs can learn entire continuous-parametric pattern classes (with the caveats pointed out in Section 3.3), which HNs cannot.

All in all, while both HNs and ACNs are models of content-addressable neural long-term memories, their underlying neuro-dynamical working mechanisms and their performance characteristics are fundamentally—and interestingly—different.

4. Discussion

This article is the first peer-reviewed publication about conceptors and thereby assumes the role of an "official" introduction of this concept and name. New scientific concepts and terminology should be introduced with care and circumspection. So, what is a "conceptor"?

This article dealt with conceptors that come in the specific form of positive-semidefinite matrices derived from reservoir state correlation matrices R via $C = R\,(R + \alpha^{-2} I)^{-1}$. In Jaeger (2014) I also describe conceptors that are realized by a forward- and backprojection to/from a higher-dimensional random feature space. In my initial explorations (unpublished) I also used affine maps, as well as pure linear subspace projectors. Currently I am investigating a kind of conceptors built on the basis of a variant of self-organizing maps. In

the light of this diversity I wouldn't want to make any specific mathematical or algorithmical format a defining part of "conceptors".

This article focussed on neural long-term memory for temporal patterns. In Jaeger (2014) however I use conceptors also for static patterns (where an input pattern is transformed to a single neural state, not a sequence of states), and without loading (which is not necessary when conceptors are employed for non-generative tasks like pattern classification). I am also aware of ongoing work by others in static pattern recognition where backprop-trained deep feedforward neural networks are combined with conceptors. Thus, neither temporality of patterns, nor loading them persistently, nor the use of random networks are necessary attributes of "conceptors", the way I want them to be seen.

Then what is left? Here is an outline of how I would like the concept of conceptors to be understood:

1. *Set-up: patterns encoded by states.* Given: some computational framework where a diversity of **patterns** p^j are **encoded** by probability distributions P^j over a metric state space X. *Comment: patterns can be temporal or static. The state space is typically high-dimensional, and encodings would typically be "distributed".*

2. *Filtering of encodings.* A **conceptor** C^j is a **map** associated with a pattern p^j which transforms state vectors $x \in X$ to state vectors $C^j(x) \in X$, and which is optimized in some way to preserve states x^j that are typical for pattern p^j: if $P^j(x)$ is large, then the distance $d(x, C^j(x))$ is small. Furthermore, C^j projects off-pattern states closer to pattern-typical regions of X: if $P^j(x)$ is small, then $P^j(C^j(x)) > P^j(x)$. *Comment: The map C^j can be instantiated by a mathematical formula, or by some neural circuit, or in any other way.*

3. *Conceptual operations.* Given a collection $\{p^j\}$ of patterns, on the associated collection of conceptors $\{C^j\}$ one has available an assortment of operations which can **construct** new conceptors from the conceptors contained in the original collection $\{C^j\}$. These operations should be interpretable as "cognitive-level" operations on concepts, and it is these operations that connect conceptors with concepts. *Comment: this is obviously vague. This article featured the following such operations: (i) conceptual blending by conceptor morphing, (ii) "focussing" by aperture adaptation. A large part of Jaeger (2014) is devoted to a third kind of conceptual operations on conceptors, namely, (iii) Boolean combinations of conceptors.*

My ultimate motivation to investigate conceptors is to establish an effective link between "low-level", "subsymbolic", "distributed" neural representations and processing mechanisms on the one hand, and a "high-level", "conceptual" organization and interpretability of such processing on the other hand. This intended use of conceptors, which is expressed in item 3. in the list above, is to some degree independent of the "mechanics" of conceptors, which is stated in item 2. It turns out that conceptors also can serve relevant "low-level" functionalities, especially neural noise suppression and stabilization of neural state dynamics.

Conceptors can be engineered into artificial neural architectures, and they might be instantiated in one way or the other by neural circuitry in biological systems. In such

cases they become procedurally effective in the concerned systems, implementing *conceptor mechanisms*. This was the perspective adopted in this article. But conceptors can also be used "epistemologically" by a researcher who investigates some (likely neural) information processing system which may or may not itself have have effective conceptor mechanisms inside. Then conceptors may become elements of a descriptive scientific language that talks *about* neural encodings of concepts. In Jaeger (2014) I describe in detail such scientific languages in the format of formal *conceptor logics*.

Acknowledgments

I am indebted to Emre Neftci for bringing the work of Sompolinsky and Kanter (1986) to my attention. The first ideas of conceptors were triggered by research within the European FP7 project AMARSi (contract 248311, `amarsi-project.eu`) on the modulation of neural pattern generators for humanoid robot motor control.

Appendix A. Documentation of Simulation Detail

Training and test errors are given as normalized root mean square error (NRMSE) between a signal $s(n)$ and its target $t(n)$, that is, by the square root of the mean square error divided by the variance of the target.

A.1 Detail for Section 2.1

Pattern definition: Pattern p^1 is a sinewave sampled at intervals of about 1.9944. Pattern p^2 alternates between -0.5 and 0.5.

Network set-up: Initial weights W^* were sampled from the normal distribution, then the weight matrix was rescaled to a spectral radius (largest absolute eigenvalue) of 1.3. Input and bias weights were sampled from the normal distribution, then scaled by 1.4 and 0.1 respectively.

Learning: For training W^{out} the reservoir was driven by iid input sampled from the normal distribution scaled by 1.5 for 200 steps (plus an initial washout of 100 steps). The linear regression was computed without regularization, resulting in a training NRMSE of 0.39 (training W^{out} from state patterns $\mathbf{x}^1(n), \mathbf{x}^2(n)$ would have given an NRMSE of 0.14). For the (not regularized) linear regression leading to W, again 200 states from each of the two patterns were used, leading to an NRMSE of 0.093 (average over the three neurons).

Recall: In order to numerically compare the re-generated patterns $y(n) = W^{\text{out}}\,\mathbf{x}^j(n)$ to the original patterns $p^j(n)$, sample trajectories of both were phase-aligned by (i) super-sampling with cubic spline interpolation by a factor of 20, (ii) finding the best-fitting phase shift, (iii) subsampling back to the original sampling rate, (iv) computing the NRMSE, which was 0.15 for the first and 0.13 for the second pattern.

Note: Parameters and the random seed for network set-up were hand-selected for good visual appearance of this demo example. Better recall accuracies were obtained with smaller apertures, but these would have given less instructive graphics. At any rate, a network size of $N = 3$ is far too small for high-precision results.

A.2 Detail for Section 2.2

Interpolation from 5-periodic to irrational sine (Fig. 3): A 100-unit reservoir was used, W^* scaled to a spectral radius of 1.6, input weights scaled to 1.6, bias vector to 0.3. Linear regression for W was regularized by a Tychonov regularizer of size $\varrho^2 = 0.0001$. Apertures for the two conceptors: 10 and 100.

Morphing between and beyond three patterns from a family (Fig. 4): Patterns were taken from the parametrized family with parameters A, B

$$p^{A,B}(n) = 2\left(\frac{1}{2}\sin(2\pi n/(P+A)) + \frac{1}{2}\right)^{\exp(B)} + 1, \quad A, B \in \mathbb{R}; |A| < P.$$

This family contains modulated versions of sines whose period lengths vary around a reference period P by differences A and whose shape is modulated by B. Figure 4 (top panels) shows the three stored patterns, which were defined by $P \approx 8.2$, $A = -1.1\sqrt{3}; 1.1\sqrt{3}/2; 0$, $B = -0.2, -0.2, 0.4$. Going upward by one tile changes a mixture from (a^1, a^2, a^3) to $(a^1 - 1/8, a^2 - 1/8, a^3 + 1/4)$, going right by one tile to $(a^1 - 1/4, a^2 + 1/4, a^3)$. The reservoir had 100 units, spectral radius of initial weight matrix: 2.2, input weight scaling: 2.2, bias scaling; 1.5, aperture of all conceptors 2.0. Ridge regression coefficient for loading: $\varrho = 0.001$.

A.3 Detail for Section 2.3

Preparation of training data. Fifteen human motion capture (mocap) sequences were retrieved from the public-domain mocap repository maintained by the Graphics Lab at Carnegie Mellon University (`http://mocap.cs.cmu.edu/`). Using the public mocap toolsuite provided by Burger and Toiviainen (2013), the marker trace format of the original mocap data was transformed into joint angle data in the following way. A human segment model with 17 segments was chosen. The original marker trace data was transformed into metric 3D trajectories of the segment endpoints (joints). The joint between the two hip joints was designated as "root". Let x, y denote the two ground coordinates and z the height coordinate (measured as distance from ground), and let δ (measured in rad) denote the horizontal pointing direction (in x, y) of the root.

The motion of the root joint was coded in the following four variables:

1. $h(n)$: simply the z coordinate of the Euclidean trajectory of the root,
2. $\dot{\delta}(n)$: the angular velocity (in rad/frame) of the root pointing direction,
3. $d_1(n), d_2(n)$: travel distance of root joint in the x, y plane of the current frame, relative to the horizontal pointing direction (Fig. 14), measured in mm/frame.

The locations of the other joints were coded relative to their predecessor joints in the kinematic chain tree, with the root joint as the top node, as follows:

1. The current frame n is horizontally shifted and rotated such that the root joint has (x, y)-position $(0, 0)$ and is pointing in the direction of the x-axis. For every joint j this results in three Euclidean coordinates $x_j(n), y_j(n), z_j(n)$.

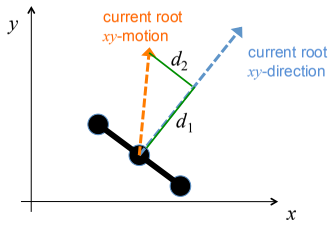


Figure 14: How the quantities d_1, d_2 geometrically relate to current root direction and root travel.

2. For each of the 19 non-root joints j, the current orientation was coded in three variables $s_j^x(n), s_j^y(n), s_j^z(n)$ as follows:

   (a) Determine the predecessor joint i of j (the adjacent joint that is closer to the root).

   (b) Compute the segment vector $s_0^j(n) = (x_j(n) - x_i(n), y_j(n) - y_i(n), z_j(n) - z_i(n))'$ and normalize it to $s^j(n) = s_0^j(n)/\|s_0^j(n)\|$. The three components of $s^j(n)$ are returned as coding the current spatial angle of the segment ending in joint j.

All in all this leads to a 61-dimensional coding of a body pose and root motion per frame. From such 61-dimensional sequences, joint trajectories in Euclidean task space can be recovered by a reversal of the coding procedure.

For each of the 15 original mocap timeseries, such a 61-dimensional encoding was produced. Each of the 61 signal components was shifted/scaled such that it ranged exactly in $[-1, 1]$, across all 15 patterns. These 15 range-normalized sequences were then loaded into the reservoir, and the readout weights were computed to recover them.

Reservoir setup. Different from all other simulations reported in the article, the motor pattern demonstration used a reservoir with leaky integration neurons:

$$\mathbf{x}(n+1) = (1-a)\,\mathbf{x}(n) + a\,\tanh(W\,\mathbf{x}(n) + W^{\text{in}}\,p(n+1) + \mathbf{b}),$$

where the leaking rate a is an additional tuning parameter. This alternative choice of network model does not affect any of the conceptor-related computations. Reservoirs based on leaky integration units effectively implement temporal smoothing which is indicated when slowly changing (relative to the network update cycle) patterns have to be processed.

Simulation parameters. Network size $N = 600$; leaking rate $a = 0.6$; spectral radius of W^* was 1; input and bias weights were sampled from the uniform distribution on $[-.8, .8]$. Loading each pattern used a washout of 50 network updates (corresponding to 5/12 seconds of simulated real time). Two of the input channels were found to act essentially only as noise sources (because they were essentially constant and after range normalization small fluctuations in the constant value were magnified); their input weights were nulled. Linear regression without regularization was used for computing W and W^{out}. The aperture was

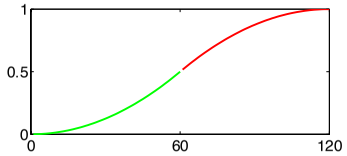


Figure 15: The morphing ramp $\varrho(n)$ used for smooth transitions between motor patterns.

set to $\alpha = 10$ for all fifteen conceptors except for the "slow walk" ($\alpha = 2$) and "boxing: jab" patterns ($\alpha = 20$; with $\alpha = 10$ the jab pattern would be reduced to a mildly active legwork without the jabs being triggered).

Training errors. The training NRMSE for W was 0.0068 and for W^{out} was 0.044.

Video creation. For creating the video, the durations of the individual patterns were set to yield an appealing overall choreography. Between every two successive pattern p^i, p^j evocations, a transition period of 120 network updates (= 1 simulated second) was inserted. In this transition period the preceding conceptor C^i was morphed into the succeeding conceptor C^j, using a morphing function $\varrho : \{1, \ldots, 120\} \to [0, 1]$ made from two suitably scaled and shifted branches of the square function (red and green lines in Fig. 15), via $C^{\text{morph}}(m) = (1 - \varrho(m))\,C^i(m) + \varrho(m)C^j(m) \quad [m = 1, \ldots, 120]$. While the reservoir was generating the behavioral sequence visualized in the video, the only external input to the system was the activation sequence of conceptors (shown in the left upper corner of the video).

The visualization of the re-generated motion sequence in the video keeps the simulated actor centered in the drawing box and illustrates its motion in the x, y plane by a moving floor tile pattern. The centering as well as the corresponding virtual floor motion is computed relative to the mean of the x, y coordinates of all 20 joints. This mean is therefore always centered on $x = y = 0$ in each displayed frame. The apparent foot slipping results from this "quick and dirty" centering method.

A.4 Detail for Section 2.4

Data generation. For the Rössler attractor, training time series were obtained from the standard Rössler ODE $\dot{x} = -(y + z),\ \ \dot{y} = x + a\,y,\ \ \dot{z} = b + x\,z - c\,z$ with $a = b = 0.2, c = 8$. The evolution of this system was Euler approximated with stepsize 1/200 and the resulting discrete time series was then subsampled by 150. The x and y coordinates were assembled in a 2-dimensional driving sequence, where each of the two channels was shifted/scaled to a range of $[0, 1]$. For the Lorenz attractor, the ODE $\dot{x} = \sigma(y - x),\ \ \dot{y} = r\,x - y - x\,z,\ \ \dot{z} = x\,y - b\,z$ with $\sigma = 10, r = 28, b = 8/3$ was Euler-approximated with stepsize 1/200 and subsequent subsampling by 15. The x and z coordinates were collected in a 2-dimensional driving sequence, again each channel normalized to a range of $[0, 1]$. The Mackey-Glass timeseries was obtained from the delay differential equation $\dot{x}(t) = \frac{\beta\,x(t-\tau)}{1+x(t-\tau)^n} - \gamma\,x(t)$ with $\beta = 0.2, n = 10, \tau = 17, \gamma = 0.1$. An Euler approximation with stepsize 1/10 was used. To obtain a 2-dim timeseries that could be fed to the reservoir through the same two input channels as the other attractor data, pairs $x(t), x(t - \tau)$ were combined into 2-dim

vectors. Again, these two signals were normalized to the $[0, 1]$ range. The Hénon attractor is governed by the iterated map $x(n+1) = y(n) + 1 - a\,x(n)$, $y(n+1) = b\,x(n)$, where I used $a = 1.4, b = 0.3$. The two components were filed into a 2-dim timeseries $(x(n), y(n))'$ with no resampling, and again normalization to a range of $[0, 1]$ in each component.

Reservoir setup. A 500-unit reservoir RNN was created with a normal distributed, 10%-sparse weight matrix W^* scaled to a spectral radius of 0.6. The bias vector b and input weights W^{in} (sized 400×2 for two input channels) were sampled from standard normal distribution and then scaled by 0.4 and 1.2, respectively. These scaling parameters were found by a coarse manual optimization of the performance of the pattern loading process. The network size was chosen large enough to warrant a robust trainability of the four chaotic patterns. Repeated executions of the experiment with different randomly initialized weights (not documented) showed no significant differences.

Pattern loading. The length of training signals used in loading was $L = 2500$ with a washout of $n_0 = 500$ for all four patterns. The ridge regression regularizer was $\varrho^2_W = 1e^{-6}$. Output weights were computed from the pattern-driven states with a regularizer $\varrho^2_{\text{out}} = 1e^{-8}$. The average (over neurons and the four patterns) NRMSEs obtained for the reservoir and readout weights were 0.0082 and 0.013, respectively.

A.5 Detail for Section 3.2

Scalings of reservoirs: spectral radius of W^*: 1.1; W^{in}: 1.1; b: 0.25. Aperture: 1000. Regularization coefficients $\varrho^2 = 0.01$ both for W^{out} and D. Data collection runlengths in loading: 500 (plus 100 washout) per pattern. Cueing: washout 100 steps, cue adaptation time 12 steps. No noise added to cue signal. State noise added during auto-adaptation with a signal-to-noise ratio of 1.

References

S.-I. Amari. Learning patterns and pattern sequences by self-organizing nets of threshold elements. *IEEE Trans. on Computers*, C-21(11):1197–1207, 1972.

A. Billard and G. Hayes. DRAMA, a connectionist architecture for control and learning in autonomous robots. *Adaptive Behavior*, 7(1):35–63, 1999.

K. J. Boström, H. Wagner, M. Prieske, and M. de Lussanet. Model for a flexible motor memory based on a self-active recurrent neural network. *Human Movement Science*, 32: 880–898, 2013.

B. Burger and P. Toiviainen. MoCap Toolbox – A Matlab toolbox for computational analysis of movement data. In Roberto Bresin, editor, *10th Sound and Music Computing Conference*, pages 172–178, Stockholm, Sweden, 2013. KTH Royal Institute of Technology. URL `https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/mocaptoolbox`.

CMU Graphics Lab. Motion Capture Database. URL `http://mocap.cs.cmu.edu/`. retrieved Feb 2013.

A. Coates, P. Abbeel, and A. Y. Ng. Learning for control from multiple demonstrations. In *Proc. 25th ICML, Helsinki*, 2008.

L. N. Cooper. A possible organization of animal memory and learning. In B. Lundqvist and S. Lundqvist, editors, *Collective properties of physical systems: Medicine and Natural Sciences*, Nobel Symposia on Medicine and Natural Sciences, pages 252–264. Academic Press, New York and London, 1973.

R. Douglas and T. Sejnowski. Future challenges for the sciene and engineering of learning: Final workshop report. Technical report, National Science Foundation, 2008. URL `http://www.nsf.gov/sbe/SLCWorkshopReportjan08.pdf`.

D. Durstewitz, J. K. Seamans, and T. J. Sejnowski. Neurocomputational models of working memory. *Nature Neuroscience*, 3:1184–91, 2000.

C. Eliasmith. A unified approach to building and controlling spiking attractor networks. *Neural Computation*, 17:1276–1314, 2005.

R. M. French. Catastrophic interference in connectionist networks. In L. Nadel, editor, *Encyclopedia of Cognitive Science*, volume 1, pages 431–435. Nature Publishing Group, 2003.

S. Fusi and X.-J. Wang. Long-term, short-term and working memory. In M. Arbib and J. Bonaiuto, editors, *From Neuron to Cognition via Computational Neuroscience*, chapter 11, pages 319–344. MIT Press, 2016.

M. Gillies and B. Spanlang. Comparing and evaluating real time character engines for virtual environments. *Presence*, 19(2):95–117, 2010.

B. Goodrich and I. Arel. Unsupervised neuron selection for mitigating catastrophic forgetting in neural networks. In *Proc. IEEE 57th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pages 997–1000. IEEE, 2014.

S. Grossberg. Linking attention to learning, expectation, competition, and consciousness. In L. Itti, G. Rees, and J. Tsotsos, editors, *Neurobiology of attention*, chapter 107, pages 652–662. San Diego: Elsevier, 2005.

X. Hinaut and P. F. Dominey. A three-layered model of primate prefrontal cortex encodes identity and abstract categorical structure of behavioral sequences. *J. Physiology - Paris*, 105(1-3):16–24, 2011.

J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. USA*, 79:2554–2558, 1982.

J. Huang and M. Hagiwara. A combined multi-winner multidirectional associative memory. *Neurocomputing*, 48:369–389, 2002.

H. Jaeger. The "echo state" approach to analysing and training recurrent neural networks. GMD Report 148, GMD - German National Research Institute for Computer Science, 2001. URL `http://minds.jacobs-university.de/pubs`.

H. Jaeger. Reservoir self-control for achieving invariance against slow input distortions. technical report 23, Jacobs University Bremen, 2010.

H. Jaeger. Long short-term memory in echo state networks: Details of a simulation study. Technical Report 27, Jacobs University Bremen, 2012. URL `http://minds.jacobs-university.de/pubs`.

H. Jaeger. Controlling recurrent neural networks by conceptors. Technical Report 31, Jacobs University Bremen, 2014. arXiv:1403.3369.

X. Jiang, V. Gripon, C. Berrou, and M. Rabbat. Storing sequences in binary tournament-based neural networks. *IEEE Trans. on Neural Networks and Learning Systems*, 27(5): 913–925, 2016.

M. I. Jordan. Serial order: a parallel distributed processing approach. In J. W. Donahoe and V. Packard Dorsel, editors, *Neural-Networks Models of Cognition*, chapter 25, pages 471–495. Elsevier, 1997. Abridged reprint of a technical report from 1986.

T. Kohonen. An adaptive associative memory principle. *IEEE Transactions on Computers*, 23(4):444–445, 1974.

J. F. Kolen and J. B. Pollack. Multiassociative memory. In *Proc. of the Thirteenth Annual Conference of the Cognitive Science Society*, pages 785–789, 1991.

A. F. Krause, V. Dürr, B. Bl̈(a)sing, and T. Schack. Evolutionary optimization of echo state networks: multiple motor pattern learning. In *ANNIIP - 6th International Workshop on Artificial Neural Networks and Intelligent Information Processing*, 2010.

R. Laje and D. V. Buonomano. Robust timing and motor patterns by taming chaos in recurrent neural networks. *Nature Neuroscience*, 16(7):925–933, 2013.

L. Lukic, J. Santos-Victor, and A. Billard. Learning coupled dynamical systems from human demonstration for robotic eye-arm-hand coordination. In *IEEE-RAS International Conference on Humanoid Robots, Osaka 2012*, 2012.

M. Lukosevicius. A practical guide to applying echo state networks. In K.-R. Müller, G. Montavon, and G. Orr, editors, *Neural Networks Tricks of the Trade, Reloaded*, LNCS, pages 659–686. Springer Verlag, 2012.

G. Manjunath and H. Jaeger. Echo state property linked to an input: Exploring a fundamental characteristic of recurrent neural networks. *Neural Computation*, 25(3):671–696, 2013.

N. M. Mayer and M. Browne. Echo state networks and self-prediction. In *Biologically Inspired Approaches to Advanced Information Technology*, volume 3141 of *LNCS*, pages 40–48. Springer Verlag Berlin / Heidelberg, 2004.

E. Oja. A simplified neuron model as a principal component analyzer. *J. Math. Biol.*, 15: 267–273, 1982.

R. W. Paine and J. Tani. How hierarchical control self-organizes in artificial adaptive systems. *Adaptive Behaviour*, 13(3):211–225, 2005.

G. Palm. On associative memory. *Biol. Cybernetics*, 36(1):19–31, 1980.

C. Pehlevan, T. Hu, and D. B. Chklovskii. A Hebbian/anti-Hebbian neural network for linear subspace learning: A derivation from multidimensional scaling of streaming data. *Neural Computation*, 27(7):1461–1495, 2015.

J. B. Pollack. Recursive distributed representations. *Artificial Intelligence*, 46(1-2):77–105, 1990.

F. R. Reinhart and J. J. Steil. Recurrent neural associative learning of forward and inverse kinematics for movement generation of the redundant pa-10 robot. In A. Stoica, E. Tunsel, T. Huntsberger, T. Arslan, S. Vijayakumar, and A. O. El-Rayis, editors, *LAB-RS 2008, vol. 1*, pages 35–40, 2008.

R. F. Reinhart and J. J. Steil. A constrained regularization approach for input-driven recurrent neural networks. *Differential Equations and Dynamical Systems*, 19(1–2):27–46, 2011. DOI 10.1007/s12591-010-0067-x is an 2010 online pre-publication.

G. J. Rinkus. *A combinatorial neural network exhibiting episodic and semantic memory properties for spatiotemporal patterns.* Phd thesis, Boston University, 1996.

H. Ritter and T. Kohonen. Self-organizing semantic maps. *Biological Cybernetics*, 61: 241–254, 1989.

L. Shastri. Advances in Shruti – a neurally motivated model of relational knowledge representation and rapid inference using temporal synchrony. *Artificial Intelligence*, 11:79–108, 1999.

H. Sompolinsky and I. Kanter. Temporal association in asymmetric neural networks. *Physical Review Letters*, 57(22):2861–2864, 1986.

D. Sussillo and L. Abbott. Transferring learning from external to internal weights in echo-state networks with sparse connectivity. *PLoS ONE*, 7(5):e37372, 2012.

I. Sutskever, G. E. Hinton, and G. W. Taylor. The recurrent temporal restricted boltzmann machine. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21 (NIPS 08)*, pages 1601–1608, 2009.

I. Sutskever, J. Martens, and G. Hinton. Generating text with recurrent neural networks. In *ICML 2011 (online)*, 2011. URL `http://www.icml-2011.org/papers.php`.

G. W. Taylor, G. E. Hinton, and S. T. Roweis. Two distributed-state models for generating high-dimensional time series. *Journal of Machine Learning Research*, 12(March):1025–1068, 2011.

D. J. Willshaw, O. P. Buneman, and H. C. Longuet-Higgins. Non-holographic associative memory. *Nature*, 222(June 7):960–962, 1969.

J. Wright and I. Jordanov. Intelligent approaches in locomotion. In *WCCI 2012 IEEE World Congress on Computational Intelligence*, pages 1–8, 2012. doi: dx.doi.org/10.1109/IJCNN.2012.6252537.

F. wyffels and B. Schrauwen. Design of a central pattern generator using reservoir computing for learning human motion. In *Advanced Technologies for Enhanced Quality of Life, 2009. AT-EQUAL '09*, pages 118–122. IEEE, 2009.

Journal of Machine Learning Research 18 (2017) 1-45 Submitted 3/16; Revised 8/16; Published 1/17

Automatic Differentiation Variational Inference

Alp Kucukelbir ALP@CS.COLUMBIA.EDU
Data Science Institute, Department of Computer Science
Columbia University
New York, NY 10027, USA

Dustin Tran DUSTIN@CS.COLUMBIA.EDU
Department of Computer Science
Columbia University
New York, NY 10027, USA

Rajesh Ranganath RAJESHR@CS.PRINCETON.EDU
Department of Computer Science
Princeton University
Princeton, NJ 08540, USA

Andrew Gelman GELMAN@STAT.COLUMBIA.EDU
Data Science Institute, Departments of Political Science and Statistics
Columbia University
New York, NY 10027, USA

David M. Blei DAVID.BLEI@COLUMBIA.EDU
Data Science Institute, Departments of Computer Science and Statistics
Columbia University
New York, NY 10027, USA

Editor: David Barber

Abstract

Probabilistic modeling is iterative. A scientist posits a simple model, fits it to her data, refines it according to her analysis, and repeats. However, fitting complex models to large data is a bottleneck in this process. Deriving algorithms for new models can be both mathematically and computationally challenging, which makes it difficult to efficiently cycle through the steps. To this end, we develop automatic differentiation variational inference (ADVI). Using our method, the scientist only provides a probabilistic model and a dataset, nothing else. ADVI automatically derives an efficient variational inference algorithm, freeing the scientist to refine and explore many models. ADVI supports a broad class of models—no conjugacy assumptions are required. We study ADVI across ten modern probabilistic models and apply it to a dataset with millions of observations. We deploy ADVI as part of Stan, a probabilistic programming system.

Keywords: Bayesian inference, approximate inference, probabilistic programming

1. Introduction

We develop an automatic method that derives variational inference algorithms for complex probabilistic models. We implement our method in a probabilistic programming system that lets a user specify a model in an intuitive programming language and then compiles that model into an inference executable. Our method enables automatic inference with large datasets on a practical class of modern probabilistic models.

The context of this research is the field of probabilistic modeling, which has emerged as a powerful language for customized data analysis. Probabilistic modeling lets us express our assumptions about data in a formal mathematical way, and then derive algorithms that use those assumptions to compute about an observed dataset. It has had an impact on myriad applications in both statistics and machine learning, including natural language processing, speech recognition, computer vision, population genetics, and computational neuroscience.

Figure 14 presents an example. Say we want to analyze how people navigate a city by car. We have a dataset of all the taxi rides taken over the course of a year: 1.7 million trajectories. To explore patterns in this data, we propose a mixture model with an unknown number of components. This is a non-conjugate model that we seek to fit to a large dataset. Previously, we would have to manually derive an inference algorithm that scales to large data. With our method, we write a probabilistic program and compile it; we can then fit the model in minutes and analyze the results with ease.

Probabilistic modeling leads to a natural research cycle. First, we use our domain knowledge to posit a simple model that includes latent variables; then, we use an inference algorithm to infer those variables from our data; next, we analyze our results and identify where the model works and where it falls short; last, we refine the model and repeat the process. When we cycle through these steps, we find expressive, interpretable, and useful models (Gelman et al., 2013; Blei, 2014). One of the broad goals of machine learning is to make this process easy.

Looping around this cycle, however, is not easy. The data we study are often large and complex; accordingly, we want to propose rich probabilistic models and scale them up. But using such models requires complex algorithms that are difficult to derive, implement, and scale. The bottleneck is this computation. The efforts involved in deriving, programming, debugging, and scaling each model precludes us from taking full advantage of the probabilistic modeling cycle.

It is this problem that motivates the ideas of probabilistic programming and automated inference. Probabilistic programming allows a user to write a probability model as a computer program and then compile that program into an efficient inference executable. Automated inference is the backbone of such a system—it inputs a probability model, expressed as a program, and outputs an efficient algorithm for computing with it. Previous approaches to automatic inference have mainly relied on Markov chain Monte Carlo (MCMC) algorithms. The results have been successful, but automated MCMC can be too slow for many real-world applications.

We approach the problem through variational inference (Jordan et al., 1999; Wainwright and Jordan, 2008), a faster alternative to MCMC that has been used in many large-scale problems (Hoffman et al., 2013; Blei et al., 2016). Though it is a promising method, developing a variational inference algorithm still requires tedious model-specific derivations and implementation; it has not seen widespread use in probabilistic programming. Here we automate the process of deriving scalable variational inference algorithms. We build on recent ideas in variational inference to leverage some of the capabilities of probabilistic programming systems, namely the ability to transform the space of latent variables and to automate derivatives of the joint distribution. The result, called automatic differentiation variational inference (ADVI), provides an automated solution to variational inference: the inputs are a probabilistic model and a dataset; the outputs are posterior inferences about the model's

©2017 Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman and David M. Blei.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-107.html.

latent variables.[1,2] We implement and deploy ADVI as part of Stan, a probabilistic programming system (Stan Development Team, 2016).

ADVI resolves the computational bottleneck of the probabilistic modeling cycle. We can easily propose a probabilistic model, analyze a large dataset, and revise the model, without worrying about computation. ADVI enables this cycle by providing automated and scalable variational inference for an expansive class of models. Sections 3 and 4 present ten direct applications of ADVI to modern probabilistic modeling examples, including an iterative analysis of 1.7 million taxi trajectories.

Technical summary. Formally, a probabilistic model defines a joint distribution of observations $\mathbf{x}$ and latent variables $\boldsymbol{\theta}$, $p(\mathbf{x}, \boldsymbol{\theta})$. The inference problem is to compute the posterior, the conditional distribution of the latent variables given the observations $p(\boldsymbol{\theta} \mid \mathbf{x})$. The posterior reveals patterns in the data and forms predictions through the posterior predictive distribution. The problem is that, for many models, the posterior is not tractable to compute.

Variational inference turns the task of computing a posterior into an optimization problem. We posit a parameterized family of distributions $q(\boldsymbol{\theta}) \in \mathcal{Q}$ and then find the member of that family that minimizes the Kullback-Leibler (KL) divergence to the exact posterior. Traditionally, using a variational inference algorithm requires the painstaking work of developing and implementing a custom optimization routine: specifying a variational family appropriate to the model, computing the corresponding objective function, taking derivatives, and running a gradient-based or coordinate-ascent optimization.

ADVI solves this problem automatically. The user specifies the model, expressed as a program, and ADVI automatically generates a corresponding variational algorithm. The idea is to first automatically transform the inference problem into a common space and then to solve the variational optimization problem. Solving the problem in this common space solves variational inference for all models in a large class. In more detail, ADVI follows these steps.

1. ADVI transforms the model into one with unconstrained real-valued latent variables. Specifically, it transforms $p(\mathbf{x}, \boldsymbol{\theta})$ to $p(\mathbf{x}, \boldsymbol{\zeta})$, where the mapping from $\boldsymbol{\theta}$ to $\boldsymbol{\zeta}$ is built into the joint distribution. This removes all original constraints on the latent variables $\boldsymbol{\theta}$. ADVI then defines the corresponding variational problem on the transformed variables, that is, to minimize $\mathrm{KL}\left(q(\boldsymbol{\zeta}) \parallel p(\boldsymbol{\zeta} \mid \mathbf{x})\right)$. With this transformation, all latent variables are defined on the same space. ADVI can now use a single variational family for all models.

2. ADVI recasts the gradient of the variational objective function as an expectation over q. This involves the gradient of the log joint with respect to the latent variables $\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}, \boldsymbol{\theta})$. Expressing the gradient as an expectation opens the door to Monte Carlo methods for approximating it (Robert and Casella, 1999; Ranganath et al., 2014).

3. ADVI further reparameterizes the gradient in terms of a standard Gaussian. To do this, it uses another transformation, this time within the variational family. This second transformation enables ADVI to efficiently compute Monte Carlo approximations—it needs only to sample from a standard Gaussian (Kingma and Welling, 2014; Rezende et al., 2014).

4. ADVI uses noisy gradients to optimize the variational distribution (Robbins and Monro, 1951). An adaptively tuned step-size sequence provides good convergence in practice.

1. This paper extends the method presented in Kucukelbir et al. (2015).
2. Automatic differentation has enjoyed a recent resurgence in machine learning; see Section 2.7.

Each step above is carefully designed to make ADVI work "out of the box" for a practical class of modern probabilistic models. This focus on developing an automated inference algorithm differentiates ADVI from other "black box" variational methods (Ranganath et al., 2014; Ruiz et al., 2016b).

We deploy ADVI in the Stan probabilistic programming system, which gives us two important types of automatic computation around probabilistic models. First, Stan provides a library of transformations—ways to convert a variety of constrained latent variables (e.g., positive reals) to be unconstrained, without changing the underlying joint distribution. Stan's library of transformations helps us with step 1 above. Second, Stan implements automatic differentiation to calculate $\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}, \boldsymbol{\theta})$ (Carpenter et al., 2015; Baydin et al., 2015). These derivatives are crucial in step 2, when computing the gradient of the ADVI objective.

Organization of paper. Section 2 develops the recipe that makes ADVI. We expose the details of each of the steps above and present a concrete algorithm. Section 3 studies the properties of ADVI. We explore its accuracy, its stochastic nature, and its sensitivity to transformations. Section 4 applies ADVI to an array of probability models. We compare its speed to MCMC sampling techniques and present a case study using a dataset with millions of observations. Section 5 concludes the paper with a discussion.

2. Automatic Differentiation Variational Inference

ADVI offers a recipe for automating the computations involved in variational inference. The strategy is as follows: transform the latent variables of the model into a common space, choose a variational approximation in the common space, and use generic computational techniques to solve the variational problem.

2.1 Differentiable Probability Models

We begin by defining the class of probability models that ADVI supports. Consider a dataset $\mathbf{x} = x_{1:N}$ with N observations. Each x_n is a realization of a discrete or continuous (multivariate) random variable. The likelihood $p(\mathbf{x} \mid \boldsymbol{\theta})$ relates the observations to a set of latent random variables $\boldsymbol{\theta}$. A Bayesian model posits a prior density $p(\boldsymbol{\theta})$ on the latent variables. Combining the likelihood with the prior gives the joint density $p(\mathbf{x}, \boldsymbol{\theta}) = p(\mathbf{x} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta})$. The goal of inference is to compute the posterior density $p(\boldsymbol{\theta} \mid \mathbf{x})$, which describes how the latent variables vary, conditioned on data.

Many posterior densities are not tractable; their normalizing constants lack analytic (closed-form) solutions. Thus we often seek to approximate the posterior. ADVI approximates the posterior of differentiable probability models. Members of this class of models have continuous latent variables $\boldsymbol{\theta}$ and a gradient of the log-joint with respect to them $\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}, \boldsymbol{\theta})$. The gradient is valid within the support of the prior

$$\mathrm{supp}(p(\boldsymbol{\theta})) = \left\{ \boldsymbol{\theta} \mid \boldsymbol{\theta} \in \mathbb{R}^K \text{ and } p(\boldsymbol{\theta}) > 0 \right\} \subseteq \mathbb{R}^K,$$

where K is the dimension of the latent variable space. This support set is important: it will play a role later in the paper. We make no assumptions about conjugacy, either full (Diaconis and Ylvisaker, 1979) or conditional (Hoffman et al., 2013).

Consider a model that contains a Poisson likelihood with unknown rate $p(x \mid \theta)$. The observed variable x is discrete; the latent rate θ is continuous and positive. Place a Weibull prior on θ, defined over the positive real numbers. The resulting joint density describes a nonconjugate probability model:

the posterior distribution of θ is not in the same class as the prior. (The conjugate prior would be a Gamma.) However, it is in the class of differentiable models. Its partial derivative $\partial/\partial\theta \log p(x, \theta)$ is valid within the support of the Weibull distribution, $\text{supp}(p(\theta)) = \mathbb{R}_{>0} \subset \mathbb{R}$. While this model would be a challenge for classical variational inference, it is not for ADVI.

Many machine learning models are differentiable. For example: linear and logistic regression, matrix factorization with continuous or discrete observations, linear dynamical systems, and Gaussian processes. (See Table 1.) At first blush, the restriction to differentiable random variables may seem to leave out other common machine learning models, such as mixture models and topic models. However, marginalizing out the discrete variables in the likelihoods of these models renders them differentiable.

Generalized linear models	*(e.g., linear / logistic / probit)*
Mixture models	*(e.g., mixture of Gaussians)*
Deep exponential families	*(e.g., deep latent Gaussian models)*
Topic models	*(e.g., latent Dirichlet allocation)*
Linear dynamical systems	*(e.g., state space models)*
Gaussian process models	*(e.g., regression / classification)*

Table 1: Popular differentiable probability models in machine learning.

Marginalization is not tractable for all models, such as the Ising model, sigmoid belief networks, and (untruncated) Bayesian nonparametric models, such as Dirichlet process mixtures (Antoniak, 1974). These are not differentiable probability models.

2.2 Variational Inference

Variational inference (VI) turns approximate posterior inference into an optimization problem (Wainwright and Jordan, 2008; Blei et al., 2016). Consider a family of approximating densities of the latent variables $q(\boldsymbol{\theta}\,;\,\boldsymbol{\phi})$, parameterized by a vector $\boldsymbol{\phi} \in \boldsymbol{\Phi}$. VI finds the parameters that minimize the KL divergence to the posterior,

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}\in\boldsymbol{\Phi}} \text{KL}\left(q(\boldsymbol{\theta}\,;\,\boldsymbol{\phi}) \parallel p(\boldsymbol{\theta} \mid \mathbf{x})\right). \tag{1}$$

The optimized $q(\boldsymbol{\theta}\,;\,\boldsymbol{\phi}^*)$ then serves as an approximation to the posterior.

The KL divergence lacks an analytic form because it involves the posterior. Instead we maximize the evidence lower bound (ELBO)

$$\mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{q(\boldsymbol{\theta})}\big[\log p(\mathbf{x}, \boldsymbol{\theta})\big] - \mathbb{E}_{q(\boldsymbol{\theta})}\big[\log q(\boldsymbol{\theta}\,;\,\boldsymbol{\phi})\big]. \tag{2}$$

The first term is an expectation of the joint density under the approximation, and the second is the entropy of the variational density. The ELBO is equal to the negative KL divergence up to the constant $\log p(\mathbf{x})$. Maximizing the ELBO minimizes the KL divergence (Jordan et al., 1999; Bishop, 2006).

Optimizing the KL divergence implies a constraint that the support of the approximation $q(\boldsymbol{\theta}\,;\,\boldsymbol{\phi})$ lie within the support of the posterior $p(\boldsymbol{\theta} \mid \mathbf{x})$.[3] With this constraint made explicit, the optimization

3. If $\text{supp}(q) \not\subseteq \text{supp}(p)$ then outside the support of p we have $\text{KL}\left(q \parallel p\right) = \mathbb{E}_q[\log q] - \mathbb{E}_q[\log p] = \infty$.

problem from Equation (1) becomes

$$\boldsymbol{\phi}^* = \arg\max_{\boldsymbol{\phi}\in\boldsymbol{\Phi}} \mathcal{L}(\boldsymbol{\phi}) \quad \text{such that} \quad \text{supp}(q(\boldsymbol{\theta}\,;\,\boldsymbol{\phi})) \subseteq \text{supp}(p(\boldsymbol{\theta} \mid \mathbf{x})). \tag{3}$$

We explicitly include this constraint because we have not specified the form of the variational approximation; we must ensure that $q(\boldsymbol{\theta}\,;\,\boldsymbol{\phi})$ stays within the support of the posterior.

The support of the posterior, however, may also be unknown. So, we further assume that the support of the posterior equals that of the prior, $\text{supp}(p(\boldsymbol{\theta} \mid \mathbf{x})) = \text{supp}(p(\boldsymbol{\theta}))$. This is a benign assumption, which holds for most models considered in machine learning. In detail, it holds when the likelihood does not constrain the prior; i.e., the likelihood must be positive over the sample space for any $\boldsymbol{\theta}$ drawn from the prior.

Our recipe for automating VI. The traditional way of solving Equation (3) is difficult. We begin by choosing a variational family $q(\boldsymbol{\theta}\,;\,\boldsymbol{\phi})$ that, by definition, satisfies the support matching constraint. We compute the expectations in the ELBO, either analytically or through approximation. We then decide on a strategy to maximize the ELBO. For instance, we might use coordinate ascent by iteratively updating the components of $\boldsymbol{\phi}$. Or, we might follow gradients of the ELBO with respect to $\boldsymbol{\phi}$ while staying within $\boldsymbol{\Phi}$. Finally, we implement, test, and debug software that performs the above. Each step requires expert thought and analysis in the service of a single algorithm for a single model.

In contrast, our approach allows a user to define any differentiable probability model for which we automate the process of developing a corresponding VI algorithm. Our recipe for automating VI has three ingredients. First, we automatically transform the support of the latent variables $\boldsymbol{\theta}$ to the real coordinate space (Section 2.3); this lets us choose from a variety of variational distributions q without worrying about the support matching constraint (Section 2.4). Second, we compute the ELBO for any model using Monte Carlo (MC) integration, which only requires being able to sample from the variational distribution (Section 2.5). Third, we employ stochastic gradient ascent to maximize the ELBO and use automatic differentiation to compute gradients without any user input (Section 2.6). With these tools, we can develop a generic method that automatically solves the variational optimization problem for a large class of models.

2.3 Automatic Transformation of Constrained Variables

We begin by transforming the support of the latent variables $\boldsymbol{\theta}$ such that they live in the real coordinate space $\mathbb{R}^K$. Once we transform the joint density, we can choose the variational approximation independent of the model.

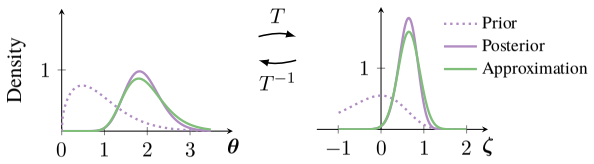
Define a one-to-one differentiable function

$$T : \text{supp}(p(\boldsymbol{\theta})) \to \mathbb{R}^K, \tag{4}$$

and identify the transformed variables as $\boldsymbol{\zeta} = T(\boldsymbol{\theta})$. The transformed joint density $p(\mathbf{x}, \boldsymbol{\zeta})$ is a function of $\boldsymbol{\zeta}$; it has the representation

$$p(\mathbf{x}, \boldsymbol{\zeta}) = p\left(\mathbf{x}, T^{-1}(\boldsymbol{\zeta})\right) \left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|,$$

where $p(\mathbf{x}, \boldsymbol{\theta} = T^{-1}(\boldsymbol{\zeta}))$ is the joint density in the original latent variable space, and $J_{T^{-1}}(\boldsymbol{\zeta})$ is the Jacobian of the inverse of T. Transformations of continuous probability densities require a Jacobian; it accounts for how the transformation warps unit volumes and ensures that the transformed density integrates to one (Olive, 2014). (See Appendix A.)



(a) Latent variable space **(b)** Real coordinate space

Figure 1: Transforming the latent variable to real coordinate space. The purple line is the posterior. The green line is the approximation. **(a)** The latent variable space is $\mathbb{R}_{>0}$. **(a→b)** T transforms the latent variable space to $\mathbb{R}$. **(b)** The variational approximation is a Gaussian in real coordinate space.

Consider again our Weibull-Poisson example from Section 2.1. The latent variable θ lives in $\mathbb{R}_{>0}$. The logarithm $\zeta = T(\theta) = \log(\theta)$ transforms $\mathbb{R}_{>0}$ to the real line $\mathbb{R}$. Its Jacobian adjustment is the derivative of the inverse of the logarithm $\left|\det J_{T^{-1}(\zeta)}\right| = \exp(\zeta)$. The transformed density is

$$p(x, \zeta) = \text{Poisson}(x \mid \exp(\zeta)) \times \text{Weibull}(\exp(\zeta) \mid 1.5, 1) \times \exp(\zeta).$$

Figures 1a and 1b depict this transformation.

As we describe in the introduction, we implement our algorithm in Stan (Stan Development Team, 2016). Stan maintains a library of transformations and their corresponding Jacobians. Specifically, it provides various transformations for upper and lower bounds, simplex and ordered vectors, and structured matrices such as covariance matrices and Cholesky factors. With Stan, we can automatically transform the joint density of any differentiable probability model to one with real-valued latent variables. (See Figure 2.)

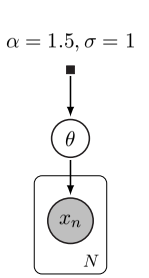
2.4 Variational Approximations in Real Coordinate Space

After the transformation, the latent variables $\boldsymbol{\zeta}$ have support in the real coordinate space $\mathbb{R}^K$. We have a choice of variational approximations in this space. Here, we consider Gaussian distributions (Figure 1b); these implicitly induce non-Gaussian variational distributions in the original latent variable space (Figure 1a).

Mean-field Gaussian. One option is to posit a factorized (mean-field) Gaussian variational approximation

$$q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi}) = \text{Normal}\left(\boldsymbol{\zeta} \mid \boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}^2)\right) = \prod_{k=1}^{K} \text{Normal}\left(\zeta_k \mid \mu_k, \sigma_k^2\right),$$

where the vector $\boldsymbol{\phi} = (\mu_1, \cdots, \mu_K, \sigma_1^2, \cdots, \sigma_K^2)$ concatenates the mean and variance of each Gaussian factor. Since the variance parameters must always be positive, the variational parameters live in the set $\boldsymbol{\Phi} = \{\mathbb{R}^K, \mathbb{R}^K_{>0}\}$. Re-parameterizing the mean-field Gaussian removes this constraint. Consider the logarithm of the standard deviations, $\boldsymbol{\omega} = \log(\boldsymbol{\sigma})$, applied element-wise. The support of $\boldsymbol{\omega}$ is now the real coordinate space and $\boldsymbol{\sigma}$ is always positive. The mean-field Gaussian becomes $q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi}) = \text{Normal}\left(\boldsymbol{\zeta} \mid \boldsymbol{\mu}, \text{diag}(\exp(\boldsymbol{\omega})^2)\right)$, where the vector $\boldsymbol{\phi} = (\mu_1, \cdots, \mu_K, \omega_1, \cdots, \omega_K)$ concatenates the mean and logarithm of the standard deviation of each factor. Now, the variational parameters are unconstrained in $\mathbb{R}^{2K}$.




```
data {
  int N;    // number of observations
  int x[N]; // discrete-valued observations
}
parameters {
  // latent variable, must be positive
  real<lower=0> theta;
}
model {
  // non-conjugate prior for latent variable
  theta ~ weibull(1.5, 1);

  // likelihood
  for (n in 1:N)
    x[n] ~ poisson(theta);
}
```

Figure 2: Specifying a simple nonconjugate probability model in Stan.

Full-rank Gaussian. Another option is to posit a full-rank Gaussian variational approximation

$$q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi}) = \text{Normal}\left(\boldsymbol{\zeta} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}\right),$$

where the vector $\boldsymbol{\phi} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ concatenates the mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. To ensure that $\boldsymbol{\Sigma}$ always remains positive semidefinite, we re-parameterize the covariance matrix using a Cholesky factorization, $\boldsymbol{\Sigma} = \mathbf{L}\mathbf{L}^\top$. We use the non-unique definition of the Cholesky factorization where the diagonal elements of $\mathbf{L}$ need not be positively constrained (Pinheiro and Bates, 1996). Therefore $\mathbf{L}$ lives in the unconstrained space of lower-triangular matrices with $K(K+1)/2$ real-valued entries. The full-rank Gaussian becomes $q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi}) = \text{Normal}\left(\boldsymbol{\zeta} \mid \boldsymbol{\mu}, \mathbf{L}\mathbf{L}^\top\right)$, where the variational parameters $\boldsymbol{\phi} = (\boldsymbol{\mu}, \mathbf{L})$ are unconstrained in $\mathbb{R}^{K+K(K+1)/2}$.

The full-rank Gaussian generalizes the mean-field Gaussian approximation. The off-diagonal terms in the covariance matrix $\boldsymbol{\Sigma}$ capture posterior correlations across latent random variables.[4] This leads to a more accurate posterior approximation than the mean-field Gaussian; however, it comes at a computational cost. Various low-rank approximations to the covariance matrix reduce this cost, yet limit its ability to model complex posterior correlations (Seeger, 2010; Challis and Barber, 2013).

The choice of a Gaussian. Choosing a Gaussian distribution may call to mind the Laplace approximation technique, where a second-order Taylor expansion around the maximum-a-posteriori estimate gives a Gaussian approximation to the posterior. However, using a Gaussian variational approximation is not equivalent to the Laplace approximation (Opper and Archambeau, 2009). Our approach is distinct in another way: the posterior approximation in the original latent variable space (Figure 1a) is non-Gaussian.

The implicit variational density. The transformation T from Equation (4) maps the support of the latent variables to the real coordinate space. Thus, its inverse T^{-1} maps back to the support of the latent variables. This implicitly defines the variational approximation in the original latent variable space as $q\left(T(\boldsymbol{\theta})\,;\,\boldsymbol{\phi}\right)\left|\det J_T(\boldsymbol{\theta})\right|$. The transformation ensures that the support of this approximation is always bounded by that of the posterior in the original latent variable space.

Sensitivity to T. There are many ways to transform the support a variable to the real coordinate space. The form of the transformation directly affects the shape of the variational approximation in the original latent variable space. We study sensitivity to the choice of transformation in Section 3.3.

4. This is a form of structured mean-field variational inference (Wainwright and Jordan, 2008; Barber, 2012).

2.5 The Variational Problem in Real Coordinate Space

Here is the story so far. We began with a differentiable probability model $p(\mathbf{x}, \boldsymbol{\theta})$. We transformed the latent variables into $\boldsymbol{\zeta}$, which live in the real coordinate space. We defined variational approximations in the transformed space. Now, we consider the variational optimization problem.

Write the variational objective function, the ELBO, in real coordinate space as

$$\mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})}\left[\log p\left(\mathbf{x}, T^{-1}(\boldsymbol{\zeta})\right) + \log\left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|\right] + \mathbb{H}\big[q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})\big]. \tag{5}$$

The inverse of the transformation T^{-1} appears in the joint model, along with the determinant of the Jacobian adjustment. The ELBO is a function of the variational parameters $\boldsymbol{\phi}$ and the entropy $\mathbb{H}$, both of which depend on the variational approximation. (Derivation in Appendix B.)

Now, we can freely optimize the ELBO in the real coordinate space without worrying about the support matching constraint. The optimization problem from Equation (3) becomes

$$\boldsymbol{\phi}^* = \arg\max_{\boldsymbol{\phi}} \mathcal{L}(\boldsymbol{\phi}) \tag{6}$$

where the parameter vector $\boldsymbol{\phi}$ lives in some appropriately dimensioned real coordinate space. This is an unconstrained optimization problem that we can solve using gradient ascent. Traditionally, this would require manual computation of gradients. Instead, we develop a stochastic gradient ascent algorithm that uses automatic differentiation to compute gradients and MC integration to approximate intractable expectations.

We cannot directly use automatic differentiation on the ELBO. This is because the ELBO involves an intractable expectation. However, we can automatically differentiate the functions inside the expectation. (The model p and transformation T are both easy to represent as computer functions (Baydin et al., 2015).) To apply automatic differentiation, we want to push the gradient operation inside the expectation. To this end, we employ one final transformation: elliptical standardization[5] (Härdle and Simar, 2012).

Elliptical standardization. Consider a transformation $S_{\boldsymbol{\phi}}$ that absorbs the variational parameters $\boldsymbol{\phi}$; this converts the Gaussian variational approximation into a standard Gaussian. In the mean-field case, the standardization is $\boldsymbol{\eta} = S_{\boldsymbol{\phi}}(\boldsymbol{\zeta}) = \operatorname{diag}\left(\exp\left(\boldsymbol{\omega}\right)\right)^{-1}\left(\boldsymbol{\zeta} - \boldsymbol{\mu}\right)$. In the full-rank Gaussian, the standardization is $\boldsymbol{\eta} = S_{\boldsymbol{\phi}}(\boldsymbol{\zeta}) = \mathbf{L}^{-1}(\boldsymbol{\zeta} - \boldsymbol{\mu})$.

In both cases, the standardization encapsulates the variational parameters; in return it gives a fixed variational density

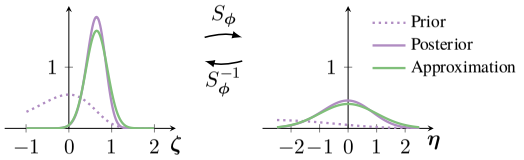
$$q(\boldsymbol{\eta}) = \text{Normal}\left(\boldsymbol{\eta} \mid \mathbf{0}, \mathbf{I}\right) = \prod_{k=1}^{K} \text{Normal}\left(\eta_k \mid 0, 1\right),$$

as shown in Figures 3a and 3b.

The standardization transforms the variational problem from Equation (5) into

$$\boldsymbol{\phi}^* = \arg\max_{\boldsymbol{\phi}} \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta}\,;\,\mathbf{0},\mathbf{I})}\left[\log p\left(\mathbf{x}, T^{-1}(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta}))\right) + \log\left|\det J_{T^{-1}}\left(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta})\right)\right|\right] + \mathbb{H}\big[q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})\big].$$

5. Also known as a "coordinate transformation" (Rezende et al., 2014), an "invertible transformation" (Titsias and Lázaro-Gredilla, 2014), and the "re-parameterization trick" (Kingma and Welling, 2014).



(a) Real coordinate space **(b)** Standardized space

Figure 3: Elliptical standardization. The purple line is the posterior. The green line is the approximation. **(a)** The variational approximation is a Gaussian in real coordinate space. **(a→b)** $S_{\boldsymbol{\phi}}$ absorbs the parameters of the Gaussian. **(b)** We maximize the ELBO in the standardized space, with a fixed approximation. The green line is a standard Gaussian.

The expectation is now in terms of a standard Gaussian density. The Jacobian of elliptical standardization evaluates to one, because the Gaussian distribution is a member of the location-scale family: standardizing a Gaussian gives another Gaussian distribution. (See Appendix A.)

We do not need to transform the entropy term as it does not depend on the model or the transformation; we have a simple analytic form for the entropy of a Gaussian and its gradient. We implement these once and reuse for all models.

2.6 Stochastic Optimization

We now reach the final step: stochastic optimization of the variational objective function.

Computing gradients. Since the expectation is no longer dependent on $\boldsymbol{\phi}$, we can directly calculate its gradient. Push the gradient inside the expectation and apply the chain rule to get

$$\nabla_{\boldsymbol{\mu}}\mathcal{L} = \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta})}\left[\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}, \boldsymbol{\theta}) \nabla_{\boldsymbol{\zeta}} T^{-1}(\boldsymbol{\zeta}) + \nabla_{\boldsymbol{\zeta}} \log\left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|\right]. \tag{7}$$

We obtain gradients with respect to $\boldsymbol{\omega}$ (mean-field) and $\mathbf{L}$ (full-rank) in a similar fashion

$$\nabla_{\boldsymbol{\omega}}\mathcal{L} = \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta})}\left[\left(\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}, \boldsymbol{\theta}) \nabla_{\boldsymbol{\zeta}} T^{-1}(\boldsymbol{\zeta}) + \nabla_{\boldsymbol{\zeta}} \log\left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|\right) \boldsymbol{\eta}^\top \operatorname{diag}(\exp(\boldsymbol{\omega}))\right] + \mathbf{1} \tag{8}$$

$$\nabla_{\mathbf{L}}\mathcal{L} = \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta})}\left[\left(\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}, \boldsymbol{\theta}) \nabla_{\boldsymbol{\zeta}} T^{-1}(\boldsymbol{\zeta}) + \nabla_{\boldsymbol{\zeta}} \log\left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|\right) \boldsymbol{\eta}^\top\right] + (\mathbf{L}^{-1})^\top. \tag{9}$$

(Derivations in Appendix C.)

We can now compute the gradients inside the expectation with automatic differentiation. The only thing left is the intractable expectation. MC integration provides a simple approximation: draw samples from the standard Gaussian and evaluate the empirical mean of the gradients within the expectation (Appendix D). In practice a single sample suffices. (We study this in detail in Section 3.2 and in the experiments in Section 4.)

This gives noisy unbiased gradients of the ELBO for any differentiable probability model. We can use these gradients in a stochastic optimization routine to automate variational inference.

Stochastic gradient ascent. Equipped with noisy unbiased gradients of the ELBO, ADVI implements stochastic gradient ascent (Algorithm 1). This algorithm is guaranteed to converge to a local maximum of the ELBO under certain conditions on the step-size sequence.[6] Stochastic gradient ascent falls

6. This is also called a *learning rate* or *schedule* in the machine learning community.

Algorithm 1: Automatic differentiation variational inference (ADVI)

Input: Dataset $\mathbf{x} = x_{1:N}$, model $p(\mathbf{x}, \boldsymbol{\theta})$.
Set iteration counter $i = 1$.
Initialize $\boldsymbol{\mu}^{(1)} = \mathbf{0}$.
Initialize $\boldsymbol{\omega}^{(1)} = \mathbf{0}$ (mean-field) or $\mathbf{L}^{(1)} = \mathbf{I}$ (full-rank).
Determine η via a search over finite values.
while *change in* ELBO *is above some threshold* **do**
  Draw M samples $\boldsymbol{\eta}_m \sim \text{Normal}(\mathbf{0}, \mathbf{I})$ from the standard multivariate Gaussian.
  Approximate $\nabla_{\boldsymbol{\mu}}\mathcal{L}$ using MC integration (Equation (7)).
  Approximate $\nabla_{\boldsymbol{\omega}}\mathcal{L}$ or $\nabla_{\mathbf{L}}\mathcal{L}$ using MC integration (Equations (8) and (9)).
  Calculate step-size $\boldsymbol{\rho}^{(i)}$ (Equation (10)).
  Update $\boldsymbol{\mu}^{(i+1)} \longleftarrow \boldsymbol{\mu}^{(i)} + \text{diag}(\boldsymbol{\rho}^{(i)})\nabla_{\boldsymbol{\mu}}\mathcal{L}$.
  Update $\boldsymbol{\omega}^{(i+1)} \longleftarrow \boldsymbol{\omega}^{(i)} + \text{diag}(\boldsymbol{\rho}^{(i)})\nabla_{\boldsymbol{\omega}}\mathcal{L}$ or $\mathbf{L}^{(i+1)} \longleftarrow \mathbf{L}^{(i)} + \text{diag}(\boldsymbol{\rho}^{(i)})\nabla_{\mathbf{L}}\mathcal{L}$.
  Increment iteration counter.
end
Return $\boldsymbol{\mu}^* \longleftarrow \boldsymbol{\mu}^{(i)}$.
Return $\boldsymbol{\omega}^* \longleftarrow \boldsymbol{\omega}^{(i)}$ or $\mathbf{L}^* \longleftarrow \mathbf{L}^{(i)}$.

under the class of stochastic approximations, where Robbins and Monro (1951) established a pair of conditions that ensure convergence. Many sequences satisfy these criteria, but their specific forms impact the success of stochastic gradient ascent in practice. We describe an adaptive step-size sequence for ADVI below.

Adaptive step-size sequence. Adaptive step-size sequences retain (possibly infinite) memory about past gradients and adapt to the high-dimensional curvature of the ELBO optimization space (Amari, 1998; Duchi et al., 2011; Ranganath et al., 2013; Kingma and Adam, 2015). These sequences enjoy theoretical bounds on their convergence rates. However, in practice, they can be slow to converge. The empirically justified RMSPROP sequence (Tieleman and Hinton, 2012) converges quickly in practice but lacks any convergence guarantees. We propose a new step-size sequence which effectively combines both approaches.

Consider the step-size $\boldsymbol{\rho}^{(i)}$ and a gradient vector $\mathbf{g}^{(i)}$ at iteration i. We define the kth element of $\boldsymbol{\rho}^{(i)}$ as

$$\rho_k^{(i)} = \eta \times i^{-1/2+\epsilon} \times \left(\tau + \sqrt{s_k^{(i)}}\right)^{-1}, \tag{10}$$

where we apply the following recursive update

$$s_k^{(i)} = \alpha g_k^{2(i)} + (1-\alpha)s_k^{(i-1)}, \tag{11}$$

with an initialization of $s_k^{(1)} = g_k^{2(1)}$.

The first factor $\eta \in \mathbb{R}_{>0}$ controls the scale of the step-size sequence. It mainly affects the beginning of the optimization. We adaptively tune η by searching over $\eta \in \{0.01, 0.1, 1, 10, 100\}$ using a subset of the data and selecting the value that leads to the fastest convergence (Bottou, 2012).

The middle term $i^{-1/2+\epsilon}$ decays as a function of the iteration i. We set $\epsilon = 10^{-16}$, a small value that guarantees that the step-size sequence satisfies the Robbins and Monro (1951) conditions.

The last term adapts to the curvature of the ELBO optimization space. Memory about past gradients are processed in Equation (11). The weighting factor $\alpha \in (0, 1)$ defines a compromise of old and new gradient information, which we set to 0.1. The quantity s_k converges to a non-zero constant. Without the previous decaying term, this would lead to possibly large oscillations around a local optimum of the ELBO. The additional perturbation $\tau > 0$ prevents division by zero and down-weights early iterations. In practice the step-size is not sensitive to this value (Hoffman et al., 2013), so we set $\tau = 1$.

Complexity and data subsampling. ADVI has complexity $\mathcal{O}(NMK)$ per iteration, where N is the number of data points, M is the number of MC samples (typically between 1 and 10), and K is the number of latent variables. Classical VI which hand-derives a coordinate ascent algorithm has complexity $\mathcal{O}(NK)$ per pass over the dataset. The added complexity of automatic differentiation over analytic gradients is roughly constant (Carpenter et al., 2015; Baydin et al., 2015).

We scale ADVI to large datasets using stochastic optimization with data subsampling (Hoffman et al., 2013; Titsias and Lázaro-Gredilla, 2014). The adjustment to Algorithm 1 is simple: sample a minibatch of size $B \ll N$ from the dataset and scale the likelihood of the model by N/B (Hoffman et al., 2013). The stochastic extension of ADVI has a per-iteration complexity $\mathcal{O}(BMK)$.

In Sections 4.3 and 4.4, we apply this stochastic extension to analyze datasets with hundreds of thousands to millions of observations.

2.7 Related Work

ADVI is an automatic variational inference algorithm, implemented within the Stan probabilistic programming system. This draws on two major themes.

Probabilistic programming. The first theme is probabilistic programming. One class of systems focuses on probabilistic models where the user specifies a joint probability distribution. Some examples are BUGS (Spiegelhalter et al., 1995), JAGS (Plummer, 2003), and Stan (Stan Development Team, 2016). Another class of systems allows the user to directly specify probabilistic programs that may not admit a closed form probability distribution. Some examples are Church (Goodman et al., 2008), Figaro (Pfeffer, 2009), Venture (Mansinghka et al., 2014), Anglican (Wood et al., 2014), and WebPPL (Goodman and Stuhlmüller, 2014). Both classes primarily rely on various forms of MCMC sampling for inference.

Variational inference. The second is a body of work that generalizes variational inference. Ranganath et al. (2014) and Salimans and Knowles (2014) propose a black box technique that only requires computing gradients of the variational approximating family. Kingma and Welling (2014) and Rezende et al. (2014) describe a reparameterization of the variational problem that simplifies optimization. Titsias and Lázaro-Gredilla (2014) leverage the gradient of the model for a class of real-valued models. Rezende and Mohamed (2015), Ranganath et al. (2016) and Tran et al. (2016b) improve the accuracy of black box variational approximations. Here we build on and extend these ideas to

automate variational inference; we highlight technical connections as we study the properties of ADVI in Section 3.

Some notable work crosses both themes. Bishop et al. (2002) present an automated variational algorithm for graphical models with conjugate exponential relationships between all parent-child pairs. Winn and Bishop (2005) and Minka et al. (2014) extend this to graphical models with non-conjugate relationships by either using custom approximations or sampling. ADVI automatically supports a more comprehensive class of nonconjugate models; see Section 2.1. Wingate and Weber (2013) study a more general setting, where the variational approximation itself is a probabilistic program.

Automatic differentiation. Automatic differentiation and machine learning enjoy a colorful and intertwined history (Baydin et al., 2015). For example, the backpropagation algorithm, rediscovered independently many times, is a form of automatic differentiation for neural network weights (Widrow and Lehr, 1990). Similarly, researchers have applied automatic differentiation to specific models, such as extended Kalman filters (Meyer et al., 2003) and computer vision models (Pock et al., 2007). Automatic differentiation also appears in recent variational inference research. For instance, the Bayes-by-backprop algorithm is a specific application of automatic differentiation to variational inference in Bayesian neural networks (Blundell et al., 2015). Many of the methods described above could, if applicable, use automatic differentiation to compute gradients of the model and the variational approximating families.

Software. ADVI can also be implemented in other general-purpose software frameworks, such as autograd (Maclaurin et al., 2015), Theano (Theano Development Team, 2016) and TensorFlow (Abadi et al., 2016). These frameworks offer features such as symbolic or automatic differentiation and abstractions for parallel computation. Two other implementations of ADVI are available, at the time of publication. The first is in PyMC3 (Salvatier et al., 2016), a probabilistic programming package, that implements ADVI in Python using Theano. The second is in Edward (Tran et al., 2016a), a Python library for probabilistic modeling, inference, and criticism, that implements ADVI in Python using TensorFlow.

3. Properties of Automatic Differentiation Variational Inference

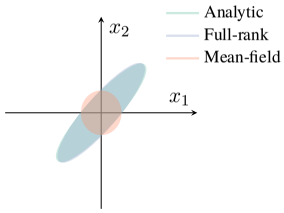
Automatic differentiation variational inference (ADVI) extends classical variational inference techniques in a few directions. In this section, we use simulated data to study three aspects of ADVI: the accuracy of mean-field and full-rank approximations, the variance of the ADVI gradient estimator, and the sensitivity to the transformation T.

3.1 Accuracy

We begin by considering three models that expose how the mean-field approximation affects the accuracy of ADVI.

Two-dimensional Gaussian. We first study a simple model that does not require approximate inference. Consider a multivariate Gaussian likelihood $\text{Normal}(\mathbf{y} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with fixed, yet highly correlated, covariance $\boldsymbol{\Sigma}$; our goal is to estimate the mean $\boldsymbol{\mu}$. If we place a multivariate Gaussian prior on $\boldsymbol{\mu}$ then the posterior is also a Gaussian that we can compute analytically (Bernardo and Smith, 2009).

We draw 1000 datapoints from the model and run both variants of ADVI, mean-field and full-rank, until convergence. Figure 4 compares the ADVI methods to the exact posterior. Both procedures correctly identify the mean of the analytic posterior. However, the shape of the mean-field approximation is incorrect. This is because the mean-field approximation ignores off-diagonal terms of the Gaussian covariance. ADVI minimizes the KL divergence from the approximation to the exact posterior; this leads to a systemic underestimation of marginal variances (Bishop, 2006).



	Analytic	Full-rank	Mean-field
Variance along x_1	0.28	0.28	0.13
Variance along x_2	0.31	0.31	0.14

Figure 4: Comparison of mean-field and full-rank ADVI on a two-dimensional Gaussian model. The figure shows the accuracy of the full-rank approximation. Ellipses correspond to two-sigma level sets of the Gaussian. The table quantifies the underestimation of marginal variances by the mean-field approximation.

Logistic regression. We now study a model for which we need approximate inference. Consider logistic regression, a generalized linear model with a binary response y, covariates $\mathbf{x}$, and likelihood $\text{Bern}(y \mid \text{logit}^{-1}(\mathbf{x}^\top \boldsymbol{\beta}))$; our goal is to estimate the coefficients $\boldsymbol{\beta}$. We place an independent Gaussian prior on each regression coefficient.

We simulated 9 random covariates from the prior distribution (plus a constant intercept) and drew 1000 datapoints from the likelihood. We estimated the posterior of the coefficients with ADVI and Stan's default MCMC technique, the no-U-turn sampler (NUTS) (Hoffman and Gelman, 2014). Figure 5 shows the marginal posterior densities obtained from each approximation. MCMC and ADVI perform similarly in their estimates of the posterior mean. The mean-field approximation, as expected, underestimates marginal posterior variances on most of the coefficients. The full-rank approximation, once again, better matches the posterior.

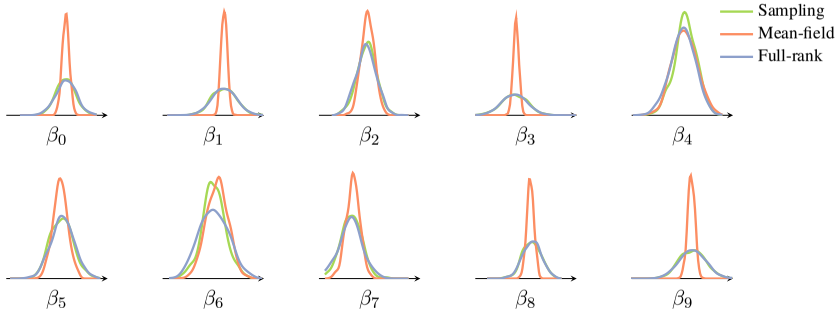


Figure 5: Comparison of marginal posterior densities for a logistic regression model. Each plot shows kernel density estimates for the posterior of each coefficient using 1000 samples. Mean-field ADVI underestimates variances for most of the coefficients.

Stochastic volatility time-series model. Finally, we study a model where the data are not exchangeable. Consider an autoregressive process to model how the latent volatility (i.e., variance) of an economic asset changes over time (Kim et al., 1998); our goal is to estimate the sequence of volatilities. We expect these posterior estimates to be correlated, especially when the volatilities trend away from their mean value.

In detail, the price data exhibit latent volatility as part of the variance of a zero-mean Gaussian

$$y_t \sim \text{Normal}\left(0, \exp(h_t/2)\right)$$

where the log volatility follows an auto-regressive process

$$h_t \sim \text{Normal}\left(\mu + \phi(h_{t-1} - \mu), \sigma\right) \quad \text{with initialization} \quad h_1 \sim \text{Normal}\left(\mu, \frac{\sigma}{\sqrt{1-\phi^2}}\right).$$

We place the following priors on the latent variables

$$\mu \sim \text{Cauchy}(0, 10), \quad \phi \sim \text{Unif}(-1, 1), \quad \text{and} \quad \sigma \sim \text{Lognormal}(0, 10).$$

We set $\mu = -1.025$, $\phi = 0.9$ and $\sigma = 0.6$, and simulate a dataset of 500 time-steps from the generative model above. Figure 6 plots the posterior mean of the log volatility h_t as a function of time. Mean-field ADVI struggles to describe the mean of the posterior, particularly when the log volatility drifts far away from μ. This is expected behavior for a mean-field approximation to a time-series model (Turner and Sahani, 2008). In contrast, full-rank ADVI matches the estimates obtained from sampling.

We further investigate this by studying posterior correlations of the log volatility sequence. We draw $S = 1000$ samples of 500-dimensional log volatility sequences $\{\mathbf{h}^{(s)}\}_1^S$. Figure 7 shows the empirical posterior covariance matrix, $1/S-1 \sum_s (\mathbf{h}^{(s)} - \overline{\mathbf{h}})(\mathbf{h}^{(s)} - \overline{\mathbf{h}})^\top$ for each method. The mean-field covariance (fig. 7a) fails to capture the locally correlated structure of the full-rank and sampling covariance matrices (figs. 7b and 7c). All covariance matrices exhibit a blurry spread due to finite sample size.

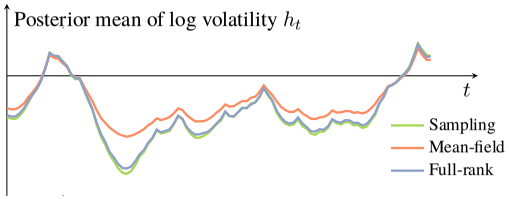
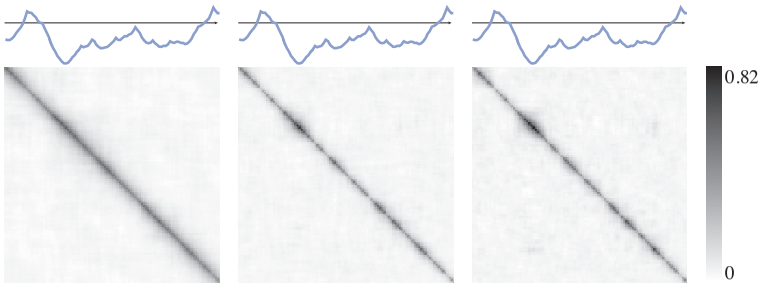


Figure 6: Comparison of posterior mean estimates of volatility h_t. Mean-field ADVI underestimates h_t, especially when it moves far away from its mean μ. Full-rank ADVI matches the accuracy of sampling.

The regions where the local correlation is strongest correspond to the regions where mean-field underestimates the log volatility. To help identify these regions, we overlay the sampling mean log volatility estimate from Figure 6 above each matrix. Both full-rank ADVI and sampling results exhibit correlation where the log volatility trends away from its mean value.



(a) Mean-field **(b)** Full-rank **(c)** Sampling

Figure 7: Comparison of empirical posterior covariance matrices. The mean-field ADVI covariance matrix fails to capture the local correlation structure seen in the full-rank ADVI and sampling results. All covariance matrices exhibit a blurry spread due to finite sample size.

Recommendations. How to choose between full-rank and mean-field ADVI? Scientists interested in posterior variances and covariances should use the full-rank approximation. Full-rank ADVI captures posterior correlations, in turn producing more accurate marginal variance estimates. For large data, however, full-rank ADVI can be prohibitively slow.

Scientists interested in prediction should initially rely on the mean-field approximation. Mean-field ADVI offers a fast algorithm for approximating the posterior mean. In practice, accurate posterior mean estimates dominate predictive accuracy; underestimated marginal variances matters less.

3.2 Variance of the Stochastic Gradients

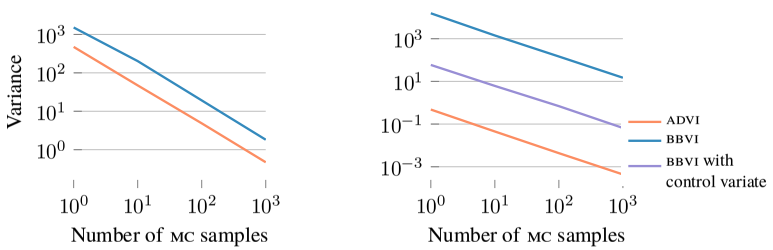
ADVI uses Monte Carlo integration to approximate gradients of the ELBO, and then uses these gradients in a stochastic optimization algorithm (Section 2). The speed of ADVI hinges on the variance of the gradient estimates. When a stochastic optimization algorithm suffers from high-variance gradients, it must repeatedly recover from poor parameter estimates.

ADVI is not the only way to compute Monte Carlo approximations of the gradient of the ELBO. Black box variational inference (BBVI) takes a different approach (Ranganath et al., 2014). The BBVI gradient estimator uses the gradient of the variational approximation and avoids using the gradient of the model. For example, the following BBVI estimator

$$\nabla_{\boldsymbol{\mu}}^{\text{BBVI}} \mathcal{L} = \mathbb{E}_{q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})}\left[\nabla_{\boldsymbol{\mu}} \log q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi}) \left\{\log p\left(\mathbf{x}, T^{-1}(\boldsymbol{\zeta})\right) + \log \left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right| - \log q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})\right\}\right]$$

and the ADVI gradient estimator in Equation (7) both lead to unbiased estimates of the exact gradient. While BBVI is more general—it does not require the gradient of the model and thus applies to more settings—its gradients can suffer from high variance.

Figure 8 empirically compares the variance of both estimators for two models. Figure 8a shows the variance of both gradient estimators for a simple univariate model, where the posterior is a Gamma$(10, 10)$. We estimate the variance using ten thousand re-calculations of the gradient $\nabla_\phi \mathcal{L}$, across an increasing number of MC samples M. The ADVI gradient has lower variance; in practice, a single sample suffices. (See the experiments in Section 4.)



(a) Univariate Model **(b)** Multivariate Nonlinear Regression Model

Figure 8: Comparison of gradient estimator variances. The ADVI gradient estimator exhibits lower variance than the BBVI estimator. Moreover, it does not require control variate variance reduction, which is not available in univariate situations.

Figure 8b shows the same calculation for a 100-dimensional nonlinear regression model with likelihood $\text{Normal}(\mathbf{y} \mid \tanh(\mathbf{x}^\top \boldsymbol{\beta}), \mathbf{I})$ and a Gaussian prior on the regression coefficients $\boldsymbol{\beta}$. Because this is a multivariate example, we also show the BBVI gradient with a variance reduction scheme using control variates described in Ranganath et al. (2014). In both cases, the ADVI gradient is more sample efficient.

3.3 Sensitivity to Transformations

ADVI uses a transformation T from the unconstrained space to the constrained space. We now study how the choice of this transformation affects the non-Gaussian posterior approximation in the original latent variable space.

Consider a posterior density in the Gamma family, with support over $\mathbb{R}_{>0}$. Figure 9 shows three configurations of the Gamma, ranging from $\text{Gamma}(1, 2)$, which places most of its mass close to $\theta = 0$, to $\text{Gamma}(10, 10)$, which is centered at $\theta = 1$. Consider two transformations T_1 and T_2

$$T_1 : \theta \mapsto \log(\theta) \quad \text{and} \quad T_2 : \theta \mapsto \log(\exp(\theta) - 1),$$

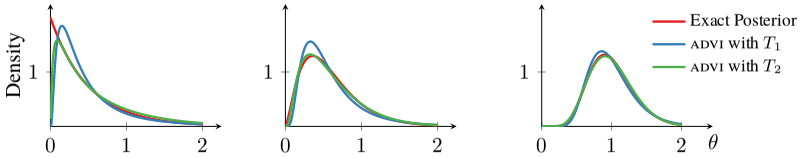
both of which map $\mathbb{R}_{>0}$ to $\mathbb{R}$. ADVI can use either transformation to approximate the Gamma posterior. Which one is better?

Figures 9a to 9c show the ADVI approximation under both transformations. Table 2 reports the corresponding KL divergences. Both graphical and numerical results prefer T_2 over T_1. A quick analysis corroborates this. T_1 is the logarithm, which flattens out for large values. However, T_2 is almost linear for large values of θ. Since both the Gamma (the posterior) and the Gaussian (the ADVI approximation) densities are light-tailed, T_2 is the preferable transformation.

Is there an optimal transformation? Without loss of generality, we consider fixing a standard Gaussian distribution in the real coordinate space.[7] The optimal transformation is then

$$T^* = \Phi^{-1} \circ P(\boldsymbol{\theta} \mid \mathbf{x})$$

7. For two transformations T_1 and T_2 from latent variable space to real coordinate space, there always exists a transformation T_3 within the real coordinate space such that $T_1(\theta) = T_3(T_2(\theta))$.



(a) $\text{Gamma}(1, 2)$ **(b)** $\text{Gamma}(2.5, 4.2)$ **(c)** $\text{Gamma}(10, 10)$

Figure 9: ADVI approximations to Gamma densities under two different transformations.

	$\text{Gamma}(1, 2)$	$\text{Gamma}(2.5, 4.2)$	$\text{Gamma}(10, 10)$
KL $(q \parallel p)$ with T_1	8.1×10^{-2}	3.3×10^{-2}	8.5×10^{-3}
KL $(q \parallel p)$ with T_2	1.6×10^{-2}	3.6×10^{-3}	7.7×10^{-4}

Table 2: KL divergence of ADVI approximations to Gamma densities for two transformations.

where P is the cumulative density function of the posterior and Φ^{-1} is the inverse cumulative density function of the standard Gaussian. P maps the posterior to a uniform distribution and Φ^{-1} maps the uniform distribution to the standard Gaussian. The optimal choice of transformation enables the Gaussian variational approximation to be exact. Sadly, estimating the optimal transformation requires estimating the cumulative density function of the posterior $P(\boldsymbol{\theta} \mid \mathbf{x})$; this is just as hard as the original goal of estimating the posterior density $p(\boldsymbol{\theta} \mid \mathbf{x})$.

This observation motivates pairing transformations with Gaussian variational approximations; there is no need for more complex variational families. ADVI takes the approach of using a library and a model compiler. This is not the only option. For example, Knowles (2015) posits a factorized Gamma density for positively constrained latent variables. In theory, this is equivalent to a mean-field Gaussian density paired with the transformation $T = P_{\text{Gamma}}$, the cumulative density function of the Gamma. (In practice, P_{Gamma} is difficult to compute.) Challis and Barber (2012) study Fourier transform techniques for location-scale variational approximations beyond the Gaussian. Another option is to learn the transformation during optimization. We discuss recent approaches in this direction in Section 5.

4. ADVI in Practice

We now apply ADVI to an array of nonconjugate probability models. With simulated and real data, we study linear regression with automatic relevance determination, hierarchical logistic regression, several variants of non-negative matrix factorization, mixture models, and probabilistic principal component analysis. We compare mean-field ADVI to two MCMC sampling algorithms: Hamiltonian Monte Carlo (HMC) (Neal, 2011) and the no-U-turn sampler (NUTS) (Hoffman and Gelman, 2014). NUTS is an adaptive extension of HMC and the default sampler in Stan.

To place ADVI and MCMC on a common scale, we report predictive likelihood on held-out data as a function of computation time. Specifically, we estimate the predictive likelihood

$$p(\mathbf{x}_{\text{held-out}} \mid \mathbf{x}) = \int p(\mathbf{x}_{\text{held-out}} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \mathbf{x}) \, \mathrm{d}\boldsymbol{\theta}$$

using Monte Carlo estimation. With MCMC, we run the chain and plug in each sample to estimate the integral above; with ADVI, we draw a sample from the variational approximation at every iteration.

We conclude with a case study: an exploratory analysis of over a million taxi rides. Here we show how a scientist might use ADVI in practice.

4.1 Hierarchical Regression Models

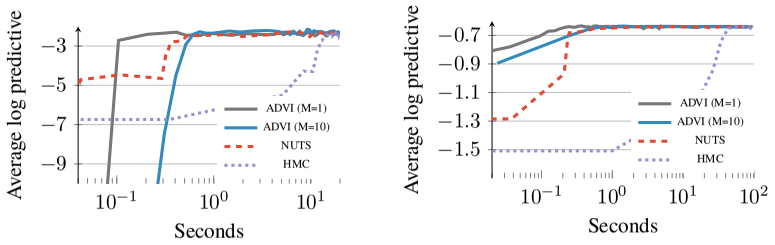
We begin with two nonconjugate regression models: linear regression with automatic relevance determination (ARD) (Bishop, 2006) and hierarchical logistic regression (Gelman and Hill, 2006).

Linear regression with ARD. This is a linear regression model with a hierarchical prior structure that leads to sparse estimates of the coefficients. (Details in Appendix F.1.) We simulate a dataset with 250 regressors such that half of the regressors have no predictive power. We use 10 000 data points for training and withhold 1000 for evaluation.

Logistic regression with a spatial hierarchical prior. This is a hierarchical logistic regression model from political science. The prior captures dependencies, such as states and regions, in a polling dataset from the United States 1988 presidential election (Gelman and Hill, 2006). The model is nonconjugate and would require some form of approximation to derive a classical VI algorithm. (Details in Appendix F.2.)

The dataset includes 145 regressors, with age, education, and state and region indicators. We use 10 000 data points for training and withhold 1536 for evaluation.

Results. Figure 10 plots average log predictive accuracy as a function of time. For these simple models, all methods reach the same predictive accuracy. We study ADVI with two settings of M, the number of MC samples used to estimate gradients. A single sample per iteration is sufficient; it is also the fastest. (We set $M = 1$ from here on.)



(a) Linear regression with ARD

(b) Hierarchical logistic regression

Figure 10: Held-out predictive accuracy results | hierarchical generalized linear models on simulated and real data.

4.2 Non-negative Matrix Factorization

We continue by exploring two nonconjugate non-negative matrix factorization models (Lee and Seung, 1999): a constrained Gamma Poisson model (Canny, 2004) and a Dirichlet Exponential Poisson model. Here, we show how easy it is to explore new models using ADVI. In both models, we use the

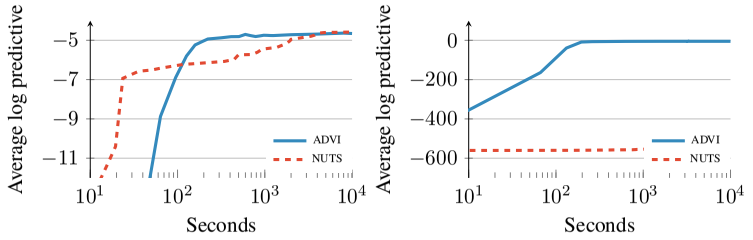
Frey Faces dataset, which contains 1956 frames (28×20 pixels) of facial expressions extracted from a video sequence.

Constrained Gamma Poisson. This is a Gamma Poisson matrix factorization model with an ordering constraint: each row of one of the Gamma factors goes from small to large values. (Details in Appendix F.3.)

Dirichlet Exponential Poisson. This is a nonconjugate Dirichlet Exponential factorization model with a Poisson likelihood. (Details in Appendix F.4.)

Results. Figure 11 shows average log predictive accuracy as well as ten factors recovered from both models. ADVI provides an order of magnitude speed improvement over NUTS (Figure 11a). NUTS struggles with the Dirichlet Exponential model (Figure 11b). In both cases, HMC does not produce any useful samples within a budget of one hour; we omit HMC from here on.

The Gamma Poisson model (Figure 11c) appears to pick significant frames out of the dataset. The Dirichlet Exponential factors (Figure 11d) are sparse and indicate components of the face that move, such as eyebrows, cheeks, and the mouth.



(a) Gamma Poisson predictive likelihood

(b) Dirichlet exponential predictive likelihood



(c) Gamma Poisson factors



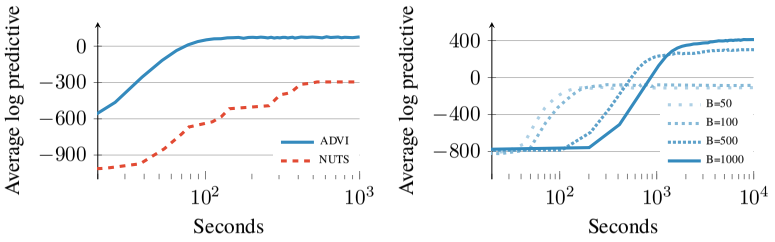
(d) Dirichlet exponential factors

Figure 11: Held-out predictive accuracy results | two non-negative matrix factorization models applied to the Frey Faces dataset.

4.3 Gaussian Mixture Model

This is a nonconjugate Gaussian mixture model (GMM) applied to color image histograms. We place a Dirichlet prior on the mixture proportions, a Gaussian prior on the component means, and a lognormal prior on the standard deviations. (Details in Appendix F.5.) We explore the imageCLEF dataset, which has 250 000 images (Villegas et al., 2013). We withhold 10 000 images for evaluation.

In Figure 12a we randomly select 1000 images and train a model with 10 mixture components. ADVI quickly finds a good solution. NUTS struggles to find an adequate solution and HMC fails altogether



(a) Subset of 1000 images **(b)** Full dataset of 250 000 images

Figure 12: Held-out predictive accuracy results | GMM of the imageCLEF image histogram dataset. **(a)** ADVI outperforms NUTS (Hoffman and Gelman, 2014). **(b)** ADVI scales to large datasets by subsampling minibatches of size B from the dataset at each iteration (Hoffman et al., 2013).

(not shown). This is likely due to label switching, which can affect HMC-based algorithms in mixture models (Stan Development Team, 2016).

Figure 12b shows ADVI results on the full dataset. We increase the number of mixture components to 30. Here we use ADVI, with additional stochastic subsampling of minibatches from the data (Hoffman et al., 2013). With a minibatch size of 500 or larger, ADVI reaches high predictive accuracy. Smaller minibatch sizes lead to suboptimal solutions, an effect also observed in Hoffman et al. (2013). ADVI converges in about two hours; NUTS cannot handle such large datasets.

4.4 A Case Study: Exploring Millions of Taxi Trajectories

How might a scientist use ADVI in practice? How easy is it to develop and revise new models? To answer these questions, we apply ADVI to a modern exploratory data analysis task: analyzing traffic patterns. In this section, we demonstrate how ADVI enables a scientist to quickly develop and revise complex hierarchical models.

The city of Porto has a centralized taxi system of 442 cars. When serving customers, each taxi reports its spatial location at 15 second intervals; this sequence of (x, y) coordinates describes the trajectory and duration of each trip. A dataset of trajectories is publicly available: it contains all 1.7 million taxi rides taken during the year 2014 (European Conference of Machine Learning, 2015).

To gain insight into this dataset, we wish to cluster the trajectories. The first task is to process the raw data. Each trajectory has a different length: shorter trips contain fewer (x, y) coordinates than longer ones. The average trip is approximately 13 minutes long, which corresponds to 50 coordinates. We want to cluster independent of length, so we interpolate all trajectories to 50 coordinate pairs. This converts each trajectory into a point in $\mathbb{R}^{100}$.

The trajectories have structure; for example, major roads and highways appear frequently. This motivates an approach where we first identify a lower-dimensional representation of the data to capture aggregate features, and then we cluster the trajectories in this representation. This is easier than clustering them in the original data space.

We begin with simple dimension reduction: probabilistic principal component analysis (PPCA) (Bishop, 2006). This is a Bayesian generalization of classical principal component analysis, which

is easy to write in Stan. However, like its classical counterpart, PPCA does not identify how many principal components to use for the subspace. To address this, we propose an extension: PPCA with automatic relevance determination (ARD).

PPCA with ARD identifies the latent dimensions that are most effective at explaining variation in the data. The strategy is similar to that in Section 4.1. We assume that there are 100 latent dimensions (i.e., the same dimension as the data) and impose a hierarchical prior that encourages sparsity. Consequently, the model only uses a subset of the latent dimensions to describe the data. (Details in Appendix F.6.)

We randomly subsample ten thousand trajectories and use ADVI to infer a subspace. Figure 13 plots the progression of the ELBO. ADVI converges in approximately an hour and finds an eleven-dimensional subspace. We omit sampling results as both HMC and NUTS struggle with the model; neither produce useful samples within an hour.

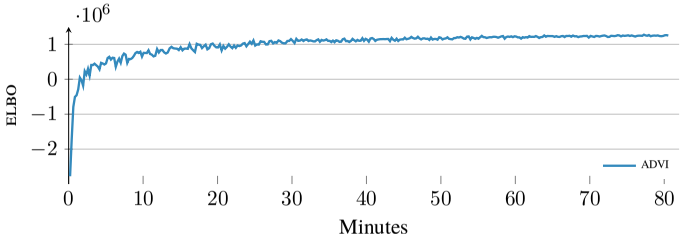


Figure 13: ELBO of PPCA model with ARD. ADVI converges in approximately an hour.

Equipped with this eleven-dimensional subspace, we turn to analyzing the full dataset of 1.7 million taxi trajectories. We first project all trajectories into the subspace. We then use the GMM from Section 4.3 ($K = 30$) components to cluster the trajectories. ADVI takes less than half an hour to converge.

Figure 14 shows a visualization of fifty thousand randomly sampled trajectories. Each color represents the set of trajectories that associate with a particular Gaussian mixture. The clustering is geographical: taxi trajectories that are close to each other are bundled together. The clusters identify frequently taken taxi trajectories.

When we processed the raw data, we interpolated each trajectory to an equal length. This discards all duration information. What if some roads are particularly prone to traffic? Do these roads lead to longer trips?

Supervised probabilistic principal component analysis (SUP-PPCA) is one way to model this. The idea is to regress the durations of each trip onto a subspace that also explains variation in a response variable, in this case, the duration. SUP-PPCA is a simple extension of PPCA (Murphy, 2012). We further extend it using the same ARD prior as before. (Details in Appendix F.7.)

ADVI enables a quick repeat of the above analysis, this time with SUP-PPCA. With ADVI, we find another set of GMM clusters in less than two hours. These clusters, however, are more informative.

Figure 15 shows two clusters that identify particularly busy roads: the bridges of Porto that cross the Duoro river. Figure 15a shows a group of short trajectories that use the two old bridges near the city center. Figure 15b show a group of longer trajectories that use the two newer bridges connecting highways that circumscribe the city.



Figure 14: A visualization of fifty thousand randomly sampled taxi trajectories. The colors represent thirty Gaussian mixtures and the trajectories associated with each.



(a) Trajectories that take the inner bridges. **(b)** Trajectories that take the outer bridges.

Figure 15: Two clusters using SUP-PPCA subspace clustering.

Analyzing these taxi trajectories illustrates how exploratory data analysis is an iterative effort: we want to rapidly evaluate models and modify them based on what we learn. ADVI, which provides automatic and fast inference, enables effective exploration of massive datasets.

5. Discussion

We presented automatic differentiation variational inference (ADVI), a variational inference algorithm that works "out of the box" for a large class of modern probabilistic models. The main idea is to transform the latent variables into a common space. Solving the variational inference problem in this

common space solves it for all models in the class. We studied ADVI using ten different probability models and deployed it as part of Stan, a probabilistic programming system.

There are several avenues for research.

Accuracy. As we showed in Section 3.3, ADVI can be sensitive to the transformations that map the constrained parameter space to the real coordinate space. Dinh et al. (2014) and Rezende and Mohamed (2015) use a cascade of simple transformations to improve accuracy. Tran et al. (2016b) place a Gaussian process to learn the optimal transformation and prove its expressiveness as a universal approximator. Hierarchical variational models (Ranganath et al., 2016) develop rich approximations for non-differentiable latent variable models.

Optimization. ADVI uses first-order automatic differentiation to implement stochastic gradient ascent. Higher-order gradients may enable faster convergence; however computing higher-order gradients comes at a computational cost (Fan et al., 2015). ADVI works with unbiased gradient estimators; introducing some bias to reduce variance could also improve convergence speed (Ruiz et al., 2016a,b). Optimization using line search could also improve convergence robustness (Mahsereci and Hennig, 2015), as well as natural gradient approaches for nonconjugate models (Khan et al., 2015).

Practical heuristics. Two things affect ADVI convergence: initialization and step-size scaling. We initialize ADVI in the real coordinate space as a standard Gaussian. A better heuristic could adapt to the model and dataset based on moment matching. We adaptively tune the scale of the step-size sequence using a finite search. A better heuristic could avoid this additional computation.

Probabilistic programming. We designed and deployed ADVI with Stan in mind. Thus, we focused on the class of differentiable probability models. How can we extend ADVI to discrete latent variables? One approach would be to adapt ADVI to use the black box gradient estimator for these variables (Ranganath et al., 2014). This requires some care as these gradients will exhibit higher variance than the gradients with respect to the differentiable latent variables. (See Section 3.2.) With support for discrete latent variables, modified versions of ADVI could be extended to more general probabilistic programming systems, such as Church (Goodman et al., 2008), Figaro (Pfeffer, 2009), Venture (Mansinghka et al., 2014), Anglican (Wood et al., 2014), WebPPL (Goodman and Stuhlmüller, 2014), and Edward (Tran et al., 2016a).

Before we conclude, we offer some general advice. ADVI, like all of variational inference, is an approximate inference technique. As such, we recommend carefully studying its accuracy for new models. While Section 3.1 indicates a potential shortcoming, the quality of ADVI's posterior approximation will differ from model to model. We recommend validating ADVI for new models by running "fake data" checks (Cook et al., 2006). One of the advantages of ADVI is that its speed of convergence gives more opportunity for such checking in practice.

In summary, ADVI is a first step towards an automated variational inference algorithm that works well for a large class of practical models on modern real-world datasets. Each step of the recipe for ADVI highlights key design decisions: automating transformation of latent variables using a compiler, choosing a variational family that leads to low-variance gradient estimators of the variational objective, and developing an adaptive stochastic optimization step-size sequence that works not only in theory, but also in practice. ADVI enables scientists to easily build, explore, and revise complex probabilistic models with large data.

Acknowledgments

We thank Bruno Jacobs, Bob Carpenter, Daniel Lee, and the reviewers for their helpful comments. This work is supported by NSF IIS-0745520, IIS-1247664, IIS-1009542, SES-1424962, ONR N00014-11-1-0651, N00014-15-1-2541, DARPA FA8750-14-2-0009, N66001-15-C-4032, Sloan G-2015-13987, IES DE R305D140059, NDSEG, Facebook, Adobe, Amazon, and the Siebel Scholar and John Templeton Foundations.

Appendix A. Transformations of Continuous Probability Densities

We present a brief summary of transformations, largely based on (Olive, 2014).

Consider a scalar (univariate) random variable X with probability density function $f_X(x)$. Let $\mathcal{X} = \text{supp}(f_X(x))$ be the support of X. Now consider another random variable Y defined as $Y = T(X)$. Let $\mathcal{Y} = \text{supp}(f_Y(y))$ be the support of Y.

If T is a one-to-one and differentiable function from $\mathcal{X}$ to $\mathcal{Y}$, then Y has probability density function

$$f_Y(y) = f_X\left(T^{-1}(y)\right) \left| \frac{\mathrm{d}T^{-1}(y)}{\mathrm{d}y} \right|.$$

Let us sketch a proof. Consider the cumulative density function Y. If the transformation T is increasing, we directly apply its inverse to the cdf of Y. If the transformation T is decreasing, we apply its inverse to one minus the cdf of Y. The probability density function is the derivative of the cumulative density function. These things combined give the absolute value of the derivative above.

The extension to multivariate variables $\mathbf{X}$ and $\mathbf{Y}$ requires a multivariate version of the absolute value of the derivative of the inverse transformation. This is the absolute determinant of the Jacobian, $|\det J_{T^{-1}}(\mathbf{Y})|$ where the Jacobian is

$$J_{T^{-1}}(\mathbf{Y}) = \begin{pmatrix} \frac{\partial T_1^{-1}}{\partial y_1} & \cdots & \frac{\partial T_1^{-1}}{\partial y_K} \\ \vdots & & \vdots \\ \frac{\partial T_K^{-1}}{\partial y_1} & \cdots & \frac{\partial T_K^{-1}}{\partial y_K} \end{pmatrix}.$$

Intuitively, the Jacobian describes how a transformation warps unit volumes across spaces. This matters for transformations of random variables, since probability density functions must always integrate to one.

Appendix B. Transformation of the Evidence Lower Bound

Recall that $\boldsymbol{\zeta} = T(\boldsymbol{\theta})$ and that the variational approximation in the real coordinate space is $q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})$.

We begin with the ELBO in the original latent variable space. We then transform the latent variable space to the real coordinate space.

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\phi}) &= \int q(\boldsymbol{\theta}) \log \left[\frac{p(\mathbf{x}, \boldsymbol{\theta})}{q(\boldsymbol{\theta})} \right] \mathrm{d}\boldsymbol{\theta} \\
&= \int q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi}) \log \left[\frac{p\left(\mathbf{x}, T^{-1}(\boldsymbol{\zeta})\right) \left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|}{q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})} \right] \mathrm{d}\boldsymbol{\zeta} \\
&= \int q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi}) \log \left[p\left(\mathbf{x}, T^{-1}(\boldsymbol{\zeta})\right) \left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right| \right] \mathrm{d}\boldsymbol{\zeta} - \int q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi}) \log \left[q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi}) \right] \mathrm{d}\boldsymbol{\zeta} \\
&= \mathbb{E}_{q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})} \left[\log p\left(\mathbf{x}, T^{-1}(\boldsymbol{\zeta})\right) + \log \left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right| \right] - \mathbb{E}_{q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})} \left[\log q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi}) \right] \\
&= \mathbb{E}_{q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})} \left[\log p\left(\mathbf{x}, T^{-1}(\boldsymbol{\zeta})\right) + \log \left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right| \right] + \mathbb{H}\big[q(\boldsymbol{\zeta}\,;\,\boldsymbol{\phi})\big].
\end{aligned}$$

Appendix C. Gradients of the Evidence Lower Bound

First, consider the gradient with respect to the $\boldsymbol{\mu}$ parameter. We exchange the order of the gradient and the integration through the dominated convergence theorem (Çınlar, 2011). The rest is the chain

rule for differentiation.

$$\begin{aligned}
\nabla_{\boldsymbol{\mu}}\mathcal{L} &= \nabla_{\boldsymbol{\mu}}\Big\{\mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\log p\left(\mathbf{x}, T^{-1}(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta}))\right) + \log\left|\det J_{T^{-1}}\left(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta})\right)\right|\right] + \mathbb{H}\big[q(\boldsymbol{\zeta};\boldsymbol{\phi})\big]\Big\} \\
&= \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\nabla_{\boldsymbol{\mu}}\left\{\log p\left(\mathbf{x}, T^{-1}(S^{-1}(\boldsymbol{\eta}))\right) + \log\left|\det J_{T^{-1}}\left(S^{-1}(\boldsymbol{\eta})\right)\right|\right\}\right] \\
&= \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\left(\nabla_{\boldsymbol{\theta}}\log p(\mathbf{x},\boldsymbol{\theta})\nabla_{\boldsymbol{\zeta}}T^{-1}(\boldsymbol{\zeta}) + \nabla_{\boldsymbol{\zeta}}\log\left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|\right)\nabla_{\boldsymbol{\mu}}S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta})\right] \\
&= \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\nabla_{\boldsymbol{\theta}}\log p(\mathbf{x},\boldsymbol{\theta})\nabla_{\boldsymbol{\zeta}}T^{-1}(\boldsymbol{\zeta}) + \nabla_{\boldsymbol{\zeta}}\log\left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|\right]
\end{aligned}$$

Then, consider the gradient with respect to the mean-field $\boldsymbol{\omega}$ parameter.

$$\begin{aligned}
\nabla_{\boldsymbol{\omega}}\mathcal{L} &= \nabla_{\boldsymbol{\omega}}\Big\{\mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\log p\left(\mathbf{x}, T^{-1}(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta}))\right) + \log\left|\det J_{T^{-1}}\left(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta})\right)\right|\right] \\
&\qquad + \frac{K}{2}(1+\log(2\pi)) + \sum_{k=1}^{K}\log(\exp(\omega_k))\Big\} \\
&= \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\nabla_{\boldsymbol{\omega}}\left\{\log p\left(\mathbf{x}, T^{-1}(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta}))\right) + \log\left|\det J_{T^{-1}}\left(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta})\right)\right|\right\}\right] + \mathbf{1} \\
&= \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\left(\nabla_{\boldsymbol{\theta}}\log p(\mathbf{x},\boldsymbol{\theta})\nabla_{\boldsymbol{\zeta}}T^{-1}(\boldsymbol{\zeta}) + \nabla_{\boldsymbol{\zeta}}\log\left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|\right)\nabla_{\boldsymbol{\omega}}S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta})\right] + \mathbf{1} \\
&= \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\left(\nabla_{\boldsymbol{\theta}}\log p(\mathbf{x},\boldsymbol{\theta})\nabla_{\boldsymbol{\zeta}}T^{-1}(\boldsymbol{\zeta}) + \nabla_{\boldsymbol{\zeta}}\log\left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|\right)\boldsymbol{\eta}^{\top}\mathrm{diag}(\exp(\boldsymbol{\omega}))\right] + \mathbf{1}.
\end{aligned}$$

Finally, consider the gradient with respect to the full-rank $\mathbf{L}$ parameter.

$$\begin{aligned}
\nabla_{\mathbf{L}}\mathcal{L} &= \nabla_{\mathbf{L}}\Big\{\mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\log p\left(\mathbf{x}, T^{-1}(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta}))\right) + \log\left|\det J_{T^{-1}}\left(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta})\right)\right|\right] \\
&\qquad + \frac{K}{2}(1+\log(2\pi)) + \frac{1}{2}\log\left|\det(\mathbf{L}\mathbf{L}^{\top})\right|\Big\} \\
&= \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\nabla_{\mathbf{L}}\left\{\log p\left(\mathbf{x}, T^{-1}(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta}))\right) + \log\left|\det J_{T^{-1}}\left(S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta})\right)\right|\right\}\right] \\
&\qquad + \nabla_{\mathbf{L}}\frac{1}{2}\log\left|\det(\mathbf{L}\mathbf{L}^{\top})\right| \\
&= \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\left(\nabla_{\boldsymbol{\theta}}\log p(\mathbf{x},\boldsymbol{\theta})\nabla_{\boldsymbol{\zeta}}T^{-1}(\boldsymbol{\zeta}) + \nabla_{\boldsymbol{\zeta}}\log\left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|\right)\nabla_{\mathbf{L}}S_{\boldsymbol{\phi}}^{-1}(\boldsymbol{\eta})\right] + (\mathbf{L}^{-1})^{\top} \\
&= \mathbb{E}_{\mathrm{N}(\boldsymbol{\eta};\mathbf{0},\mathbf{I})}\left[\left(\nabla_{\boldsymbol{\theta}}\log p(\mathbf{x},\boldsymbol{\theta})\nabla_{\boldsymbol{\zeta}}T^{-1}(\boldsymbol{\zeta}) + \nabla_{\boldsymbol{\zeta}}\log\left|\det J_{T^{-1}}(\boldsymbol{\zeta})\right|\right)\boldsymbol{\eta}^{\top}\right] + (\mathbf{L}^{-1})^{\top}
\end{aligned}$$

Appendix D. Automating Expectations: Monte Carlo Integration

Expectations of continuous random variables are integrals. We can use MC integration to approximate them (Robert and Casella, 1999). All we need are samples from q.

$$\mathbb{E}_{q(\boldsymbol{\eta})}\big[f(\boldsymbol{\eta})\big] = \int f(\boldsymbol{\eta})q(\boldsymbol{\eta})\,\mathrm{d}\boldsymbol{\eta} \approx \frac{1}{S}\sum_{s=1}^{S} f(\boldsymbol{\eta}_s) \text{ where } \boldsymbol{\eta}_s \sim q(\boldsymbol{\eta}).$$

MC integration provides noisy, yet unbiased, estimates of the integral. The standard deviation of the estimates decrease as $\mathcal{O}(1/\sqrt{s})$.

Appendix E. Running ADVI in Stan

Visit `http://mc-stan.org/` to download the latest version of Stan. Follow instructions on how to install Stan. You are then ready to use ADVI.

Stan offers multiple interfaces. We describe the command line interface (`cmdStan`) below.

```
./myModel  variational
           grad_samples=M              ( M = 1 default )
           data file=myData.data.R
           output file=output_advi.csv
           diagnostic_file=elbo_advi.csv
```

Figure 16: Syntax for using ADVI via `cmdStan`.

Here, `myData.data.R` is the dataset stored in the R language `Rdump` format. `output_advi.csv` contains samples from the posterior and `elbo_advi.csv` reports the ELBO.

Appendix F. Details of Studied Models

F.1 Linear Regression with Automatic Relevance Determination

Linear regression with ARD is a high-dimensional sparse regression model (Bishop, 2006; Drugowitsch, 2013). This sort of regression model is sometimes referred to as an *hierarchical* or *multilevel* regression model. We describe the model below. The Stan program is in Figure 17.

The inputs are $\mathbf{x} = x_{1:N}$ where each x_n is D-dimensional. The outputs are $\mathbf{y} = y_{1:N}$ where each y_n is 1-dimensional. The weights vector $\mathbf{w}$ is D-dimensional. The likelihood

$$p(\mathbf{y} \mid \mathbf{x}, \mathbf{w}, \sigma) = \prod_{n=1}^{N} \mathrm{Normal}\left(y_n \mid \mathbf{w}^{\top}\mathbf{x}_n\,,\ \sigma\right)$$

describes measurements corrupted by iid Gaussian noise with unknown standard deviation σ.

The ARD prior and hyper-prior structure is as follows

$$\begin{aligned}
p(\mathbf{w}, \sigma, \boldsymbol{\alpha}) &= p(\mathbf{w}, \sigma \mid \boldsymbol{\alpha})p(\boldsymbol{\alpha}) \\
&= \mathrm{Normal}\left(\mathbf{w} \mid 0\,,\ \sigma\left(\mathrm{diag}\sqrt{\boldsymbol{\alpha}}\right)^{-1}\right)\mathrm{InvGamma}(\sigma \mid a_0, b_0)\prod_{i=1}^{D}\mathrm{Gamma}(\alpha_i \mid c_0, d_0)
\end{aligned}$$

where $\boldsymbol{\alpha}$ is a D-dimensional hyper-prior on the weights, where each component gets its own independent Gamma prior.

We simulate data such that only half the regressors have predictive power. The results in Figure 10a use $a_0 = b_0 = c_0 = d_0 = 1$ as hyper-parameters for the Gamma priors.

F.2 Hierarchical Logistic Regression

Hierarchical logistic regression is an intuitive way to model structured classification problems. We study a model of voting preferences, republican or democrat, from the 1988 United States presidential election. Chapter 14.1 of (Gelman and Hill, 2006) motivates the model and explains the dataset in

detail. We also briefly describe the model below. The Stan program is in Figure 18, based on (Stan Development Team, 2016).

$$\Pr(y_n = 1) = \text{sigmoid}\Big(\beta^0 + \beta^{\text{female}} \cdot \text{female}_n + \beta^{\text{black}} \cdot \text{black}_n + \beta^{\text{female.black}} \cdot \text{female.black}_n + \alpha^{\text{age}}_{k[n]} + \alpha^{\text{edu}}_{l[n]} + \alpha^{\text{age.edu}}_{k[n],l[n]} + \alpha^{\text{state}}_{j[n]}\Big)$$

$$\alpha^{\text{state}}_j \sim \text{Normal}\left(\alpha^{\text{region}}_{m[j]} + \beta^{\text{v.prev}} \cdot \text{v.prev}_j\,,\ \sigma_{\text{state}}\right).$$

The hierarchical variables are

$$\begin{aligned}
\alpha^{\text{age}}_k &\sim \text{Normal}\left(0\,,\ \sigma_{\text{age}}\right) \text{ for } k = 1, \ldots, K \\
\alpha^{\text{edu}}_l &\sim \text{Normal}\left(0\,,\ \sigma_{\text{edu}}\right) \text{ for } l = 1, \ldots, L \\
\alpha^{\text{age.edu}}_{k,l} &\sim \text{Normal}\left(0\,,\ \sigma_{\text{age.edu}}\right) \text{ for } k = 1, \ldots, K, l = 1, \ldots, L \\
\alpha^{\text{region}}_m &\sim \text{Normal}\left(0\,,\ \sigma_{\text{region}}\right) \text{ for } m = 1, \ldots, M.
\end{aligned}$$

The regression coefficient β has a Normal(0, 10) prior and all standard deviation latent variables have half Normal(0, 10) priors.

F.3 Non-negative Matrix Factorization: Constrained Gamma Poisson Model

The Gamma Poisson factorization model describes discrete data matrices (Canny, 2004; Cemgil, 2009).

Consider a $U \times I$ matrix of observations. We find it helpful to think of $u = \{1, \cdots, U\}$ as users and $i = \{1, \cdots, I\}$ as items, as in a recommendation system setting. The generative process for a Gamma Poisson model with K factors is

1. For each user u in $\{1, \cdots, U\}$:
   - For each component k, draw $\theta_{uk} \sim \text{Gamma}(a_0, b_0)$.
2. For each item i in $\{1, \cdots, I\}$:
   - For each component k, draw $\beta_{ik} \sim \text{Gamma}(c_0, d_0)$.
3. For each user and item:
   - Draw the observation $y_{ui} \sim \text{Poisson}(\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$.

A potential downfall of this model is that it is not uniquely identifiable: swapping rows and columns of $\boldsymbol{\theta}$ and $\boldsymbol{\beta}$ give the same inner product. One way to contend with this is to constrain either vector to be an ordered vector during inference. We constrain each $\boldsymbol{\theta}_u$ vector in our model in this fashion. The Stan program is in Figure 19. We set $K = 10$ and all the Gamma hyper-parameters to 1 in our experiments.

F.4 Non-negative Matrix Factorization: Dirichlet Exponential Poisson Model

Another model for discrete data is a Dirichlet Exponential model. The Dirichlet enforces uniqueness while the exponential promotes sparsity. This is a non-conjugate model that does not appear to have been studied before.

The generative process for a Dirichlet Exponential model with K factors is

1. For each user u in $\{1, \cdots, U\}$:
   - Draw the K-vector $\boldsymbol{\theta}_u \sim \text{Dirichlet}(\boldsymbol{\alpha}_0)$.
2. For each item i in $\{1, \cdots, I\}$:
   - For each component k, draw $\beta_{ik} \sim \text{Exponential}(\lambda_0)$.
3. For each user and item:
   - Draw the observation $y_{ui} \sim \text{Poisson}(\boldsymbol{\theta}_u^\top \boldsymbol{\beta}_i)$.

The Stan program is in Figure 20. We set $K = 10$, $\alpha_0 = 1000$ for each component, and $\lambda_0 = 0.1$. With this configuration of hyper-parameters, the factors $\boldsymbol{\beta}_i$ appear sparse.

F.5 Gaussian Mixture Model

The Gaussian mixture model (GMM) is a versatile probability model (Bishop, 2006), often used for density estimation and clustering. Here we use it to group a dataset of natural images based on their color histograms. We build a high-dimensional GMM with a Gaussian prior for the mixture means, a lognormal prior for the mixture standard deviations, and a Dirichlet prior for the mixture components.

Represent the images as $\mathbf{y} = y_{1:N}$ where each y_n is D-dimensional and there are N observations. The likelihood for the images is

$$p(\mathbf{y} \mid \boldsymbol{\theta}, \boldsymbol{\mu}, \boldsymbol{\sigma}) = \prod_{n=1}^{N} \sum_{k=1}^{K} \theta_k \prod_{d=1}^{D} \text{Normal}(y_{nd} \mid \mu_{kd}, \sigma_{kd})$$

with a Dirichlet prior for the mixture proportions

$$p(\boldsymbol{\theta}) = \text{Dirichlet}(\boldsymbol{\theta}\,;\, \boldsymbol{\alpha}_0),$$

a Gaussian prior for the mixture means

$$p(\boldsymbol{\mu}) = \prod_{k=1}^{D} \prod_{d=1}^{D} \text{Normal}(\mu_{kd}\,;\, 0, 1)$$

and a lognormal prior for the mixture standard deviations

$$p(\boldsymbol{\sigma}) = \prod_{k=1}^{D} \prod_{d=1}^{D} \text{Lognormal}(\sigma_{kd}\,;\, 0, 1).$$

The dimension of the color histograms in the imageCLEF dataset is $D = 576$. This is a concatenation of three 192-length histograms, one for each color channel (red, green, blue) of the images.

We scale the image histograms to have zero mean and unit variance. Setting α_0 to a small value encourages the model to use fewer components to explain the data. Larger values of α_0 encourage the model to use all K components. We set $\alpha_0 = 1\,000$ in our experiments.

The Stan program is in Figure 21. The stochastic data subsampling version of the code is in Figure 22.

F.6 Probabilistic Principal Component Analysis with Automatic Relevance Determination

Probabilistic principal component analysis (PPCA) is a Bayesian extension of classical principal component analysis (Bishop, 2006). Consider a dataset of $\mathbf{x} = x_{1:N}$ where each x_n is D-dimensional. Let $M < D$ be the dimension of the subspace we will use for analysis.

First define a set of latent variables $\mathbf{z} = z_{1:N}$ where each z_n is M-dimensional. Draw each z_n from a standard normal

$$p(\mathbf{z}) = \prod_{n=1}^{N} \text{Normal}(z_n \,;\, \mathbf{0}, \mathbf{I}).$$

Then define a set of principal components $\mathbf{w} = w_{1:D}$ where each w_d is M-dimensional. Similarly, draw the principal components from a standard normal

$$p(\mathbf{w}) = \prod_{d=1}^{D} \text{Normal}(w_d \,;\, \mathbf{0}, \mathbf{I}).$$

Finally define the likelihood through an inner product as

$$p(\mathbf{x} \mid \mathbf{w}, \mathbf{z}, \sigma) = \prod_{n=1}^{N} \text{Normal}(x_n \,;\, \mathbf{w}^\top z_n, \sigma\mathbf{I}).$$

The standard deviation σ is also a latent variable. Place a lognormal prior on it as

$$p(\sigma) = \text{Lognormal}(\sigma \,;\, 0, 1).$$

We extend PPCA by adding an ARD hierarchical prior. The extended model introduces a M-dimensional vector $\boldsymbol{\alpha}$ which chooses which principal components to retain. ($M < D$ now represents the maximum number of principal components to consider.) This extends the above by

$$\begin{aligned}
p(\boldsymbol{\alpha}) &= \prod_{m=1}^{M} \text{InvGamma}(\alpha_m \,;\, 1, 1) \\
p(\mathbf{w} \mid \boldsymbol{\alpha}) &= \prod_{d=1}^{D} \text{Normal}(w_d \,;\, \mathbf{0}, \sigma\text{diag}(\boldsymbol{\alpha})) \\
p(\mathbf{x} \mid \mathbf{w}, \mathbf{z}, \sigma) &= \prod_{n=1}^{N} \text{Normal}(x_n \,;\, \mathbf{w}z_n, \sigma\mathbf{I}).
\end{aligned}$$

The Stan program that implements PPCA is in Figure 23.

F.7 Supervised Probabilistic Principal Component Analysis with Automatic Relevance Determination

Supervised probabilistic principal component analysis (SUP-PPCA) augments PPCA by regressing a vector of observed random variables y onto the principal component subspace. The idea is to not only find a set of principal components that describe variation in the dataset $\mathbf{x}$, but to also predict y. The complete model is

$$\begin{aligned}
p(\mathbf{z}) &= \prod_{n=1}^{N} \text{Normal}(z_n \,;\, \mathbf{0}, \mathbf{I}) \\
p(\sigma) &= \text{Lognormal}(\sigma \,;\, 0, 1) \\
p(\boldsymbol{\alpha}) &= \prod_{m=1}^{M} \text{InvGamma}(\alpha_m \,;\, 1, 1) \\
p(\mathbf{w}_\mathbf{x} \mid \boldsymbol{\alpha}) &= \prod_{d=1}^{D} \text{Normal}(w_d \,;\, \mathbf{0}, \sigma\text{diag}(\boldsymbol{\alpha})) \\
p(w_y \mid \boldsymbol{\alpha}) &= \text{Normal}(w_y \,;\, \mathbf{0}, \sigma\text{diag}(\boldsymbol{\alpha})) \\
p(\mathbf{x} \mid \mathbf{w}_\mathbf{x}, \mathbf{z}, \sigma) &= \prod_{n=1}^{N} \text{Normal}(x_n \,;\, \mathbf{w}_\mathbf{x} z_n, \sigma\mathbf{I}) \\
p(y \mid w_y, \mathbf{z}, \sigma) &= \prod_{n=1}^{N} \text{Normal}(y_n \,;\, w_y z_n, \sigma).
\end{aligned}$$

The Stan program that implements SUP-PPCA is in Figure 24.

```
data {
  int<lower=0> N;   // number of data items
  int<lower=0> D;   // dimension of input features
  matrix[N, D] x;   // input matrix
  vector[N]    y;   // output vector

  // hyperparameters for Gamma priors
  real<lower=0> a0;
  real<lower=0> b0;
  real<lower=0> c0;
  real<lower=0> d0;
}

parameters {
  vector[D] w;              // weights (coefficients) vector
  real<lower=0> sigma;      // standard deviation
  vector<lower=0>[D] alpha; // hierarchical latent variables
}

transformed parameters {
  vector[D] one_over_sqrt_alpha;
  for (d in 1:D)
    one_over_sqrt_alpha[d] = 1 / sqrt(alpha[d]);
}

model {
  // alpha: hyper-prior on weights
  alpha ~ gamma(c0, d0);

  // sigma: prior on standard deviation
  sigma ~ inv_gamma(a0, b0);

  // w: prior on weights
  w ~ normal(0, sigma * one_over_sqrt_alpha);

  // y: likelihood
  y ~ normal(x * w, sigma);
}
```

Figure 17: Stan program for Linear Regression with Automatic Relevance Determination.


```
data {
  int<lower=0> N;
  int<lower=0> n_age;
  int<lower=0> n_age_edu;
  int<lower=0> n_edu;
  int<lower=0> n_region_full;
  int<lower=0> n_state;
  int<lower=0, upper=n_age> age[N];
  int<lower=0, upper=n_age_edu> age_edu[N];
  vector<lower=0, upper=1>[N] black;
  int<lower=0, upper=n_edu> edu[N];
  vector<lower=0, upper=1>[N] female;
  int<lower=0, upper=n_region_full> region_full[N];
  int<lower=0, upper=n_state> state[N];
  vector[N] v_prev_full;
  int<lower=0, upper=1> y[N];
}
parameters {
  vector[n_age] a;
  vector[n_edu] b;
  vector[n_age_edu] c;
  vector[n_state] d;
  vector[n_region_full] e;
  vector[5] beta;
  real<lower=0> sigma_a;
  real<lower=0> sigma_b;
  real<lower=0> sigma_c;
  real<lower=0> sigma_d;
  real<lower=0> sigma_e;
}
transformed parameters {
  vector[N] y_hat;
  for (i in 1:N)
    y_hat[i] = beta[1]
               + beta[2] * black[i]
               + beta[3] * female[i]
               + beta[5] * female[i] * black[i]
               + beta[4] * v_prev_full[i]
               + a[age[i]]
               + b[edu[i]]
               + c[age_edu[i]]
               + d[state[i]]
               + e[region_full[i]];
}
model {
  a ~ normal (0, sigma_a);
  b ~ normal (0, sigma_b);
  c ~ normal (0, sigma_c);
  d ~ normal (0, sigma_d);
  e ~ normal (0, sigma_e);
  beta ~ normal(0, 10);
  sigma_a ~ normal(0, 10);
  sigma_b ~ normal(0, 10);
  sigma_c ~ normal(0, 10);
  sigma_d ~ normal(0, 10);
  sigma_e ~ normal(0, 10);
  y ~ bernoulli_logit(y_hat);
}
```

Figure 18: Stan program for Hierarchical Logistic Regression, adapted from (Stan Development Team, 2016).

```
data {
  int<lower=0> U;
  int<lower=0> I;
  int<lower=0> K;
  int<lower=0> y[U, I];
  real<lower=0> a;
  real<lower=0> b;
  real<lower=0> c;
  real<lower=0> d;
}

parameters {
  positive_ordered[K] theta[U]; // user preference
  vector<lower=0>[K] beta[I];   // item attributes
}

model {
  for (u in 1:U)
    theta[u] ~ gamma(a, b); // componentwise gamma
  for (i in 1:I)
    beta[i] ~ gamma(c, d);  // componentwise gamma

  for (u in 1:U) {
    for (i in 1:I) {
      y[u, i] ~ poisson(theta[u]' * beta[i]);
    }
  }
}
```

Figure 19: Stan program for the Gamma Poisson non-negative matrix factorization model.


```
data {
  int<lower=0> U;
  int<lower=0> I;
  int<lower=0> K;
  int<lower=0> y[U, I];
  real<lower=0> lambda0;
  real<lower=0> alpha0;
}

transformed data {
  vector<lower=0>[K] alpha0_vec;
  for (k in 1:K)
    alpha0_vec[k] = alpha0;
}

parameters {
  simplex[K] theta[U];          // user preference
  vector<lower=0>[K] beta[I];   // item attributes
}

model {
  for (u in 1:U)
    theta[u] ~ dirichlet(alpha0_vec); // componentwise dirichlet
  for (i in 1:I)
    beta[i] ~ exponential(lambda0);   // componentwise exponential

  for (u in 1:U) {
    for (i in 1:I) {
      y[u, i] ~ poisson(theta[u]' * beta[i]);
    }
  }
}
```

Figure 20: Stan program for the Dirichlet Exponential non-negative matrix factorization model.

```
data {
  int<lower=0> N;       // number of data points in entire dataset
  int<lower=0> K;       // number of mixture components
  int<lower=0> D;       // dimension
  vector[D] y[N];       // observations

  real<lower=0> alpha0; // dirichlet prior
}

transformed data {
  vector<lower=0>[K] alpha0_vec;
  for (k in 1:K)
    alpha0_vec[k] = alpha0;
}

parameters {
  simplex[K] theta;             // mixing proportions
  vector[D] mu[K];              // locations of mixture components
  vector<lower=0>[D] sigma[K];  // standard deviations of mixture components
}

model {
  // priors
  theta ~ dirichlet(alpha0_vec);
  for (k in 1:K) {
      mu[k] ~ normal(0, 1);
      sigma[k] ~ lognormal(0, 1);
  }

  // likelihood
  for (n in 1:N) {
    real ps[K];
    for (k in 1:K) {
      ps[k] = log(theta[k]) + normal_lpdf(y[n] | mu[k], sigma[k]);
    }
    target += log_sum_exp(ps);
  }
}
```

Figure 21: Stan program for the GMM example.


```
functions {
  real divide_promote_real(int x, int y) {
    real x_real;
    x_real = x;
    return x_real / y;
  }
}

data {
  int<lower=0> NFULL;     // total number of datapoints in dataset
  int<lower=0> N;         // number of data points in minibatch

  int<lower=0> K;         // number of mixture components
  int<lower=0> D;         // dimension

  vector[D] yFULL[NFULL]; // dataset
  vector[D] y[N];         // minibatch

  real<lower=0> alpha0;   // dirichlet hyper-prior parameter
}

transformed data {
  real minibatch_factor;
  vector<lower=0>[K] alpha0_vec;
  for (k in 1:K)
    alpha0_vec[k] = alpha0 / K;
  minibatch_factor = divide_promote_real(N, NFULL);
}

parameters {
  simplex[K] theta;             // mixing proportions
  vector[D] mu[K];              // locations of mixture components
  vector<lower=0>[D] sigma[K];  // standard deviations of mixture components
}

model {
  // priors
  theta ~ dirichlet(alpha0_vec);
  for (k in 1:K) {
      mu[k] ~ normal(0, 1);
      sigma[k] ~ lognormal(0, 1);
  }

  // likelihood
  for (n in 1:N) {
    real ps[K];
    for (k in 1:K) {
      ps[k] = log(theta[k]) + normal_lpdf(y[n] | mu[k], sigma[k]);
    }
    target += minibatch_factor * log_sum_exp(ps);
  }
}
```

Figure 22: Stan program for the GMM example, with stochastic subsampling of the dataset.

```
data {
  int<lower=0> N;            // number of data points in dataset
  int<lower=0> D;            // dimension
  int<lower=0> M;            // maximum dimension of latent space to consider

  vector[D] x[N];            // data
}

parameters {
  matrix[M, N] z;            // latent variable
  matrix[D, M] w;            // weights parameters
  real<lower=0> sigma;       // standard deviation parameter
  vector<lower=0>[M] alpha;  // hyper-parameters on weights
}

model {
  // priors
  to_vector(z) ~ normal(0, 1);
  for (d in 1:D)
    w[d] ~ normal(0, sigma * alpha);
  sigma ~ lognormal(0, 1);
  alpha ~ inv_gamma(1, 1);

  // likelihood
  for (n in 1:N)
    x[n] ~ normal(w * col(z, n), sigma);
}
```

Figure 23: Stan program for PPCA with ARD.


```
data {
  int<lower=0> N;            // number of data points in dataset
  int<lower=0> D;            // dimension
  int<lower=0> M;            // maximum dimension of latent space to consider

  vector[D] x[N];            // data
  vector[N] y;               // data
}

parameters {
  matrix[M, N] z;            // latent variable
  matrix[D, M] w_x;          // weight parameters for x
  vector[M] w_y;             // weight parameters for y
  real<lower=0> sigma;       // standard deviation parameter
  vector<lower=0>[M] alpha;  // hyper-parameters on weights
}

model {
  // priors
  to_vector(z) ~ normal(0, 1);
  for (d in 1:D)
    w_x[d] ~ normal(0, sigma * alpha);
  w_y ~ normal(0, sigma * alpha);

  sigma ~ lognormal(0, 1);
  alpha ~ inv_gamma(1, 1);

  // likelihood
  for (n in 1:N) {
    x[n] ~ normal(w_x * col(z, n), sigma);
    y[n] ~ normal(w_y' * col(z, n), sigma);
  }
}
```

Figure 24: Stan program for SUP-PPCA with ARD.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek Gordon Murray, Benoit Steiner, Paul A. Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zhang. Tensorflow: a system for large-scale machine learning. *arXiv:1605.08695*, 2016.

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

Charles E Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2(6):1152–1174, 1974.

David Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

Atilim Gunes Baydin, Barak A Pearlmutter, and Alexey Andreyevich Radul. Automatic differentiation in machine learning: a survey. *arXiv:1502.05767*, 2015.

José M Bernardo and Adrian FM Smith. *Bayesian Theory*. John Wiley & Sons, 2009.

Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer New York, 2006.

Christopher M Bishop, David Spiegelhalter, and John Winn. VIBES: a variational inference engine for Bayesian networks. In *Neural Information Processing Systems*, 2002.

David M Blei. Build, compute, critique, repeat: data analysis with latent variable models. *Annual Review of Statistics and Its Application*, 1:203–232, 2014.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: a review for statisticians. *arXiv:1601.00670*, 2016.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 1613–1622, 2015.

Léon Bottou. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, pages 421–436. Springer, 2012.

John Canny. GaP: a factor model for discrete data. In *SIGIR Conference on Research and Development in Information Retrieval*, 2004.

Bob Carpenter, Matthew D Hoffman, Marcus Brubaker, Daniel Lee, Peter Li, and Michael Betancourt. The Stan math library: reverse-mode automatic differentiation in C++. *arXiv:1509.07164*, 2015.

Ali Taylan Cemgil. Bayesian inference for nonnegative matrix factorisation models. *Computational Intelligence and Neuroscience*, 2009.

Edward Challis and David Barber. Affine independent variational inference. In *Neural Information Processing Systems*, 2012.

Edward Challis and David Barber. Gaussian Kullback-Leibler approximate inference. *The Journal of Machine Learning Research*, 14(1):2239–2286, 2013.

Erhan Çınlar. *Probability and Stochastics*. Springer, 2011.

Samantha R Cook, Andrew Gelman, and Donald B Rubin. Validation of software for Bayesian models using posterior quantiles. *Journal of Computational and Graphical Statistics*, pages 675–692, 2006.

Persi Diaconis and Donald Ylvisaker. Conjugate priors for exponential families. *The Annals of Statistics*, 7(2):269–281, 1979.

Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: non-linear independent components estimation. *arXiv:1410.8516*, 2014.

Jan Drugowitsch. Variational Bayesian inference for linear and logistic regression. *arXiv:1310.5438*, 2013.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *The Journal of Machine Learning Research*, 12:2121–2159, 2011.

Kai Fan, Ziteng Wang, Jeff Beck, James Kwok, and Katherine A Heller. Fast second order stochastic backpropagation for variational inference. In *Neural Information Processing Systems*, 2015.

Andrew Gelman and Jennifer Hill. *Data Analysis using Regression and Multilevel/Hierarchical Models*. Cambridge University Press, 2006.

Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian Data Analysis*. CRC Press, 2013.

Noah D Goodman and Andreas Stuhlmüller. The Design and Implementation of Probabilistic Programming Languages, 2014.

Noah D Goodman, Vikash K Mansinghka, Daniel Roy, Keith Bonawitz, and Joshua B Tenenbaum. Church: a language for generative models. In *Uncertainty in Artificial Intelligence*, 2008.

Wolfgang Härdle and Léopold Simar. *Applied Multivariate Statistical Analysis*. Springer, 2012.

Matthew D Hoffman and Andrew Gelman. The No-U-Turn sampler. *The Journal of Machine Learning Research*, 15(1):1593–1623, 2014.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999.

Mohammad E Khan, Pierre Baqué, François Fleuret, and Pascal Fua. Kullback-Leibler proximal variational inference. In *Neural Information Processing Systems*, 2015.

Sangjoon Kim, Neil Shephard, and Siddhartha Chib. Stochastic volatility: likelihood inference and comparison with ARCH models. *The Review of Economic Studies*, 65(3):361–393, 1998.

Diederik Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

Diederik P Kingma and Jimmy Ba Adam. A method for stochastic optimization. In *International Conference on Learning Representation*, 2015.

David A Knowles. Stochastic gradient variational Bayes for Gamma approximating distributions. *arXiv:1509.01631*, 2015.

Alp Kucukelbir, Rajesh Ranganath, Andrew Gelman, and David Blei. Automatic variational inference in Stan. In *Neural Information Processing Systems*, 2015.

D. Lee and H. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401 (6755):788–791, 1999.

Dougal Maclaurin, David Duvenaud, and Ryan P Adams. Autograd: reverse-mode differentiation of native Python. In *ICML Workshop on Automatic Machine Learning*, 2015.

Maren Mahsereci and Philipp Hennig. Probabilistic line searches for stochastic optimization. In *Neural Information Processing Systems*, 2015.

Vikash Mansinghka, Daniel Selsam, and Yura Perov. Venture: a higher-order probabilistic programming platform with programmable inference. *arXiv:1404.0099*, 2014.

Renate Meyer, David A Fournier, and Andreas Berg. Stochastic volatility: Bayesian computation using automatic differentiation and the extended Kalman filter. *The Econometrics Journal*, 6(2): 408–420, 2003.

T. Minka, J.M. Winn, J.P. Guiver, S. Webster, Y. Zaykov, B. Yangel, A. Spengler, and J. Bronskill. Infer.NET 2.6, 2014. Microsoft Research Cambridge. http://research.microsoft.com/infernet.

Kevin P Murphy. *Machine Learning: a Probabilistic Perspective*. MIT Press, 2012.

Radford M Neal. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, 2: 113–162, 2011.

David J Olive. *Statistical Theory and Inference*. Springer, 2014.

Manfred Opper and Cédric Archambeau. The variational Gaussian approximation revisited. *Neural Computation*, 21(3):786–792, 2009.

Avi Pfeffer. Figaro: an object-oriented probabilistic programming language. *Charles River Analytics Technical Report*, 137, 2009.

José C Pinheiro and Douglas M Bates. Unconstrained parametrizations for variance-covariance matrices. *Statistics and Computing*, 6(3):289–296, 1996.

Martyn Plummer. JAGS: a program for analysis of Bayesian graphical models using Gibbs sampling. In *International Workshop on Distributed Statistical Computing*, 2003.

Thomas Pock, Michael Pock, and Horst Bischof. Algorithmic differentiation: application to variational problems in computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(7):1180–1193, 2007.

Rajesh Ranganath, Chong Wang, Blei David, and Eric Xing. An adaptive learning rate for stochastic variational inference. In *International Conference on Machine Learning*, 2013.

Rajesh Ranganath, Sean Gerrish, and David M Blei. Black box variational inference. In *Artificial Intelligence and Statistics*, 2014.

Rajesh Ranganath, Dustin Tran, and David M Blei. Hierarchical variational models. In *International Conference on Machine Learning*, 2016.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International Conference on Machine Learning*, 2015.

Danilo J Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 1951.

Christian P Robert and George Casella. *Monte Carlo Statistical Methods*. Springer, 1999.

Francisco JR Ruiz, Michalis K Titsias, and David M Blei. The generalized reparameterization gradient. In *Neural Information Processing Systems*, 2016a.

Francisco JR Ruiz, Michalis K Titsias, and David M Blei. Overdispersed black-box variational inference. In *Uncertainty in Artificial Intelligence*, 2016b.

Tim Salimans and David Knowles. On using control variates with stochastic approximation for variational Bayes. *arXiv:1401.1022*, 2014.

John Salvatier, Thomas V Wiecki, and Christopher Fonnesbeck. Probabilistic programming in Python using PyMC3. *PeerJ Computer Science*, 2:e55, 2016.

Matthias Seeger. Gaussian covariance and scalable variational inference. In *International Conference on Machine Learning*, 2010.

David J Spiegelhalter, Andrew Thomas, Nicky G Best, and Wally R Gilks. BUGS: Bayesian inference using Gibbs sampling, version 0.50. *MRC Biostatistics Unit, Cambridge*, 1995.

Stan Development Team. *Stan Modeling Language Users Guide and Reference Manual*, 2016.

Theano Development Team. Theano: a Python framework for fast computation of mathematical expressions. *arXiv:1605.02688*, 2016.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4, 2012.

Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational Bayes for non-conjugate inference. In *International Conference on Machine Learning*, 2014.

Dustin Tran, Alp Kucukelbir, Adji B Dieng, Maja Rudolph, Dawen Liang, and David M Blei. Edward: a library for probabilistic modeling, inference, and criticism. *arXiv:1610.09787*, 2016a.

Dustin Tran, Rajesh Ranganath, and David M Blei. The variational Gaussian process. In *International Conference on Learning Representations*, 2016b.

Richard E Turner and Maneesh Sahani. Two problems with variational expectation maximisation for time-series models. In *Workshop on Inference and Estimation in Probabilistic Time-Series Models*, 2008.

Mauricio Villegas, Roberto Paredes, and Bart Thomee. Overview of the ImageCLEF 2013 Scalable Concept Image Annotation Subtask. In *CLEF Evaluation Labs and Workshop*, 2013.

Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

Bernard Widrow and Michael A Lehr. 30 years of adaptive neural networks: perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, 78(9):1415–1442, 1990.

David Wingate and Theophane Weber. Automated variational inference in probabilistic programming. *arXiv:1301.1299*, 2013.

John Winn and Christopher M Bishop. Variational message passing. *Journal of Machine Learning Research*, 6:661–694, 2005.

Frank Wood, Jan Willem van de Meent, and Vikash Mansinghka. A new approach to probabilistic programming inference. In *Artificial Intelligence and Statistics*, 2014.

Journal of Machine Learning Research 18 (2017) 1-35 Submitted 4/16; Revised 11/16; Published 2/17

Empirical Evaluation of Resampling Procedures for Optimising SVM Hyperparameters

Jacques Wainer wainer@ic.unicamp.br
Computing Institute
University of Campinas
Campinas, SP, 13083-852, Brazil

Gavin Cawley g.cawley@uea.ac.uk
School of Computing Sciences
University of East Anglia
Norwich, NR4 7TJ, U.K.

Editor: Russ Greiner

Abstract

Tuning the regularisation and kernel hyperparameters is a vital step in optimising the generalisation performance of kernel methods, such as the support vector machine (SVM). This is most often performed by minimising a resampling/cross-validation based model selection criterion, however there seems little practical guidance on the most suitable form of resampling. This paper presents the results of an extensive empirical evaluation of resampling procedures for SVM hyperparameter selection, designed to address this gap in the machine learning literature. We tested 15 different resampling procedures on 121 binary classification data sets in order to select the best SVM hyperparameters. We used three very different statistical procedures to analyse the results: the standard multi-classifier/multi-data set procedure proposed by Demšar, the confidence intervals on the excess loss of each procedure in relation to 5-fold cross validation, and the Bayes factor analysis proposed by Barber. We conclude that a 2-fold procedure is appropriate to select the hyperparameters of an SVM for data sets for 1000 or more datapoints, while a 3-fold procedure is appropriate for smaller data sets.

Keywords: Hyperparameters; SVM; resampling; cross-validation; k-fold; bootstrap

1. Introduction

The support vector machine (SVM) (Boser et al., 1992; Cortes and Vapnik, 1995) is a powerful machine learning algorithm for statistical pattern recognition tasks, with strong theoretical foundations (Vapnik, 1998) and excellent performance in a range of real-world applications (e.g. Joachims, 1998; Furey et al., 2000; Fernández-Delgado et al., 2014). Perhaps the most common variant of the SVM uses the radial basis function (RBF) kernel, as recommended as a default approach in a popular guide to SVM (Hsu et al., 2010). The application of the RBF SVM to a classification problem requires the selection of appropriate values for two hyperparameters: a regularisation parameter, C, and a parameter governing the sensitivity of the kernel, γ. Given values for these two hyperparameters and the training data, an SVM solver, such as libSVM (Chang and Lin, 2011), can find the unique solution of the constrained quadratic optimization problem defining the SVM formulation and return a classifier. We assume that the reader is familiar with the theory of SVMs and in particular of the SVM with the RBF (also known as Gaussian) kernel.

Unfortunately, the situation is less straightforward for model selection; there is no similarly principled means of optimising the hyperparameters. The simplest approach is to divide the data set into training and testing sets, and for each C and γ from a suitable set, select the pair that result in the SVM that when trained on the training set has lowest error rate over the corresponding test set. More commonly, resampling approaches, such as *cross-validation*, use multiple test/training sets in order to form a better model selection criterion from the available data.

This paper presents an empirical investigation of the effects of different resampling approaches to hyperparameter tuning on the generalisation performance of the final classifier. The investigation is focussed primarily on the SVM with an RBF kernel, but the main conclusions are repeated and validated for the linear and polynomial kernel SVM, as discussed in Section 5.

1.1 Resampling Approaches to Performance Evaluation

Performance evaluation is a key component of model selection procedures typically used in practical applications of support vector machines. Assume we have a sample of data, $\mathcal{G} = \{z_i = (\boldsymbol{x}_i, y_i)\}_{i=1}^n$, where $\boldsymbol{x}_i \in \mathcal{X} \subset \mathbb{R}^d$ is a vector of attributes describing the i^{th} example and y_i is the corresponding class label; for binary classification tasks, $y_i \in \{-1, +1\}$. Resampling procedures provide a performance estimate based on repeatedly dividing $\mathcal{G}$ to form a *training set* and a *test set* (sometimes known as a *validation set*). More formally, in the i^{th} iteration of the resampling procedure, TR_i represents the training set and TE_i is the test set, such that

$$TR_i \cap TE_i = \emptyset \quad \text{and} \quad TR_i \cup TE_i \subseteq \mathcal{G}.$$

Let $\epsilon(\mathcal{B} \mid \mathcal{A}, C, \gamma)$ represent the error rate of an SVM trained on the training sample $\mathcal{A}$, using hyperparameter values C and γ, evaluated on the test set $\mathcal{B}$. The performance estimate provided by resampling methods is then typically the mean of the error rates obtained on the test set in each fold, i.e.

$$\text{Error}(C, \gamma) = \frac{1}{N} \sum_{i=1}^{N} \epsilon(\, TE_i \mid \, TR_i, C, \gamma),$$

where N is the number of iterations, or *folds*, of the resampling procedure. Different resampling *procedures*, such as *k-fold cross-validation*, *bootstrap* and *leave-one-out cross-validation* differ only in the way in which the data are partitioned to form TR_i and TE_i in each fold. Some common resampling procedures include:

- *k-fold cross-validation*: Partition $\mathcal{G}$ to form k disjoint sets F_j of approximately similar size, such that $\bigcup_j F_j = \mathcal{G}$. Then in each of the k iterations, a different set is used for testing and the others for training, i.e. $TE_i = F_i$ and $TR_i = \bigcup_{j \neq i} F_j$. In *stratified* cross-validation, $\mathcal{G}$ is partitioned such that each subset has a similar proportion of patterns belonging to each class. In repeated k-fold cross-validation, this procedure is performed repeatedly, with a different initial partitioning in each iteration.

©2017 Jacques Wainer and Gavin Cawley.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-174.html.

- *Leave one out cross-validation*: This is the most extreme form of k-fold cross-validation, in which each set F_i consists of a single training pattern, i.e. $TE_i = \{z_i\}$ and $TR_i = \mathcal{G} \setminus \{z_i\}$.

- *Hold-out*: A single training set, TR_1, is defined, with size $p \times n$, and $TE_1 = \mathcal{G} \setminus TR_1$, where $p \in [0,\ 1]$. In stratified hold-out, the partitioning is performed such that TE_1 and TR_1 have a similar proportion of patterns of each class. In repeated hold-out resampling, this procedure is performed repeatedly with different random partitions of $\mathcal{G}$.

- *Bootstrap:* In each iteration, TR_i is obtained by sampling n items, with replacement from $\mathcal{G}$, and $TE_i = \mathcal{G} \setminus TR_i$.

- *Subsampling*: A hold-out resampling procedure where $TR_i \cup TE_i \subset \mathcal{G}$, that is, where only a subset of the available data set is used in each iteration. This is useful where a very large amount of data is provided.

Unfortunately, the names of these procedures are not well standardised. Appendix A discusses alternative names used for the concepts and procedures discussed in this paper.

Finally, resampling should be contrasted with *resubstitution*, a performance estimation method that uses the same set for both training the SVM and measuring its error rate. There are variations on the resubstitution procedure, where the data used to measure the error rate is the same data used in training, but they are given different weights (Braga-Neto and Dougherty, 2004).

1.2 Model Selection

The process of *model selection*, in the case of kernel learning methods, refers to the tuning of the kernel and regularisation hyperparameters in order to maximise generalisation performance. The generalisation error of a classifier can be expressed as an expectation over random samples, $\boldsymbol{z}$, drawn from the distribution $\mathcal{D}$ from which the training set was obtained,

$$\check{\epsilon}(C, \gamma) = \mathbb{E}_{z \sim \mathcal{D}} \left[\epsilon(z \mid \mathcal{G}, C, \gamma) \right]$$

Ideally we would like to choose the hyperparameters C and γ so that $\check{\epsilon}$ is minimized, that is:

$$C^*, \gamma^* = \underset{C,\gamma}{\operatorname{argmin}}\ \check{\epsilon}(C, \gamma) \tag{1}$$

Unfortunately, the distribution giving rise to the data is generally unknown, and so we are unable to evaluate or directly optimise $\check{\epsilon}$. The solution is to optimise instead an *estimate*, $\tilde{\epsilon}$, of the true generalisation error, $\check{\epsilon}$. By far the most common approach is to optimise a resampling-based estimate. The estimate, $\tilde{\epsilon}$, of $\check{\epsilon}$ for a particular resampling procedure rs is defined as:

$$\tilde{\epsilon}_{rs}(C, \gamma) = \frac{1}{N} \sum_{i=1}^{N} \epsilon(TE_i \mid TR_i, C, \gamma).$$

Given that a particular resampling procedure (rs) was selected, the choice of the SVM hyperparameters is governed by:

$$C^*_{rs}, \gamma^*_{rs} = \underset{C,\gamma}{\operatorname{argmin}}\ \tilde{\epsilon}_{rs}(C, \gamma).$$

It would be computationally infeasible to evaluate every possible combination of the hyperparameters, C and γ, so in general the search evaluates combinations from a finite set $\mathcal{S}$.

$$C^*_{rs}, \gamma^*_{rs} = \underset{C,\gamma \in \mathcal{S}}{\operatorname{argmin}}\ \tilde{\epsilon}_{rs}(C, \gamma). \tag{2}$$

Different model selection procedures adopt different methods to generate the set $\mathcal{S}$; they can be specified a-priori, as in the case of grid-search or random search, or successive elements of $\mathcal{S}$ can be generated according to the results obtained from evaluating existing elements, as in the case of Nelder-Mead simplex (Nelder and Mead, 1965), gradient descent (Chapelle et al., 2002) or other non-convex methods (Friedrichs and Igel, 2005; De Souza et al., 2006).

The choice of resampling procedure depends on two possibly conflicting criteria: firstly we would like to maximise generalisation performance, and secondly reduce computational expense. The error of a resampling estimate of generalisation consists of two components, *bias* and *variance*. The bias component represents the degree to which the estimate differs *on average* from the true value, over a large number of datasets of the same size as $\mathcal{G}$ sampled from the same underlying distribution, $\mathcal{D}$. The generalisation performance of classifiers tends to improve as the size of the training set increases. Resampling estimates therefore tend to have a pessimistic bias, systematically underestimating the generalisation performance of a classifier trained on $\mathcal{G}$, as in each fold a classifier is trained on only a subset of $\mathcal{G}$. The optimal hyperparameters for an SVM, particularly the regularisation parameter (C), can also demonstrate some degree of dependence on the size of the training set, which also leads to a bias in the hyperparameter estimates from resampling based model selection procedures. The variance component reflects the difference between the estimated and true values due to the particular sample of data on which the estimate was computed (and also due the random partitioning of the sample). As the model selection procedure directly minimises the resampling estimate, the presence of a non-negligible variance component introduces a risk of over-fitting in model selection (Cawley and Talbot, 2010), which results in suboptimal hyperparameter selection. Let us define the *true* excess loss (el) of a resampling procedure rs, as

$$el(rs) = \mathbb{E}_{\boldsymbol{z} \sim \mathcal{D}} \left[\epsilon(\boldsymbol{z} \mid \mathcal{G}, C^*_{rs}, \gamma^*_{rs}) - \epsilon(\boldsymbol{z} \mid \mathcal{G}, C^*, \gamma^*) \right],$$

where C^* and γ^* are the choices of optimal hyperparameters defined in Equation 1. The first important consideration in the choice of procedure is then to minimise the excess loss.

Unfortunately, the true excess loss is unknowable: Firstly we do not know the true values of γ^* and C^*, i.e. the values minimizing the generalisation error (1), indeed if these values were known, model selection using resampling methods (2) would be entirely redundant. Secondly, we do not know the true distribution of the data, $\mathcal{D}$, so we cannot perform the expectation involved in evaluating the generalisation error (1). The best we can do is to estimate the generalisation error using a finite test sample and compare the performance obtained using model selection based on a given resampling method against model selection

using some sensible baseline method, such as five-fold cross-validation. We therefore define the excess loss relative to five-fold cross-validation as the observed difference in the error rate on a fixed test set for classifiers trained with hyper-parameters adjusted so as to minimise a given resampling based performance estimate and so as to minimise the five-fold cross-validation estimate. This gives a relative indication of the improvement in generalisation performance due to the resampling method used to tune the hyper-parameters.

The second consideration in the choice of resampling procedure is the computational cost. The different resampling procedures have very different computational costs for the hyperparameter search. For example, for each possible pair of values C and γ, 10-fold cross-validation will require the fitting of 10 separate SVM classifiers, each with a training set of $0.9 \times |\mathcal{G}|$. For 5-fold cross validation, for each pair of hyperparameters, there will be 5 classifiers trained, each with a training set of size $0.8 \times |\mathcal{G}|$ — not only fewer support vector machines need be constructed, but each is fitted to a smaller training sample, with lower computational expense. If one is using a batch SVM solver, such as SVMlight (Joachims, 1999) or libSVM (Chang and Lin, 2011), one can assume that the learning time is at least quadratic in relation to the training size (Bottou and Lin, 2007).

A trade-off between these two criteria must be reached in selecting a suitable resampling estimator. We would like to minimise the number of folds to reduce computational expense, however the variance of resampling estimators is generally reduced by increasing the number of folds, resulting in improved hyperparameter estimates. Again, we would like to reduce the size of the training set to reduce training time, but this also tends to increase the variance of the estimator. If the training set is made smaller, the uncertainty in estimating the model parameters will be greater, and hence the test set error more variable. However this leaves more data available for the test set, which tends to reduce the variance of the estimator. In reaching a compromise, we must ensure that the training and test sets are both sufficiently large, and a sufficient number of folds are used, for the estimator to have a suitably low variance, whilst at the same time limiting computational expense so that the procedure remains practical. The goal of this research is to understand the balance of these two conflicting considerations.

We must point out that this paper evaluates different *fixed* resampling procedures, as opposed to adaptive resampling (Kuhn, 2014; Krueger et al., 2015) to select the SVM hyperparameters. Fixed resampling will use the same resampling procedure for each of the possible hyperparameter combinations being tested, by far the most common procedure. However there has been some recent work on adaptive resampling where, for example, the full resampling procedure is not performed for some of the hyperparameter combination if the results for tests sets so far indicate that one can be sure that the results are suboptimal (Kuhn, 2014), or where a small subsampling is used first for all hyperparameter combinations and as one becomes increasingly sure that some of the combinations have better results than others, subsampling with increasing larger training sets are tested (Krueger et al., 2015).

1.3 Related Literature

The problems of the computational cost of hyperparameter selection for the SVM, and the generalisation performance of the resulting classifier, have been discussed in the literature in

many forms. There are in general three alternatives to improve hyperparameter selection, which will be discussed separately:

- Use a different metric derived from the training set, typically a lower bound on generalisation performance, such as Xi-alpha, span and radius/margin bounds to select the hyperparameters (e.g. Vapnik and Chapelle, 2000; Keerthi, 2002; Joachims, 2000; Wahba, 1999).
- Use different search/optimisation procedures such as random search, Nelder-Mead simplex, non-convex optimization procedures, to select the combinations of C and γ to be evaluated during model selection (e.g. Bergstra and Bengio, 2012; Huang et al., 2007; Friedrichs and Igel, 2005; De Souza et al., 2006).
- Use different resampling procedures to estimate $\tilde{\epsilon}$ (e.g. Anguita et al., 2005, 2012).

The first approach, given above, optimises an alternative metric, rather than the error rate, ϵ, i.e.:

$$C^*, \gamma^* = \underset{C,\gamma \in \mathcal{S}}{\operatorname{argmin}}\ \phi(\mathcal{G}, C, \gamma)$$

The metric, ϕ, is sometimes called an *internal metric* or a model selection criterion and they are computed from the training set alone. Internal metrics, proposed in the literature include: the span bound (Vapnik and Chapelle, 2000); the radius/margin bound (Keerthi, 2002); the Xi-Alpha bound (Joachims, 2000); GACV (Wahba, 1999); and maximal discrepancy (Anguita et al., 2005). Duan et al. (2003) compare 5-fold cross-validation with some internal metrics, such as Xi-alpha and GACV as methods to select SVM hyperparameters on five data sets and find that the 5-fold has lower excess loss. Anguita et al. (2005) compare many cross-validation procedures and some internal metrics (maximal discrepancy and compression bound) for hyperparameter selection on 13 data sets, and find that cross-validation based procedures have lower excess loss than the internal metrics.

The second approach to improving the hyperparameter selection procedure usually fixes a particular model selection criterion, say 10-fold cross-validation, and proposes different means by which the C, γ are selected from the $\mathcal{S}$ set, and more generally, how the $\mathcal{S}$ set is dynamically computed given the error for the previously selected pairs C, γ. We will call this approach, the *hyperparameter search procedure*. Most search procedures are based on the fact that the error response surface, that is the error for each value of C and γ, is generally non-convex, and thus methods based on gradient descent can only be *guaranteed* to find a local minima. The standard, or most common search procedure is a simple *grid search*, where the set $\mathcal{S}$ is predefined, usually a geometrically spaced grid in both C and γ, i.e. the $\mathcal{S}$ points are taken from a uniform 2-dimensional grid in the $\log C \times \log \gamma$ space. This is the search procedure used in this research. Bergstra and Bengio (2012) propose a random search in the space $\log C \times \log \gamma$. Huang et al. (2007) propose selecting from fixed points in the $\log C \times \log \gamma$ space following the principles of uniform design (Fang et al., 2000). Keerthi and Lin (2003) discuss the asymptotic behaviour of the error surface for an SVM in the $\log C \times \log \gamma$ space and proposes a method by which the C is optimized in a 1D grid search for the *linear* SVM problem, which yields a $\hat{C}$ value, and the C and γ for the RBF SVM is selected using a 1D grid search on the line that satisfies $\log \gamma = \log C - \log \hat{C}$. Davenport et al. (2010) propose that one should see the non-convexity of

the error surface for different C and γ as a noisy convex surface, and proposes a filtered coordinate descent search where the "true value" of the error at a particular point C, γ is a Gaussian filtered value of the error rate in a neighbourhood of C, γ. Keerthi et al. (2007) define an approximation to the gradient of the error surface and proposes that a gradient descent search should be performed to select the optimal C and γ. Finally, many researchers have proposed the use of non-convex optimization procedures to select the optimal hyperparameters, including evolutionary algorithms (Friedrichs and Igel, 2005), and particle swarm optimisation (De Souza et al., 2006; Li and Tan, 2010)

The third approach, selection of different resampling procedures, is the one explored in this paper. We know of few papers that discuss relative merits of resampling procedures and the selection of hyperparameters: The closest to this research is Anguita et al. (2005). Anguita et al. (2005) compare 9-fold cross-validation (kf9), 10-fold cv (kf10), 10 repetitions of the bootstrap procedure (10xboot), 100 repetitions of the bootstrap (100xboot), leave-one-out (loo), and 70/30 hold-out on 13 data sets (along with two internal metrics) for hyperparameter selection. They use a different experimental procedure — a fixed test set (in our notation a fixed $\mathcal{F}$ data set) and thus they can estimate $\epsilon(\mathcal{F} \mid \mathcal{G}, C^*, \gamma^*)$ and the excess loss using the fixed test set. They average the relative excess loss across the data sets (for each procedure). They report that the k-fold procedures have lower relative excess loss, followed by the 100xboot, the 10xboot, the loo and the 70/30 in that order. They did not include any discussion on whether the differences can be considered negligible. Another result reported in the paper is that 100xboot has the lower estimate error for $\epsilon(\mathcal{F} \mid \mathcal{G}, C^*, \gamma^*)$, followed by loo, 10xboot, 70/30, and last kf.

2. Methods and data

This paper describes an experimental evaluation of different resampling-based model selection criteria, using 121 different data sets (described in detail in Section 2.2). The error rate of the SVM trained with the choice of hyperparameters selected using the different resampling procedures (discussed in Section 2.1) are evaluated using a 2-fold cross-validation. That is, each data set is divided into two halves, the different resampling procedures are used to select the hyperparameters using the first half, the SVM is trained in this first half and its error rate evaluated for the second half. The procedure is repeated using the second half as training set and the first half as test set. The estimate of the error rate for the resampling procedure (or more precisely the error rate of the SVM with hyperparameters selected by the resampling procedure) is the average of the two measured error rates. If i denotes a data set, i_a and i_b two halves of the data set, then the estimated error rate for a resampling procedure rs is

$$eer(rs, i) = \frac{\epsilon(i_b \mid i_a, C^*_{rs,a}, \gamma^*_{rs,a}) + \epsilon(i_a \mid i_b, C^*_{rs,b}, \gamma^*_{rs,b})}{2} \tag{3}$$

where $C^*_{rs,a}$ and $\gamma^*_{rs,a}$ are computed as described in Equation 2, where the i_a half of the data set i corresponds to $\mathcal{G}$ (we discuss the candidate set $\mathcal{S}$ below).

For the 112 smallest data sets, we used three different statistical methods to compare the estimated error rate for each resampling procedure with that of a baseline procedure, in this case the 5-fold cross-validation (details of the comparison methods in Section 3). We

used the 9 remaining large data sets, for which the experiments above would require too much computational time, to verify the conclusions derived from the experiments on the smaller data sets.

The time required to perform hyperparameter selection for each procedure, for each data set, was also recorded and the ratio with 5-fold cross-validation calculated. The time ratio was also averaged over all procedures to compute an "expected time ratio" of the resampling procedures in relation to that of 5-fold cross-validation.

For all procedures and data sets, the hyperparameter search procedure used an 11×10 grid search (the $\mathcal{S}$ set) following the ranges and steps popularized by `libsvm` (Hsu et al., 2010) i.e. $C = \{2^{-5}, 2^{-3}, \ldots, 2^{15}\}$, and $\gamma = \{2^{-15}, 2^{-13}, \ldots, 2^{3}\}$.

2.1 Resampling Procedures

The following resampling procedures are investigated:

1. 2-fold cross-validation (kf2)
2. 3-fold cross-validation (kf3)
3. 5-fold cross-validation (kf5)
4. 10-fold cross-validation (kf10)
5. 2 times repeated 5-fold (2xkf5)
6. 2 times repeated 10-fold (2xkf10)
7. 5, 10, and 20 times repeated bootstrap (5xboot, 10xboot, 20xboot)
8. 80/20 hold-out (80/20) — a training set of size approximately 80% of the original data, and test set of 20%, with similar proportion of classes
9. resubstitution (resub), training and testing in the whole data set
10. inverted 5-fold (invkf5): learning on a single fold, and testing on the remaining.
11. 20/20 hold out (20/20) — training and test sets of 20%
12. 5 times repeated 20/20 hold out (5x20/20)
13. 20/10 holdout (20/10)
14. 10/10 hold out (10/10)
15. 5 times repeated 10/10 hold out (5x10/10)

We describe procedures 3 to 8 as *large-training set* resampling procedures. If n is the size of the $\mathcal{G}$ set ($n = |\mathcal{G}|$) then the large-training procedures will perform training on sets ranging from $0.8 \times n$ for the kf5, 2xkf5, 80/20, to n for the bootstrap[1]. The 3-fold and 2-fold (item

1. Technically the size of the training set for the bootstrap depends on the details of the implementation. The set TR_i has size n, but some of the data are necessarily repeated. If one uses a naive implementation, the training set has size n, but if one makes use of weights for each data point to account for the repetition of the data, then the training size for the bootstrap is on average $0.632n$.

1 and 2 in the list) will be called *medium-training set* procedures, since they will perform training on $0.67n$ to $0.5n$ data points. The other procedures are called *small-training set* procedures, since the training size ranges from $0.1n$ (10/10 and others) to $0.2n$ (20/20 and others). Somewhat counter-intuitively, the *small-training set* procedures are more useful for *large* data sets, When it is the case that a small fraction of G can represent the underlying data-generating distribution, such resampling procedures can obtain good accuracy at a tiny fraction of the computational cost of resampling procedures that build larger training sets.

2.2 Data sets

The 121 data sets used in this study were collected from the UCI repository (Lichman, 2013), processed and converted by the authors of Fernández-Delgado et al. (2014) into a unified format. The data is derived from the 165 available at UCI repository in March 2013. From this set, they discarded 56, mainly because they were too large (number of data and/or number of features), or because they were not in a "common UCI format". Four further data sets were added to those from the UCI repository, and finally some data sets that had two or more definitions of "classes" were converted into different problems. For example the data set *cardiotocography* defined two classification problems, one with 3 classes and the other with 10 classes — each became a different data set. The names of the data sets are a simplification of the original UCI name. For each data set, categorical features were converted to numerical data, and each feature was *standardised* to have a mean of zero and unit standard deviation. No further pre-processing or feature selection was performed by Fernández-Delgado et al. (2014). We used the data generated by the authors of Fernández-Delgado et al. (2014), downloaded in November 2014. In 19 data sets the data were divided into a training and test set, but the test set was not standardized (in the data available in Nov 2014). In these cases we standardized the test set (independently of the training set) and joined the two into a single data set. Some of the 121 original data sets are multi-class. Since the SVM is essentially a binary classification procedure, we converted the multi-class problems into binary problems by ordering each of the original classes according to their names and alternately assigned the original classes to the new positive and negative classes. Finally, we used all the data sets with less than 10,000 data for the experiments, and used the 9 data sets with more than 10,000 data to verify the conclusions from the experiments.

The characteristics of all data sets used are reported in Appendix B. The tables report the size of each half of the data set. The average data set size is 2250 patterns (median 341.5, max 65030); the average number of features is 30 (median 17, max 263). The proportion of samples belonging to the positive class is displayed in the histogram in Figure 6 in Appendix B. The proportions are approximately normally distributed with mean 0.60 and standard deviation of 0.17.

3. Metrics and statistical procedures

We perform three different statistical analyses of the results. The first is the standard multi-classifier/multiple-data set comparison procedure proposed by Demšar (2006), where the multiple resampling procedures play the part of the multiple classifiers. The second

analysis computes confidence intervals for the excess loss of different resampling procedures in relation to kf5, and verify whether this interval is within a range of equivalence. If the confidence interval on the excess loss is smaller than a threshold the error rates of the two different resampling procedures are considered to be "equivalent". This second analysis addresses some of the general criticism of procedures based on the "null hypothesis significance test" (NHST) framework and is in consonance of what is usually referred to in the medical literature as "practical significance" (as opposed to mere "statistical significance"). The final form of analysis is the Bayes factor test proposed by (Barber, 2012, chapter 12).

3.1 Demšar procedure

Demšar (2006) proposed a method to determine whether a statistically significant difference exists in the performance of multiple classifiers on many data sets. In our case, the different resampling procedures used in hyperparameter optimisation play the role of the different classifiers. Demšar suggests using the Friedman test as an omnibus nonparametric paired test (the pairing is each data set). This test computes the p-value based on the null hypothesis that all classifiers are "equivalent" in terms of their true rankings. If the p-value is low enough one would reject the claim that the classifiers are "equivalent" and should proceed to determine *which* differences are "statistically significant" or not. In the case that all classifiers are being compared on an equal basis, Demšar (2006) proposes the Nemeyi post-hoc test. In our case, we are not interested in comparing all pairs of procedures, but only in comparing each resampling procedure with the baseline provided by kf5. In this case, Demšar (2006) proposes pairwise Wilcoxon signed-rank tests (the paired version of the non-parametric Wilcoxon test), but with the correction to the resulting p-values due to the multiple comparison. He proposes either the Bonferroni-Dunn procedure, or one of the step-up/down procedures of Holm, Hochberg, or Hommel.

3.2 Confidence interval on the excess loss

The Demšar's procedure falls under the null hypothesis significance testing framework, that is, one assumes that there are no differences among the resampling procedures and declare that there is a "statistically significant" difference if the probability of a difference in mean rankings at least as large as that actually observed is below a pre-determined threshold (usually 0.05). However, even if a "statistically significant" difference exists, the effect size may be sufficiently small that it is of no relevance to practical applications. For example, it may be that the difference between one procedure and another is a decrease of (say) 0.0001 in the error rate. This difference could be "statistically significant" but would be unlikely to be "practically significant" (Kirk, 1996). One possible way of determining if there is a practically significant variation between a resampling procedure and kf5 (for instance) for selecting SVM hyperparameters would be to determine a confidence interval for the excess loss and show that (with 95% confidence) the excess loss exceeds a *threshold of equivalence*.

The important aspect of the confidence interval procedure is the definition of a "equivalence" threshold — above what level of excess loss should a difference be considered a "practically relevant" change in the error rate? In this paper we will propose a method to determine this threshold of equivalence. Besides the resampling procedures described in Section 2.1, we also evaluated another procedure which was a repetition of the kf5 procedure

but with different folds; we call it the kf5bis procedure. The difference between the kf5 and kf5bis procedure was the random generator seed, and thus the "luck/bad luck" the experimenter has in creating the folds. Thus, in some sense the excess loss of the kf5bis procedure is a limit of equivalence, not necessarily because it is small, but because it is an excess loss one cannot further reduce, since it represents the effect of "luck" in the resampling procedure for selecting the hyperparameter. Thus, in this paper we use the mean excess loss of the kf5bis procedure as the threshold of equivalence. Table 2 in Section 4.2 reports among the other resampling procedures, the mean excess loss of the kf5bis procedure as -0.0031. Thus in this paper we will consider excess loss within the range $[-0.0031, 0.0031]$ as irrelevant. Simplifying, the NHST approach would compute the confidence interval for the excess loss and declare that the excess loss is statistically significant if the confidence interval does not cross the zero. A "practical significance" approach computes the same confidence interval for the excess loss, but declares that the excess loss is "irrelevant" if the interval is fully contained in the $[-0.0031, 0.0031]$ range.

3.3 Bayes factor

The third method to compare error rates across different data sets is the Bayesian analysis proposed by Barber (2012, chapter 12). The excess loss analysis above is based solely on the value of the error rate. The Bayesian method is based on both the magnitude of the error rate and the number of samples used in evaluating that error rate. For example, one will be more willing to assume that a classifier a which made 50 errors over 500 samples is equivalent to a classifier b that made 55 errors in 500, than if the first made 700 errors in 7000 samples, while the second made 770. Although the change in error rate is the same in both cases (0.10 versus 0.11), nevertheless, because of the larger test set, one is less sure that the two classifiers are equivalent in the second scenario. The Bayesian analysis measures the ratio between the probability that the classifiers are the equivalent versus the probability that they are not equivalent (given the data), and this *Bayes factor* should be much lower in the second scenario.

Given two classifiers evaluated on the same data set (or in our case the classifiers based on the different choices of hyperparameters derived using different resampling procedures), the method computes "How much evidence is there that the two samples of correct and incorrect predictions in the test set comes from independent multinomial distributions?" which is a possible rephrasing of the question "How much evidence is there supporting the contention that the two classifiers are performing differently?" Let us assume that classifier a when applied to the test set results in $e_a = \langle c_a, i_a \rangle$ where c_a is the number of correct predictions, and i_a the number of incorrect predictions, and similarly for classifier b. $P(H_{\text{same}} \mid e_a, e_b)$ is the posterior probability that the pairs of correct and incorrect results e_a and e_b come from the same (unknown) binomial distribution, which would indicate that both classifiers are equivalent. $P(H_{\text{indep}} \mid e_a, e_b)$ is the posterior probability that they came from independent distributions and therefore that the two classifiers are not equivalent (more precisely, it will be very unlikely that the two independent distributions are the same). The Bayes factor (BF) is the ratio of these two probabilities:

$$BF = \frac{P(H_{\text{same}} \mid e_a, e_b)}{P(H_{\text{indep}} \mid e_a, e_b)} \tag{4}$$

The larger the Bayes factor, the higher is the evidence towards the hypothesis that the two classifiers are equivalent. If $Z(x)$ is the beta function of a pair, and $u = \langle 1, 1 \rangle$, then the BF is calculated as

$$BF = \frac{Z(u)Z(u + e_a + e_b)}{Z(u + e_a)Z(u + e_b)} \tag{5}$$

Appendix C contains the derivation of this formula. In our case, we will consider 5-fold cross-validation as the baseline, and we will compare all other procedures to it, and thus we will calculate for each resampling procedure its Bayes factor in relation to 5-fold cross-validation.

We will report the $2 \log_e BF$ as defined in Equation 4. The reason to use the log is that the BF is a multiplicative factor, and for the mean and confidence interval calculations we need an additive factor. The use of the constant 2 is to follow the table of Kass and Raftery (1995) regarding the interpretation of strength of the evidence in favour of one or the other hypothesis, which is based on $2 \log_e$.

3.4 Computational Expense

Considering the time consumed for hyperparameter selection via the different resampling procedures, we ran all of the procedures for a single data set in sequence on a single core (of a multiple core machine). Different data sets were distributed to different cores of the same machine. We collected the total time to perform the resampling procedure to select the hyperparameters — the time to learn the final classifier with the optimal hyperparameters and to apply it to the other half of the data set was not included in the time measure. Again we use 5-fold cross-validation as baseline, and report the ratio of the execution time of each procedure and the 5-fold cross-validation execution time. The statistical calculations are performed with the log of the time ratio, which was then converted back to report the mean and confidence interval.

3.5 Statistical procedure

Besides the statistical tests performed by the Demšar procedure, we are interested in estimation of the mean excess loss, the log BF, and the ratio of execution time for each resampling procedure, in relation to the 5-fold. To evaluate the confidence interval of the mean of each of these measures, we use a bootstrap procedure with the "BCA" (bias- corrected and accelerated) evaluation method for the confidence interval (Efron, 1987). We use a 95% confidence level and $5,000$ replications of the bootstrap procedure.

3.6 Reproducibility

The 121 data sets, the program used to run the hyperparameter search, the raw results for each resampling procedure and data set, the program to analyse the results and generate the figures in this paper are available at `https://dx.doi.org/10.6084/m9.figshare.1359901`.

4. Results

In this section we report the results obtained from all experiments using the 112 smallest data sets, for which the experiments were computationally feasible.

4.1 Demšar procedure

We performed the Friedman test (as implemented in the libraries of the R programming language) with the following results: Friedman $\chi^2 = 382.8071$, d.f. $= 17$, p-value $<$ 2.2e-16. Thus, we reject the hypothesis that all resampling procedures are equivalent. We then performed the Wilcoxon signed-rank test for each procedure against kf5. The resulting p-values are given in Table 1. The first column is the average rank of each resampling procedure. The table includes kf5 as the first entry. The results indicate that there is no clear winner among the different resampling procedures. Notice that all average ranks for the medium and large training set procedures are similar (around 7) with the exception of the resubstitution (average rank of 15.2). Note that the average rank for the kf5 procedure itself is 7.2; thus procedures with average rank less than 7.2 are "better" then kf5, while a larger average rank indicates "worse" resampling procedures.

The second column of Table 1 displays the original p-value (from the Wilcoxon test) without any correction for multiple comparisons. The third column shows the result of Holm's correction, followed by the results of the Hochberg, Hommel, and Bonferroni-Dunn corrections, where p-values below 0.05 (which indicates a 95% confidence) are in bold. Again the results of these tests suggest that resubstitution, and the *small training set* procedures (except 5x20/20) are inferior to kf5, and the other methods are statistically equivalent, or more precisely, we cannot show that the other methods are statistically dissimilar. Table 1 shows that besides resub, all other large and medium training procedure are not statistically significantly different from kf5.

4.2 Excess loss

Table 2 reports the mean and 95% confidence interval of the excess losses for each of the resampling procedures. Figure 1 repeats the excess loss data of Table 2, removing the resubstitution data. The result of the mean excess loss for the kf5bis is -0.0031. As discussed, the absolute value of that excess loss is our threshold of equivalence; excess losses with absolute value lower than 0.0031 are considered in this paper as irrelevant.

4.3 Bayes factor

Table 3 reports the results of the Bayes factor calculations. Most of the results for the log BF are within the range from 5.5 to 6.1 in favour of the hypothesis that the error rate of each of the resampling procedures are the same as kf5. Following the scale of interpretation given by Kass and Raftery (1995), if the $2\log_e$ of the Bayes factor is in the range from 2 to 6, the evidence should be considered "positive", and from 6 to 10 "strong". Thus most of the results are in the direction of a positive evidence in favour that the resampling procedures results are the same as the 5-fold results.

A more specific threshold is derived from the kf5 and kf5bis results (first two lines in Table 3). kf5 is the result of calculating the Bayes factor for the kf5 procedure itself, so we

procedure	mean rank	original	Holm	Hochberg	Hommel	Bonferroni
kf5	7.2					
kf5bis	6.7	0.08	0.51	0.51	0.40	1.00
kf2	6.8	0.66	1.00	0.78	0.78	1.00
kf3	6.3	0.04	0.39	0.36	0.30	0.74
2xkf5	6.2	0.23	0.99	0.71	0.68	1.00
kf10	6.5	0.04	0.39	0.36	0.30	0.76
5xboot	7.2	0.78	1.00	0.78	0.78	1.00
10xboot	5.8	0.20	0.99	0.71	0.60	1.00
20xboot	5.8	0.06	0.41	0.41	0.35	0.99
80/20	9.3	0.01	0.14	0.14	0.14	0.24
resub	15.2	0.00	**0.00**	**0.00**	**0.00**	**0.00**
invkf5	7.8	0.24	0.99	0.71	0.71	1.00
20/80	8.8	0.00	**0.02**	**0.02**	**0.02**	**0.02**
20/20	10.6	0.00	**0.00**	**0.00**	**0.00**	**0.00**
5x20/20	10.3	0.01	0.14	0.14	0.12	0.21
20/10	10.3	0.00	**0.00**	**0.00**	**0.00**	**0.00**
10/10	11.1	0.00	**0.00**	**0.00**	**0.00**	**0.00**
5x10/10	10.3	0.00	**0.00**	**0.00**	**0.00**	**0.00**

Table 1: The result of the Demšar procedure comparing the resampling procedures. First column names the procedure, second display the average rank. The following columns display the p-value of the Wilcoxon signed rank test when compared with kf5: the original p-value, and then after the Holm, Hochberg, Hommel, and Bonferroni corrections

are sure that both procedures have exactly the same expected generalization error, and yet for this case, the log Bayes factor is on average 5.89. Thus, 5.89 is the mean theoretical maximum of the log Bayes factor for the data sets used in the experiment. Thus, the results in Table 3 are very close to the mean maximum. The kf5bis provides a limit to what is a "relevant" certainty. By design, the kf5bis results should be irrelevantly different than their kf5 counterpart — it could be that for one data set, the kf5bis result is sufficiently different to the kf5 result, but we are averaging over 112 data sets. So, we would like to make the claim that Bayes factor above 5.70, which is the average of the kf5bis, show that the differences are irrelevant to this problem. Figure 2 repeats the Bayes factor data Table 3 for data points around the 5.70 threshold above which one should consider that there is enough evidence that the resampling procedure is equivalent to the 5-fold.

4.4 Computational Expense

Table 4 reports the mean and confidence interval of the ratio of the time required for each resampling procedure and that for kf5. We would like to point out some anomalies or

procedure	mean	(95%CI)
kf5bis	−0.0031	−0.0073 , 0.0002
kf2	−0.0012	−0.0056 , 0.0024
kf3	−0.0031	−0.0094 , 0.0002
2xkf5	−0.0014	−0.0043 , 0.0019
kf10	−0.0020	−0.0054 , 0.0023
5xboot	0.0014	−0.0021 , 0.0053
10xboot	−0.0022	−0.0065 , 0.0009
20xboot	−0.0018	−0.0047 , 0.0016
80/20	**0.0053**	−0.0003 , 0.0115
resub	**0.1354**	0.1105 , 0.1632
invkf5	0.0009	−0.0038 , 0.0044
20/80	**0.0074**	0.0009 , 0.0133
20/20	**0.0191**	0.0102 , 0.0323
5x20/20	**0.0052**	−0.0012 , 0.0162
20/10	**0.0217**	0.0118 , 0.0357
10/10	**0.0255**	0.0154 , 0.0372
5x10/10	**0.0160**	0.0083 , 0.0274

Table 2: Excess loss in relation to kf5 for all the resampling procedures. The second column is the mean excess loss (in relation to kf5) of each procedure; the third and fourth columns are the 95% confidence interval for the mean. In bold, the mean excess losses that are of practical significance.

unexpected results regarding the computational expense. The first one is that resubstitution is slower than kf5. A second one is that the 5-times repeated 20/20 and 10/10 are only three times as expensive as the corresponding non-repeated procedure. We do not know how to explain these results, but it is possible that they are also due to interference between different runs; as discussed in Section 3.4 the many cores of the machines were executing the experiments in different data sets at the same time.

5. Discussion

The results for the medium-training procedures (kf2 and kf3) are a welcome surprise. The Demšar analysis shows that they are not significantly different to those obtained using kf5. Furthermore, the excess loss confidence interval shows that their excess loss is mostly within the irrelevance threshold, and the Bayes factor indicates that there is good evidence that their performances are equivalent to that of kf5. However, these two procedures select the SVM hyperparameters in 33% and 55% of the time of the kf5 procedure — a useful computational saving.

Most of the large-training procedures have a negative mean excess loss, that is, they select hyperparameters that result in slightly lower error rates for the future data. The two

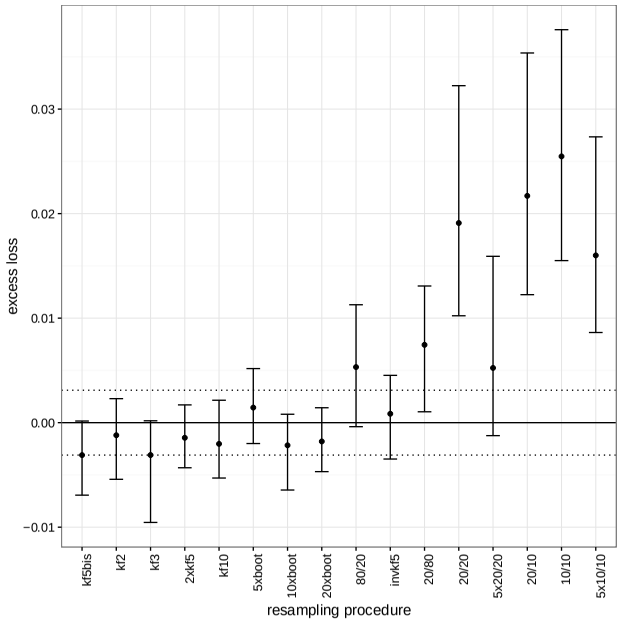


Figure 1: Excess loss in relation to kf5 for the resampling procedures (resubstitution not included). The dotted line indicates the limit for what is considered an irrelevant change in the loss.

exceptions are the 80/20 and the resub procedures discussed below. But for all procedures that result in a better selection of the hyperparameters the mean excess loss it is still within our range of equivalence to kf5. That is, even using computationally more expensive procedures such as 20- and 10-times bootstrap, 2 times repeated 5-fold, on average it is not likely that there will be relevant changes in the final error rate of the classifier. This result is consistent with the Demšar analysis, which show no statistically significant differences between these procedures and kf5. The two large-training resampling procedures that have positive excess loss are the 20% hold-out (80/20) and the resubstitution estimator. The result for resubstitution is widely known—the use of the same data for training and testing causes severe overfitting—but we have shown that the overfitting is also severe in regards to hyperparameter selection.

The small-training procedures all incur positive excess losses, to varying extents, and thus are in general "worse" than kf5, but the inverse-kf5 and the 20/80 holdout have the lowest mean excess loss. The excess loss for the 20/20 procedures (sample 20% for training and 20% for testing), and for the 20/10 and 10/10 procedures are very significant. We

procedure	mean	(95%CI)
kf5	5.89	5.52 , 6.33
kf5bis	5.70	5.33 , 6.11
kf2	5.61	5.24 , 6.05
kf3	5.73	5.35 , 6.16
2xkf5	5.74	5.37 , 6.19
kf10	5.67	5.30 , 6.11
5xboot	5.67	5.29 , 6.11
10xboot	5.73	5.35 , 6.16
20xboot	5.72	5.34 , 6.15
80/20	5.32	4.88 , 5.75
resub	−150.53	−274.41 , −81.73
invkf5	5.48	5.05 , 5.94
20/80	5.19	4.69 , 5.70
20/20	3.14	−1.49 , 4.45
5x20/20	4.93	3.85 , 5.52
20/10	−2.85	−30.38 , 3.26
10/10	2.97	1.26 , 4.05
5x10/10	4.51	3.45 , 5.16

Table 3: $2\log_e$ of the Bayes factor of the hypothesis that the resampling procedure results are the same as the kf5 results versus the hypothesis that they are independent. The numbers in the table are the mean difference of the log of the Bayes factor between the procedures indicated in the left column and the kf5. The last two columns are the 95% confidence interval for the mean.

conclude from these observations, that there seems to be no real gain in using resampling procedures more costly than 5-fold cross-validation. On the other hand, the kf2 and kf3 procedures seem to achieve a similar result to kf5, with lower computational costs.

An important issue is whether the results for the equivalence, for practical purposes, of the kf2, kf3 and kf5 procedures carries forward to larger and higher dimensional data. We tested the kf2 and kf3 on the 9 largest data sets. The results are in Table 5. The table shows the excess loss in relation to kf5, the log BF of both procedures and the kf5 log BF (which indicates the maximum value) and the time ratios in relation to kf5. With the exception of the nursery data set, all excess losses are within our limit or irrelevance, and the BF are close to the maximum, a very strong evidence that these procedures perform equally well for larger size data sets. Figures 3 and 4 show the values of the excess loss as a function of the data set size and the number of dimensions of the data. For data sets with sizes beyond the examples tested, one can look at the trend of excess loss as a function of the data set size in Figures 3 and 4: there is a compelling trend of diminishing excess loss as the data set size increase. Therefore we are very confident that one can safely use kf2 or kf3 to tune

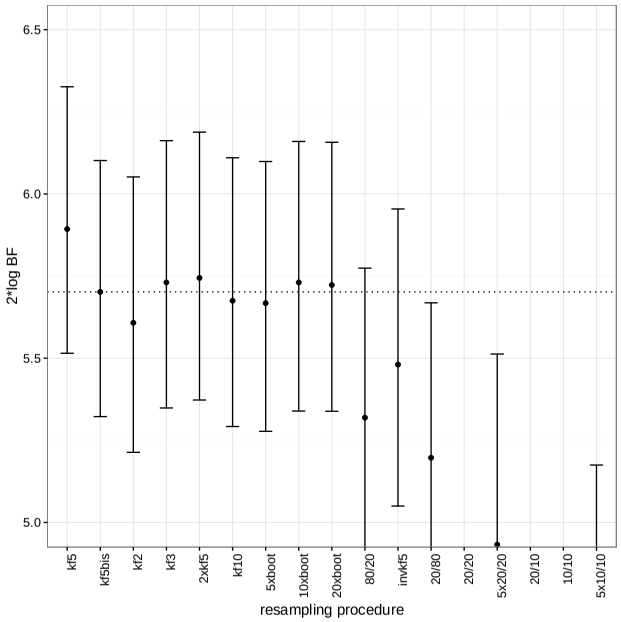


Figure 2: Bayes factor of the ratio of the probabilities that the resampling procedure and the 5-fold are equivalent. Data around the 5.7 threshold, above which one should consider that there is enough evidence to claim the equivalence.

SVM hyperparameters in large data sets. Note that as the size of the dataset increases, the bias and variance of most resampling procedures tend to decrease, so this is entirely in accord with a-priori intuition.

The evidence of a decreasing trend for larger dimensionality is less compelling for the kf2 than it is for the kf3. For one data set with larger dimensionality, the excess loss of kf2 is outside our range of irrelevance, but that is not true for the kf3. It should be noted though that the RBF kernel is often unsuitable for high-dimensional datasets and a linear kernel is often preferable in such cases (Hsu et al., 2010).

Another important issue is whether the practical equivalence of the kf2 and kf3 to kf5 is also valid for an SVM with other kernels. We evaluated the excess loss of kf2 and kf3 on the 112 smallest datasets on SVM with polynomial and linear kernels. The results are displayed in Table 6. The data are also displayed in Figure 5. The results are not too different than the excess loss for the RBF SVM, with the exception of a somewhat higher excess loss for the kf2 for linear SVM. So we believe that the conclusions of practical equivalence of the kf2

procedure	mean	(95%CI)
kf2	0.35	0.31 , 0.39
kf3	0.54	0.52 , 0.57
2xkf5	1.95	1.90 , 2.01
kf10	2.39	2.32 , 2.45
5xboot	1.24	1.20 , 1.28
10xboot	2.28	2.21 , 2.35
20xboot	4.43	4.28 , 4.56
80/20	0.31	0.30 , 0.32
resub	1.45	1.37 , 1.52
invkf5	0.46	0.40 , 0.51
20/80	0.22	0.19 , 0.25
20/20	0.11	0.09 , 0.13
5x20/20	0.33	0.27 , 0.39
20/10	0.17	0.14 , 0.20
10/10	0.09	0.07 , 0.12
5x10/10	0.28	0.22 , 0.35

Table 4: Time ratio between the resampling procedure and the kf5.

db	size	kf2 loss	kf3 loss	kf2 BF	kf3 BF	kf5 BF	kf2 time	kf3 time
pendigits	5496	−0.0004	0.0002	11.8	11.7	11.9	0.56	0.40
nursery	6480	0.0014	0.0014	10.2	10.2	10.8	0.21	0.44
magic	9510	−0.0004	0.0018	8.8	8.7	8.8	0.20	0.40
letter	10000	0.0007	−0.0000	10.4	10.6	10.6	0.20	0.39
chess-krvk	14028	0.0073	0.0000	6.6	8.8	8.8	0.13	0.31
adult	24421	−0.0002	−0.0008	9.6	9.6	9.6	0.14	0.31
statlog-shuttle	29000	−0.0001	0.0	15.5	15.6	15.6	0.32	0.45
connect-4	33778	0.0012	0.0012	9.9	9.9	10.1	0.27	0.36
miniboone	65032	−0.0002	−0.0005	11.3	11.2	11.4	0.17	0.52

Table 5: kf2 and kf3 excess loss in relation to kf5, the 2*log BF of kf2, kf3 and kf5 (which shows the maximum value of the BF) and the time ratio for kf2 and kf3

and kf3 resampling procedures in relation to the kf5 resampling to select hyperparameters are also valid for the linear and polynomial SVM.

5.1 Limits of this research

The strength of the conclusions of this research rests on the assumption that the set of 121 data set used are a good sample of "real life" data sets. There are limits to this assumption. Fernández-Delgado et al. (2014) did not include large data sets (large number of data points

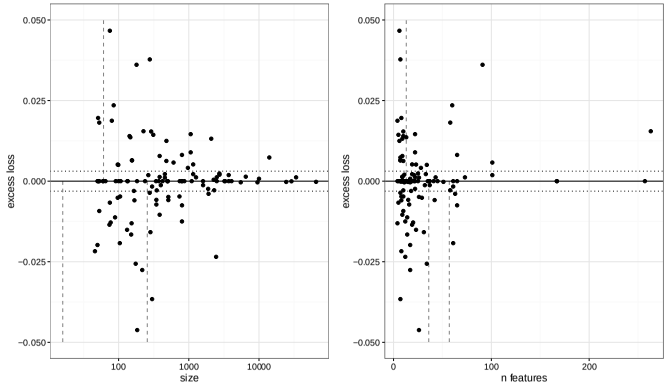


Figure 3: The excess loss of the kf2 procedure in relation to the dataset size and dimensionality. The dotted horizontal line is our threshold of irrelevance. The dashed transparent vertical line indicates that there is a data point at that position that was clipped from the graph.

Kernel	procedure	mean	min	max
Polynomial	kf2	**0.006**	0.002	0.010
	kf3	0.001	-0.003	0.004
Linear	kf2	0.002	0.000	0.005
	kf3	0.002	0.000	0.007

Table 6: Excess loss for the kf2 and kf3 procedures on SVm with Polynomial and Linear kernels.

or large number of features). Thus, our sample does not reflect large data sets (especially text-classification datasets which also have a high dimensionality). Practitioners working with these kinds of data sets are advised to check our conclusions before following them.

It has been suggested that the outer 2-fold procedure to estimate $\check{\epsilon}$ was probably too noisy since kf2 is a high variance estimator. Appendix E shows that even using a 5xkf2 (on the 42 data sets with at most 400 data) the results do not change significantly — but the confidence intervals are somewhat reduced. Therefore, even if a lower variance estimator of $\check{\epsilon}$ was used, we do not believe the results would change substantively.

This research assumed that kf5 was the "standard" resampling procedure for hyperparameter selection, and compared all other procedures against kf5. The many results in the paper show, among others, kf2 and kf3 are equivalent from a practical point of view to kf5. But that might not be the case for more costly, low variance procedures such as kf10 or 20xboot. Appendix F shows the result of the Nemenyi test for comparing all resampling

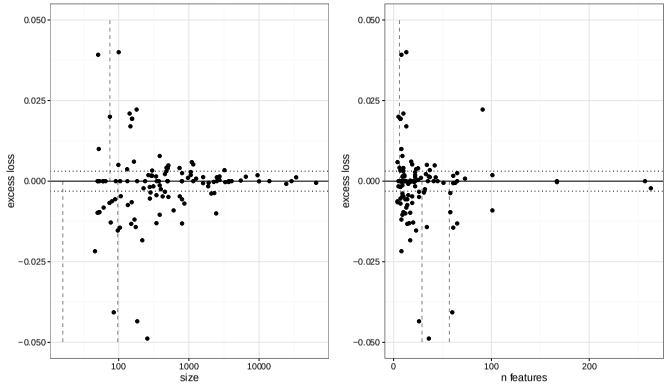


Figure 4: The excess loss of the kf3 procedure in relation to the dataset size and dimensionality. The dotted horizontal line is our threshold of irrelevance. The dashed transparent vertical line indicates that there is a data point at that position that was clipped from the graph.

procedures against each other. None of the large-training and medium-training procedures are statistically significantly different from each other, with the exception of 80/20 and resub which were also not equivalent to kf5.

5.2 Methodological discussion

We believe that besides the important result of a strong suggestion of using 2-fold or 3-fold cross validation to select the hyperparameters of an SVM this research also opens an important methodological discussion on at least three topics. The first is the use of a Bayesian framework for the comparison of classifiers. The second is the use of concepts of "practical irrelevance" or "practical equivalence" in machine learning, and in particular the use of the kf5bis to discover the threshold of equivalence. The final issue, is whether it is methodologically sound to average excess losses. The Bayes factor approach resulted in a metric that is not very sensitive. Most of the results in Table 3 are similar to each other and almost all are in the range of positive evidence, but as we have shown using the results of kf5, this seems to be the strongest evidence possible in this problem. Our decisions regarding the BF should be reviewed by future researchers in search of a more sensitive metric. For example, we used only the number of correct and incorrect predictions in the Bayesian calculations. Barber (2012) implicitly suggest both these measures and also using the 4-tuple of true positives, true negatives, false positives, and false negatives.

We must explain why we used two other analysis procedures beyond the usual Demšar proposal. As we discussed, the Demšar (2006) procedure falls within the NHST framework, and other empirical sciences realized that NHST are rarely the correct tool to analyse data

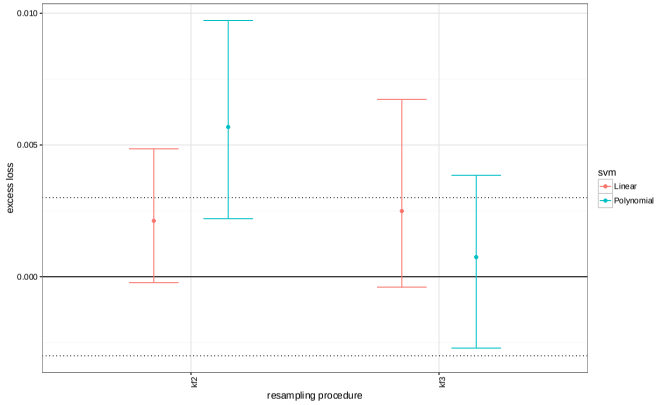


Figure 5: The excess loss of the kf2 and kf3 procedures for Linear and Polynomial SVM, measured on the 112 smallest dataset. The dotted line is the threshold of practical equivalence.

(Gardner and Altman, 1986; Thompson, 2002; Woolston, 2015). Significance tests can tell us that two samples are different enough that it is unlikely that they came from the same population, but not whether the difference matters for any practical purpose. Alternatively, significance tests can tell us that there is not enough evidence to make the claim that the two samples came from the same population, which in no way means that the two samples are "the same." Our proposal of using a confidence interval on an effect size and a measure of practical equivalence is one possible direction, as is our usage of Barber (2012) Bayesian analysis.

Another methodological proposal was to use the kf5bis (a different random selection for the kf5) as a way of determining a threshold of equivalence. The differences between kf5 and kf5bis should on average over many data sets be irrelevant, either because they are too small to matter or because there is no way to reduce them.

The final discussion is whether computing the mean (and confidence interval) of a excess loss is a sound procedure. An argument against it is that error rates (and therefore differences in error rates) cannot be compared across different data sets. One can argue that an improvement of 0.05 in the error rate would be a substantial improvement if the error rate was originally 0.08; however an improvement of 0.05, where the original error rate was 0.28, is rather less significant. The same number (0.05) seems to mean different things in these two situations, and therefore adding those two numbers (to compute the average) does not seems a reasonable operation. We believe that the source of this problem is that the variance of the error rates across different resampling procedures, for example, varies as a function of the error rate itself. If the variance of error rates for low error rates is low, then 0.05 is a considerable magnitude (in standard deviations) for an error rate of 0.08. The

variance should be much larger for an error rate of 0.28, which in turn means that 0.05 is not such a large change in that case. But as Appendix D shows, the variance of the excess loss (for all procedures) does not increase as the error rate (for the kf5 procedure) increases. In fact the variance can be seen as constant across the different error rates, and that would allow one to compute the average of the excess losses.

6. Conclusions

This research has evaluated the impact of different resampling procedures in the selection of SVM hyperparameters, by comparing 17 different procedures in 121 data sets. The conclusion is that the 2-fold procedure should be used in data sets with more than 1000 points. In these cases the user may expect a difference of -0.0031 to 0.0031 in the error rate of the classifier if a 5-fold procedure was used, which we believe is the limit of what one should consider an irrelevant change in the classifier error rate. For smaller data sets, we could not detect any significant difference (on average) between 5-fold and computationally more costly procedures such as 10-fold, 5 to 20 times repeated bootstrap, or 2 times repeated 5-fold. We believe that a 3-fold is appropriate for smaller data sets.

Acknowledgments

The authors would like to thank the anonymous reviewers and the editor for their detailed and constructive comments that have significantly improved this paper, and Nicola Talbot for her careful proof-reading and copy-editing. The first author would like to thank a Microsoft Azure Research Award that allowed to run some of the experiments in this research in the Azure cloud.

Appendix A. Terminology

resampling vs cross-validation. What we called *resampling* is frequently called *cross-validation*. We decided to use *resampling* instead of the most common term *cross-validation*, because we understand that the "cross" term in *cross validation* implies a subset that is used in training and then again in testing, as in the k-fold procedure. That is not necessarily true for the bootstrap procedure — there is no subset that is by construction used in the training and then in the testing. Resampling procedures are also called *out-of-sample* techniques (Anguita et al., 2012).

k-fold also known as *k-fold cross validation* or *cross-validation* by itself (Kuhn et al., 2014).

hold out is sometimes called *leave group out* (Kuhn et al., 2014) or *split sample* (Molinaro et al., 2005).

model selection (Cawley and Talbot, 2010) is also called *selection of hyperparameters* or *hyperparameter optimization* (Bergstra and Bengio, 2012) or *hyperparameter tuning* (Duan et al., 2003)

bootstrap: in the version of bootstrap described here the error rate is measured solely on the test set which contains only the data not included in the training set. Two other variations of bootstrap are commonly used: the .632-bootstrap (Efron, 1983) and the .632+-bootstrap (Efron and Tibshirani, 1997). For these methods the error estimate is calculated using not only the test set, but also the error rate of the training set itself.

Appendix B. Data sets

Table 7 lists the characteristics of all data sets, ordered by size. The name of the data set is the same as those used in Fernández-Delgado et al. (2014). The size refers to one half of the data set (please refer to section 2 regarding the methodology to compute the excess log loss). The *nfeat* column is the number of features or number of dimensions of the data in the data sets; *ndata* the number of patterns; *prop* the proportion of data in the positive class. *Notes* has the following values:

- r the data set was removed from all analysis because all cross validation tests have less than 5 examples of a particular class
- s some of the experiments did not run because their test sets had less than 5 examples of a particular class
- m the data set describes a multiclass problem, and so was converted to a binary problem using the procedure discussed in section 2
- l the data set has more than 5000 patterns so it was used in the large data set validation.
- t the data set was originally divided into a test and train set, where the test set was not standardized (see section 2).

Figure 6 displays the histogram of the proportion of the positive class.

Table 7: The data sets

data set	nfeatures	ndata	prop	note
trains	30	5	0.40	r
balloons	5	8	0.50	r
lenses	5	12	0.83	r
lung-cancer	57	16	0.56	r
pittsburg-bridges-SPAN	8	46	0.52	m
fertility	10	50	0.96	s
zoo	17	50	0.62	m
pittsburg-bridges-REL-L	8	51	0.82	m
pittsburg-bridges-T-OR-D	8	51	0.86	
pittsburg-bridges-TYPE	8	52	0.62	m
breast-tissue	10	53	0.53	m
molec-biol-promoter	58	53	0.49	
pittsburg-bridges-MATERIAL	8	53	0.94	ms
acute-inflammation	7	60	0.52	
acute-nephritis	7	60	0.67	
heart-switzerland	13	61	0.38	m
echocardiogram	11	65	0.74	
lymphography	19	74	0.46	m
iris	5	75	0.72	m
teaching	6	75	0.64	m
hepatitis	20	77	0.22	
hayes-roth	4	80	0.57	mt
wine	14	89	0.60	m
planning	13	91	0.74	
flags	29	97	0.47	m
parkinsons	23	97	0.25	
audiology-std	60	98	0.64	mt
breast-cancer-wisc-prog	34	99	0.77	
heart-va	13	100	0.52	m
conn-bench-sonar-mines-rocks	61	104	0.54	
seeds	8	105	0.64	m
glass	10	107	0.40	m
spect	23	132	0.67	mt
spectf	45	133	0.19	t
statlog-heart	14	135	0.53	
breast-cancer	10	143	0.69	
heart-hungarian	13	147	0.63	
heart-cleveland	14	151	0.75	m
haberman-survival	4	153	0.75	
vertebral-column-2clases	7	155	0.68	

continued in the next page

Table 7: The data sets (continued)

data set	nfeatures	ndata	prop	note
vertebral-column-3clases	7	155	0.67	m
primary-tumor	18	165	0.68	m
ecoli	8	168	0.64	m
ionosphere	34	175	0.30	
libras	91	180	0.51	m
dermatology	35	183	0.64	m
horse-colic	26	184	0.64	t
congressional-voting	17	217	0.59	
arrhythmia	263	226	0.67	m
musk-1	167	238	0.58	
cylinder-bands	36	256	0.38	
low-res-spect	101	265	0.25	m
monks-3	7	277	0.48	t
monks-1	7	278	0.51	t
breast-cancer-wisc-diag	31	284	0.63	
ilpd-indian-liver	10	291	0.72	
monks-2	7	300	0.64	t
synthetic-control	61	300	0.51	m
balance-scale	5	312	0.53	m
soybean	36	341	0.42	mt
credit-approval	16	345	0.43	
statlog-australian-credit	15	345	0.32	
breast-cancer-wisc	10	349	0.66	
blood	5	374	0.76	
energy-y1	9	384	0.80	m
energy-y2	9	384	0.75	m
pima	9	384	0.66	
statlog-vehicle	19	423	0.52	m
annealing	32	449	0.81	mt
oocytes_trisopterus_nucleus_2f	26	456	0.41	
oocytes_trisopterus_states_5b	33	456	0.98	m
tic-tac-toe	10	479	0.34	
mammographic	6	480	0.56	
conn-bench-vowel-deterding	12	495	0.53	mt
led-display	8	500	0.51	m
statlog-german-credit	25	500	0.72	
oocytes_merluccius_nucleus_4d	42	511	0.31	
oocytes_merluccius_states_2f	26	511	0.93	m
hill-valley	101	606	0.49	t
contrac	10	736	0.79	m

continued in the next page

Table 7: The data sets (continued)

data set	nfeatures	ndata	prop	note
yeast	9	742	0.55	m
semeion	257	796	0.50	m
plant-texture	65	799	0.50	m
wine-quality-red	12	799	0.56	m
plant-margin	65	800	0.52	m
plant-shape	65	800	0.52	m
car	7	864	0.28	m
steel-plates	28	970	0.66	m
cardiotocography-10clases	22	1063	0.39	m
cardiotocography-3clases	22	1063	0.86	m
titanic	4	1100	0.66	
image-segmentation	19	1155	0.58	mt
statlog-image	19	1155	0.55	m
ozone	73	1268	0.97	
molec-biol-splice	61	1595	0.75	m
chess-krvkp	37	1598	0.48	
abalone	9	2088	0.68	m
bank	17	2260	0.88	
spambase	58	2300	0.61	
wine-quality-white	12	2449	0.48	m
waveform-noise	41	2500	0.66	m
waveform	22	2500	0.68	m
wall-following	25	2728	0.78	m
page-blocks	11	2736	0.92	m
optical	63	2810	0.51	mt
statlog-landsat	37	3217	0.56	t
musk-2	167	3299	0.85	m
thyroid	22	3600	0.94	mt
ringnorm	21	3700	0.49	
twonorm	21	3700	0.49	
mushroom	22	4062	0.51	
pendigits	17	5496	0.51	mlt
nursery	9	6480	0.68	ml
magic	11	9510	0.65	l
letter	17	10000	0.50	ml
chess-krvk	7	14028	0.53	ml
adult	15	24421	0.76	lt
statlog-shuttle	10	29000	0.84	mlt
connect-4	43	33778	0.75	l
miniboone	51	65032	0.28	c

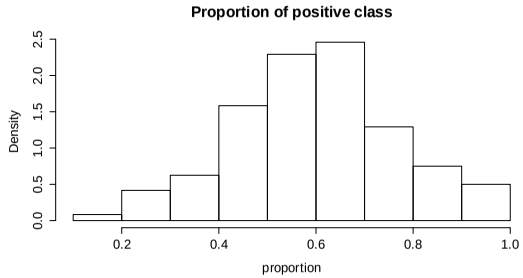


Figure 6: Histogram of the proportion of the positive class for the 121 data sets analysed.

Appendix C. Derivation of the Bayes factor formula

The derivation below is taken from chapter 12 of Barber (2012) by fixing the data for each classification (e_i) to be a pair number-of-correct-predictions and number-of-incorrect-predictions, and by computing the BF as $\frac{p(H_{\text{same}}|e_a,e_b)}{p(H_{\text{indep}}|e_a,e_b)}$ instead of $\frac{p(h_{\text{indep}}|e_a,e_b)}{p(h_{\text{same}}|e_a,e_b)}$. Let us assume that classifier a when running on the test set results in $e_a = \langle c_a, i_a \rangle$ where c_a is the number of correct predictions, and i_a the number of incorrect predictions, and similarly for classifier b. The standard Bayesian model selection method is to calculate:

$$\frac{P(H_1 \mid e_a, e_b)}{P(H_2 \mid e_a, e_b)} = \frac{P(e_a, e_b \mid H_1)}{P(e_a, e_b \mid H_2)} \frac{P(H_1)}{P(H_2)}.$$

In this case, H_1 is the hypothesis that the two data e_a and e_b come from independent multinomial distributions (H_{indep}) and H_2 that they come from the same distribution (H_{same}). Also let us assume that there is no prior preference for choosing any of these two hypothesis, and thus $P(H_{\text{indep}}) = P(H_{\text{same}})$.

For both hypothesis we will assume that the two values c_i and i_i are sampled from a multinomial distribution, $P(\langle c_a, i_a \rangle = \langle a_1, a_2 \rangle \mid \alpha = \rangle \alpha_1, \alpha_2 \rangle) = \alpha_1^{a_1} \alpha_2^{a_2}$, with an unknown α. α is sampled from a Dirichlet distribution:

$$P(\alpha) = \frac{1}{Z(u)} \alpha_1^{u_1 - 1} \alpha_2^{u_2 - 1},$$

where $Z(u)$ is the normalizing constant

$$Z(u) = \frac{\Gamma(u_1)\Gamma(u_2)}{\Gamma(u_1 + u_2)},$$

$\Gamma(.)$ is the gamma function and $u = \langle u_1, u_2 \rangle$ is the parameters of the Dirichelet distribution. $u = \langle 1, 1 \rangle$ results in a uniform distribution over the possible values of α_1 and α_2. The same is assumed for c_b and i_b.

For the independent hypothesis, we assume that the two data e_a and e_b comes from independent distributions, with independent α and β, that is

$$\begin{aligned} P(e_a, e_b \mid H_{\text{indep}}) &= \int P(e_a, e_b \mid \alpha, \beta, H_{\text{indep}}) P(\alpha, \beta \mid H_{\text{indep}}) d\alpha d\beta, \\ &= \int P(e_a \mid \alpha, H_{\text{indep}}) P(\alpha \mid H_{\text{indep}}) d\alpha \int P(e_b \mid \beta, H_{\text{indep}}) P(\beta \mid H_{\text{indep}}) d\beta, \\ &= \frac{Z(u+e_a)}{Z(u)} \frac{Z(u+e_b)}{Z(u)}. \end{aligned}$$

For the same hypothesis, we assume that both e_a and e_b were sampled from the same (unknown) multinomial distribution:

$$\begin{aligned} P(e_a, e_b \mid H_{\text{same}}) &= \int P(e_a, e_b \mid \alpha, H_{\text{same}}) P(\alpha, \mid H_{\text{same}}) d\alpha, \\ &= \int P(e_a \mid \alpha, H_{\text{same}}) P(e_b \mid \alpha, H_{\text{same}}) P(\alpha \mid H_{\text{same}}) d\alpha, \\ &= \frac{Z(u+e_a+e_b)}{Z(u)}. \end{aligned}$$

Thus, the Bayes factor is:

$$BF = \frac{p(H_{\text{same}} \mid e_a, e_b)}{p(H_{\text{indep}} \mid e_a, e_b)} = \frac{Z(u) Z(u+e_a+e_b)}{Z(u+e_a) Z(u+e_b)}. \tag{6}$$

Appendix D. Variance of the excess loss

Figure 7 is a scatter plot of all excess losses for all procedures as a function of the kf5 error rate. The figure seems to indicate that the variance ~~is~~ does not increase as the error rate increases, as discussed in the text. Figure 8 computes the variance of the excess losses for all procedures for different kf5 error rates. The kf5 error rate was divided into 20 equal ranges, and the variance of the excess loss was computed for each range. The figure places the variance at the mean kf5 error rate for each range. Not all ranges are represented in the figure because 3 of them fall between the last two kf5 error rates.

Appendix E. The outer loop

The outer loop in the experiments is a 2-fold, that is we run the experiments in half of the data set and measure the error rate in the second half, then we change the train and test halves, and average the two measures. But kf2 is a high variance estimation. To evaluate the consequence of this choice, we ran an experiment where the outer loop is a 5xkf2, a 5-times repeated kf2, for all data sets with less than 400 points. The results are reported in Figure 9. The figure plots the mean and confidence intervals for the original experiment with a kf2 outer loop and the 5xkf2 experiment, for 42 data sets with less than 400 data. The results show that there was a reduction in the confidence interval for the large- and mid-training procedures, but no systematic difference in the mean excess loss. For the small training procedures there seems to be a small decrease in the mean excess loss, and usually

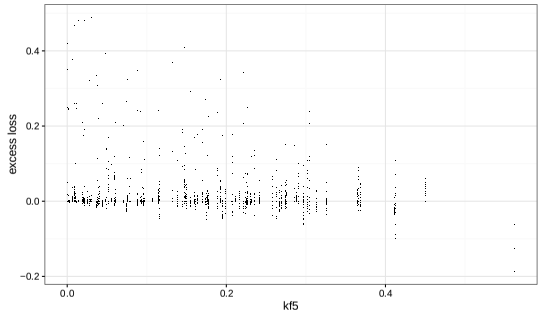


Figure 7: Scatter plot of all excess losses (for all procedures) as a function of the kf5 error rate.

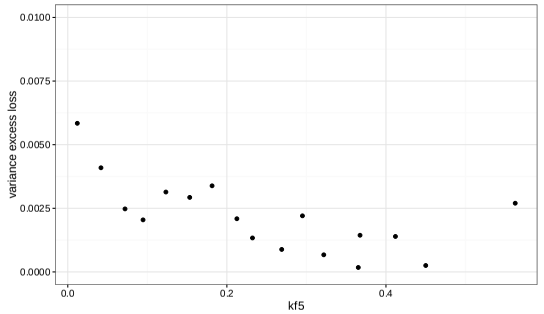


Figure 8: Variance of all excess losses for different kf5 error rates.

a small decrease in the confidence intervals, but not for all procedures. These differences should diminish as the data set increases in size. Thus had we used a repeated kf2 in the outer loop, the results would likely be similar, with maybe a small reduction on the sizes of the confidence intervals.

Appendix F. The full significance test comparison of all procedures

	kf5	kf5bis	kf2	kf3	2xkf5	kf10	5xboot	10xboot	20xboot	80/20	resub	invkf5	20/80	20/20	5x20/20	20/10	10/10
kf5bis	1.00																
kf2	1.00	1.00															
kf3	1.00	1.00	1.00														
2xkf5	1.00	1.00	1.00	1.00													
kf10	1.00	1.00	1.00	1.00	1.00												
5xboot	1.00	1.00	1.00	1.00	1.00	1.00											
10xboot	1.00	1.00	1.00	1.00	1.00	1.00	0.99										
20xboot	1.00	1.00	1.00	1.00	1.00	1.00	0.99	1.00									
80/20	0.45	0.12	0.04	0.04	0.06	0.02	0.58	0.01	0.01								
resub	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00							
invkf5	1.00	1.00	0.96	0.96	0.98	0.88	1.00	0.82	0.78	0.94	0.00						
20/80	0.81	0.38	0.17	0.17	0.24	0.09	0.90	0.06	0.05	1.00	0.00	1.00					
20/20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.91	0.00	0.02	0.61				
5x20/20	0.94	0.61	0.34	0.34	0.44	0.20	0.98	0.15	0.12	1.00	0.00	1.00	1.00	0.37			
20/10	0.01	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.00	1.00	0.00	0.14	0.95	1.00	0.82		
10/10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.55	0.00	0.00	0.21	1.00	0.10	1.00	
5x10/10	0.01	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.99	0.00	0.09	0.89	1.00	0.72	1.00	1.00

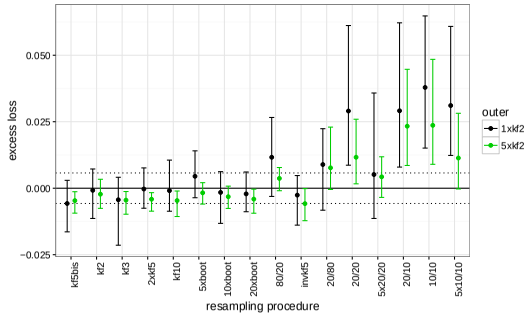


Figure 9: Comparison of using kf2 and 5xkf2 for the outer loop, for the 42 data sets with less than 400 data.

References

Davide Anguita, Andrea Boni, Sandro Ridella, Fabio Rivieccio, and Dario Sterpi. Theoretical and practical model selection methods for support vector classifiers. In *Support vector machines: theory and applications*, pages 159–179. Springer, 2005.

Davide Anguita, Alessandro Ghio, Luca Oneto, and Sandro Ridella. In-sample and out-of-sample model selection and error estimation for support vector machines. *IEEE Transactions on Neural Networks and Learning Systems*, 23(9):1390–1406, 2012. doi: 10.1109/TNNLS.2012.2202401.

David Barber. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(1):281–305, 2012.

Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Fifth Annual Workshop on Computational Learning Theory*, pages 144–152. ACM, 1992.

Léon Bottou and Chih-Jen Lin. Support vector machine solvers. In *Large Scale Kernel Machines*, pages 301–320, 2007.

Ulisses Braga-Neto and Edward Dougherty. Bolstered error estimation. *Pattern Recognition*, 37(6):1267–1281, 2004.

Gavin C. Cawley and Nicola L.C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11:2079–2107, 2010.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, 2011.

Olivier Chapelle, Vladmir Vapnik, Olivier Bousquet, and Sayan Mukherjee. Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1–3):131–159, January 2002.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3): 273–297, 1995.

Mark A. Davenport, Richard G. Baraniuk, and Clayton D. Scott. Tuning support vector machines for minimax and Neyman-Pearson classification. *IEEE Transactions on Pattern Analysis and Machine Learning*, 32(10):1888–1898, 2010.

Bruno F. de Souza, André C.P.L.F. de Carvalho, Rodrigo Calvo, and Renato P. Ishii. Multiclass SVM model selection using particle swarm optimization. In *International Conference on Hybrid Intelligent Systems*, page 31, 2006.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

Kaibo Duan, Sathiya Keerthi, and Aun Neow Poo. Evaluation of simple performance measures for tuning SVM hyperparameters. *Neurocomputing*, 51:41–59, 2003.

Bradley Efron. Estimating the error rate of a prediction rule: Improvement on cross-validation. *Journal of the American Statistical Association*, 78(382), 1983.

Bradley Efron. Better bootstrap confidence intervals. *Journal of the American Statistical Association*, 82(397):171–185, 1987.

Bradley Efron and Robert Tibshirani. Improvements on cross-validation: The 632+ bootstrap method. *Journal of the American Statistical Association*, 92(438):548–560, 1997.

Kai-Tai Fang, Dennis K.J. Lin, Peter Winker, and Yong Zhang. Uniform design: theory and application. *Technometrics*, 42(3):237–248, 2000.

Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, 15:3133–3181, 2014.

Frauke Friedrichs and Christian Igel. Evolutionary tuning of multiple SVM parameters. *Neurocomputing*, 64:107–117, 2005.

Terrence S. Furey, Nello Cristianini, Nigel Duffy, David W. Bednarski, Michl Schummer, and David Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, 2000.

Martin J. Gardner and Douglas G. Altman. Confidence intervals rather than p values: estimation rather than hypothesis testing. *BMJ*, 292(6522):746–750, 1986.

Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification, 2010. `https://www.cs.sfu.ca/people/Faculty/teaching/726/spring11/svmguide.pdf` accessed 1/2015.

Chien-Ming Huang, Yuh-Jye Lee, Dennis K.J. Lin, and Su-Yun Huang. Model selection for support vector machines via uniform design. *Computational Statistics & Data Analysis*, 52(1):335–346, 2007.

Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Machine Learning: ECML-98*, volume 1398 of *Lecture Notes in Computer Science*, pages 137–142. Springer, 1998.

Thorsten Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, 1999.

Thorsten Joachims. *The Maximum-Margin Approach to Learning Text Classifiers: Methods, Theory, and Algorithms*. PhD thesis, Department of Computer Science, University of Dortmund, 2000.

Robert E. Kass and Adrian E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995.

Sathiya Keerthi. Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms. *IEEE Transactions on Neural Networks*, 13(5):1225–1229, 2002.

Sathiya Keerthi and Chih-Jen Lin. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural computation*, 15(7):1667–1689, 2003.

Sathiya Keerthi, Vikas Sindhwani, and Olivier Chapelle. An efficient method for gradient-based adaptation of hyperparameters in SVM models. In B. Schölkopf, J. Platt, and T. Hofmann, editors, *Advances in Neural Information Processing Systems*, pages 673 – 680. MIT Press, 2007.

Roger E. Kirk. Practical significance: A concept whose time has come. *Educational and psychological measurement*, 56(5):746–759, 1996.

Tammo Krueger, Danny Panknin, and Mikio Braun. Fast cross-validation via sequential testing. *Journal of Machine Learning Research*, 16:1103–1155, 2015.

Max Kuhn. Futility analysis in the cross-validation of machine learning models. *arXiv preprint arXiv:1405.6974*, 2014.

Max Kuhn et al. *Package caret: Classification and Regression Training*, version 6.0-37 edition, 2014. `http://cran.r-project.org/web/packages/caret/caret.pdf`.

Shutao Li and Mingkui Tan. Tuning SVM parameters by using a hybrid CLPSO-BFGS algorithm. *Neurocomputing*, 73(10-12):2089–2096, 2010.

Moshe Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

Annette M. Molinaro, Richard Simon, and Ruth M. Pfeiffer. Prediction error estimation: a comparison of resampling methods. *Bioinformatics*, 21(15):3301–7, 2005.

John A. Nelder and Roger Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

Bruce Thompson. What future quantitative social science research could look like: Confidence intervals for effect sizes. *Educational Researcher*, 31(3):25–32, 2002.

Vladimir Vapnik. *Statistical learning theory*. Adaptive and learning systems for signal processing, communications and control series. Wiley, 1998.

Vladimir Vapnik and Olivier Chapelle. Bounds on error expectation for support vector machines. *Neural computation*, 12(9):2013–2036, 2000.

Grace Wahba. Advances in kernel methods. chapter Support Vector Machines, Reproducing Kernel Hilbert Spaces, and Randomized GACV, pages 69–88. MIT Press, Cambridge, MA, USA, 1999.

Chris Woolston. Psychology journal bans p values. *Nature*, 519(7541), 2015.

Journal of Machine Learning Research 18 (2017) 1-49 Submitted 5/16; Revised 1/17; Published 3/17

A Unified Formulation and Fast Accelerated Proximal Gradient Method for Classification

Naoki Ito naoki_ito@mist.i.u-tokyo.ac.jp
Department of Mathematical Informatics
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan

Akiko Takeda atakeda@ism.ac.jp
Department of Mathematical Analysis and Statistical Inference
The Institute of Statistical Mathematics
10-3 Midori-cho, Tachikawa, Tokyo 190-8562 Japan

Kim-Chuan Toh mattohkc@nus.edu.sg
Department of Mathematics
National University of Singapore
Blk S17, 10 Lower Kent Ridge Road, Singapore 119076, Singapore

Editor: Moritz Hardt

Abstract

Binary classification is the problem of predicting the class a given sample belongs to. To achieve a good prediction performance, it is important to find a suitable model for a given dataset. However, it is often time consuming and impractical for practitioners to try various classification models because each model employs a different formulation and algorithm. The difficulty can be mitigated if we have a unified formulation and an efficient universal algorithmic framework for various classification models to expedite the comparison of performance of different models for a given dataset. In this paper, we present a unified formulation of various classification models (including C-SVM, ℓ_2-SVM, ν-SVM, MM-FDA, MM-MPM, logistic regression, distance weighted discrimination) and develop a general optimization algorithm based on an accelerated proximal gradient (APG) method for the formulation. We design various techniques such as backtracking line search and adaptive restarting strategy in order to speed up the practical convergence of our method. We also give a theoretical convergence guarantee for the proposed fast APG algorithm. Numerical experiments show that our algorithm is stable and highly competitive to specialized algorithms designed for specific models (e.g., sequential minimal optimization (SMO) for SVM).

Keywords: restarted accelerated proximal gradient method, binary classification, minimum norm problem, vector projection computation, support vector machine

1. Introduction

Binary classification is one of the most important problems in machine learning. Among the wide variety of binary classification models which have been proposed to date, the most popular ones include support vector machines (SVMs) (Cortes and Vapnik, 1995; Schölkopf et al., 2000) and logistic regression (Cox, 1958). To achieve a good prediction performance, it is often important for the user to find a suitable model for a given dataset. However, the task of finding a suitable model is often time consuming and tedious as different classification models generally employ different formulations and algorithms. Moreover, the user might have to change not only the optimization algorithms but also solvers/software in order to solve different models. The goal of this paper is to present a unified formulation for various classification models and also to design a fast universal algorithmic framework for solving different models. By doing so, one can simplify and speed up the process of finding the best classification model for a given dataset. We can also compare various classification methods in terms of computation time and prediction performance in the same platform.

In this paper, we first propose a unified classification model which can express various models including C-SVM (Cortes and Vapnik, 1995), ν-SVM (Schölkopf et al., 2000), ℓ_2-SVM, logistic regression (Cox, 1958), MM-FDA, MM-MPM (Nath and Bhattacharyya, 2007), distance weighted discrimination (Marron et al., 2007). The unified model is first formulated as an unconstrained ℓ_2-regularized loss minimization problem, which is further transformed into the problem of minimizing a convex objective function (quadratic function, plus additional terms only for logistic regression) over a simple feasible region such as the intersection of a box and a hyperplane, truncated simplex, unit ball, and so on. Taking different loss functions (correspondingly different feasible regions in the transformed problem) will lead to different classification models. For example, when the feasible region is given by the intersection of a box and a hyperplane, the unified formulation coincides with the well-known C-SVM or logistic regression, and when the region is given by a truncated simplex, the problem is the same as the ν-SVM.

It is commonly acknowledged that there is "no free lunch" in supervised learning in the sense that no single algorithm can outperform all other algorithms in all cases. Therefore, there can be no single "best" software for binary classification. However, by taking advantage of the above-mentioned unified formulation, we can design an efficient algorithm which is applicable to the various existing models mentioned in the last paragraph. Our proposed algorithm is based on the accelerated proximal gradient (APG) method (Beck and Teboulle, 2009; Nesterov, 2005) and during the algorithm, only the procedure for computing the projection onto the associated feasible region differs for different models. In other words, by changing the computation of the projection, our algorithm can be applied to arbitrary classification models (i.e., arbitrary feasible region) without changing the optimization algorithmic framework. The great advantage of our algorithm is that most existing models have simple feasible regions, which make the computation of the projection easy and efficient.

The APG method is known to be one of the most efficient first-order methods theoretically. In order to make our APG based method practically efficient, we further employ various techniques such as backtracking line search (Beck and Teboulle, 2009; Scheinberg et al., 2014) and adaptive restarting strategies (O'Donoghue and Candès, 2015) to speed up the convergence of the algorithm. Our method incorporates several speed-up strategies while guaranteeing its convergence even for a general non-strongly convex function. Since our method is a further improvement of the standard APG method, we will call it as fast accelerated proximal gradient (FAPG) method. Our FAPG method improves several other restarting schemes (Nesterov, 2013; Lin and Xiao, 2015; Su et al., 2014) which have guaranteed convergence in the case when the objective function is strongly convex. To make our method even more efficient in practice, we simplify some steps in the implementation

©2017 Naoki Ito, Akiko Takeda, and Kim-Chuan Toh.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-274.html.

of our FAPG algorithm, though we can no longer ensure its theoretical convergence. To summarize, while our method has extensive generality, numerical experiments show that it performs stably and is highly competitive to specialized algorithms designed for specific classification models. Indeed, our method solved SVMs with a linear kernel substantially faster than LIBSVM (Chang and Lin, 2011) which implemented the SMO (Platt, 1998) and SeDuMi (Sturm, 1999) which implemented an interior-point method. Moreover, our FAPG method often run faster than the highly optimized LIBLINEAR (Fan et al., 2008) especially for large-scale datasets with feature dimension $n > 2000$. The FAPG method can be applied not only to the unified classification model but also to the general convex composite optimization problem such as ℓ_1-regularized classification models, which are solved faster than LIBLINEAR in most cases. It may be better to think of a stochastic variant of our method for further improvement and we leave it as a future research topic. We focus here on the deterministic one as the first trial to provide an efficient unified algorithm which is applicable to all well-known existing classification models.

The rest of this paper is organized as follows. Section 2 introduces some preliminary definitions and results on binary classification models and the APG method. Section 3 presents a unified formulation of binary classification models. In Section 4, we provide solution methods for the unified formulation. We develop efficient algorithms for computing projections which are used in the APG method. Then we design our FAPG method combined with various techniques to speed-up its practical convergence. The iteration complexity of $O\big((\log k/k)^2\big)$ of our algorithm is also established, where k is the iteration counter. Numerical experiments are presented in Section 5.

2. Preliminaries

In this section, we introduce some preliminary definitions on binary classification models and the APG method.

2.1 Binary Classification Models

Let $\mathcal{X} \subset \mathbb{R}^n$ be the input domain and $\{+1, -1\}$ be the set of the binary labels. Suppose that we have samples,

$$(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m) \in \mathcal{X} \times \{+1, -1\}.$$

Define $M := \{1, \ldots, m\}$, $M_+ := \{i \in M \mid y_i = +1\}$, and $M_- := \{i \in M \mid y_i = -1\}$. Let $m_+ = |M_+|$ and $m_- = |M_-|$.

We compute $(\boldsymbol{w}, b)$ for a decision function $h(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} - b$ using these samples and use $h(\boldsymbol{x})$ to predict the label for a new input point $\hat{\boldsymbol{x}} \in \mathcal{X}$. If $h(\hat{\boldsymbol{x}})$ is positive (resp. negative), then the label of $\hat{\boldsymbol{x}}$ is predicted to be $+1$ (resp. -1). Here we focus on linear learning models by using linear functions $h(\boldsymbol{x})$, but the discussions in this paper can be directly applied to non-linear kernel models (Schölkopf and Smola, 2002) using nonlinear maps $\phi(\boldsymbol{x})$ mapping $\boldsymbol{x}$ from the original space to a high dimensional space.

There are various binary classification models to compute $(\boldsymbol{w}, b)$ such as the support vector machines (SVMs), e.g. (Cortes and Vapnik, 1995; Schölkopf et al., 2000; Schölkopf and Smola, 2002), the margin maximized minimax probability machine (MM-MPM) (Nath and Bhattacharyya, 2007), the model based on Fisher's discriminant analysis (MM-FDA) (Bhattacharyya, 2004; Takeda et al., 2013), and the logistic regression (Cox, 1958).

Many binary classification models have the following formulation which consists of the sum of loss of each sample and a regularization term:

$$\min_{\boldsymbol{w},b} \sum_{i=1}^{m} \ell\big(y_i(\boldsymbol{w}^\top \boldsymbol{x}_i - b)\big) + \frac{1}{C}\|\boldsymbol{w}\|_p^p, \tag{1}$$

where $\ell : \mathbb{R} \to \mathbb{R}$ is a proper, closed, and convex function; $C > 0$ is a parameter; and $\|\cdot\|_p$ is the p-norm with $p \in [1, \infty]$.

2.2 Accelerated Proximal Gradient Method

In this section, we first introduce the accelerated proximal gradient (APG) method (Beck and Teboulle, 2009) which is designed for the following problem:

$$\min_{\boldsymbol{\alpha} \in \mathbb{R}^d} F(\boldsymbol{\alpha}) := f(\boldsymbol{\alpha}) + g(\boldsymbol{\alpha}). \tag{2}$$

Note that one can express a minimization problem constrained over a set S in the form of (2) by setting $g = \delta_S$, where

$$\delta_S(\boldsymbol{\alpha}) = \begin{cases} 0 & (\boldsymbol{\alpha} \in S) \\ +\infty & (\boldsymbol{\alpha} \notin S) \end{cases}$$

is the indicator function of the set S.

To apply the APG method to (2), we need to assume the following conditions:

1. $g : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ is a proper, closed, and convex function which is possibly nonsmooth. Its effective domain $\mathrm{dom}(g) = \{\boldsymbol{\alpha} \in \mathbb{R}^d \mid g(\boldsymbol{\alpha}) < +\infty\}$ is closed and convex.

2. $f : \mathbb{R}^d \to \mathbb{R}$ is a proper, closed, convex, and continuously differentiable function, and its gradient $\nabla f(\cdot)$ is Lipschitz continuous on $\mathbb{R}^d$,[1] i.e., there exists a constant $L > 0$ such that
$$\|\nabla f(\boldsymbol{\alpha}) - \nabla f(\boldsymbol{\beta})\|_2 \le L\|\boldsymbol{\alpha} - \boldsymbol{\beta}\|_2 \quad \forall \boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{R}^d. \tag{3}$$
The minimum value of such L is referred to as the Lipschitz constant L_f of $\nabla f(\cdot)$.

3. The problem (2) is solvable, i.e., the optimal value is finite and an optimal solution $\boldsymbol{\alpha}^*$ exists.

Let $L \ge L_f$. We define an approximate function $Q_L : \mathbb{R}^d \to \mathbb{R}$ of $f(\boldsymbol{\alpha})$ around $\boldsymbol{\beta}$ and a mapping $T_L(\boldsymbol{\alpha}) : \mathbb{R}^d \to \mathbb{R}^d$ as follows:

$$Q_L(\boldsymbol{\alpha}; \boldsymbol{\beta}) = f(\boldsymbol{\beta}) + \langle \nabla f(\boldsymbol{\beta}), \boldsymbol{\alpha} - \boldsymbol{\beta} \rangle + \frac{L}{2}\|\boldsymbol{\alpha} - \boldsymbol{\beta}\|_2^2 + g(\boldsymbol{\alpha})$$
$$T_L(\boldsymbol{\beta}) = \operatorname*{argmin}_{\boldsymbol{\alpha} \in \mathbb{R}^d} Q_L(\boldsymbol{\alpha}; \boldsymbol{\beta}).$$

1. It is sufficient if $\nabla f(\cdot)$ is Lipschitz continuous on a neighborhood of $\mathrm{dom}(g)$: the convex hull of $\mathrm{dom}(g) \cup \{\boldsymbol{\beta}^k \mid k = 1, 2, \ldots\}$, where $\boldsymbol{\beta}^k$ $(k = 1, 2, \ldots)$ are points generated at Step 3 of the APG method. Obviously, however, it is not possible to know the points a priori.

The basic proximal gradient (PG) method generates a sequence $\{\boldsymbol{\alpha}^k\}_{k=0}^{\infty}$ by

$$\boldsymbol{\alpha}^{k+1} = T_L(\boldsymbol{\alpha}^k) \;=\; \operatorname*{argmin}_{\boldsymbol{\alpha}\in\mathbb{R}^d} \Big\{ g(\boldsymbol{\alpha}) + \frac{L}{2}\Big\|\boldsymbol{\alpha} - \Big(\boldsymbol{\alpha}^k - \frac{1}{L}\nabla f(\boldsymbol{\alpha}^k)\Big)\Big\|_2^2 \Big\}$$
$$= \operatorname{prox}_{g,L}\Big(\boldsymbol{\alpha}^k - \frac{1}{L}\nabla f(\boldsymbol{\alpha}^k)\Big),$$

where $\operatorname{prox}_{g,L}(\bar{\boldsymbol{\alpha}}) := \operatorname{argmin}_{\boldsymbol{\alpha}\in\mathbb{R}^d}\Big\{g(\boldsymbol{\alpha}) + \frac{L}{2}\|\boldsymbol{\alpha}-\bar{\boldsymbol{\alpha}}\|_2^2\Big\}$ is the proximal operator of g. If $g = \delta_S$, then $\operatorname{prox}_{g,L_k} = P_S$, where $P_S(\bar{\boldsymbol{\alpha}}) = \operatorname{argmin}_{\boldsymbol{\alpha}\in S}\|\boldsymbol{\alpha}-\bar{\boldsymbol{\alpha}}\|_2$ is the orthogonal projection of $\bar{\boldsymbol{\alpha}}$ onto the set S. In this case, the above PG method coincides with the gradient projection method. If $g(\cdot) = \|\cdot\|_1$, then $\big(\operatorname{prox}_{g,L}(\boldsymbol{\alpha})\big)_i = \operatorname{sign}(\alpha_i)\max\{0, |\alpha_i| - L\}$ $(i = 1, 2, \ldots, d)$ which is known as the soft-thresholding operator. Other analytical computations of the proximal operators of various g can be found in (Parikh and Boyd, 2014, Section 6).

It is known that the PG method has the iteration complexity such that $F(\boldsymbol{\alpha}^k) - F(\boldsymbol{\alpha}^*) \le O(1/k)$, where $\boldsymbol{\alpha}^*$ is an optimal solution of (2). The APG method (Beck and Teboulle, 2009), which is also known as *FISTA*, is an acceleration of the PG method. It generates two sequences $\{\boldsymbol{\beta}^k\}_{k=1}^{\infty}$ and $\{\boldsymbol{\alpha}^k\}_{k=0}^{\infty}$. For an arbitrary initial point $\boldsymbol{\beta}^1 = \boldsymbol{\alpha}^0 \in \mathbb{R}^d$ and $t_1 = 1$, the APG method solves (2) through the following steps $(k = 1, 2, \ldots)$:

Accelerated Proximal Gradient Method

Step 1. Compute
$$\boldsymbol{\alpha}^k = T_L(\boldsymbol{\beta}^k) = \operatorname{prox}_{g,L}\Big(\boldsymbol{\beta}^k - \frac{1}{L}\nabla f(\boldsymbol{\beta}^k)\Big).$$

Step 2. Compute $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$.

Step 3. Compute $\boldsymbol{\beta}^{k+1} = \boldsymbol{\alpha}^k + \frac{t_k - 1}{t_{k+1}}(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1})$.

For the APG method, the iteration complexity result such that $F(\boldsymbol{\alpha}^k) - F(\boldsymbol{\alpha}^*) \le O(1/k^2)$ is known (see (Beck and Teboulle, 2009, Theorem 4.4)). The second term $\frac{t_k-1}{t_{k+1}}(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1})$ in Step 3 can be seen as the *momentum* of the sequence $\{\boldsymbol{\alpha}^k\}_{k=0}^{\infty}$. It enlarges the moving distance of the sequences $\{\boldsymbol{\alpha}\}_{k=0}^{\infty}, \{\boldsymbol{\beta}\}_{k=1}^{\infty}$ which may lead them closer to the optimum $\boldsymbol{\alpha}^*$ more quickly. While various other APG methods (e.g, (Nesterov, 2013; Monteiro et al., 2016; Su et al., 2014)) are proposed, the above APG (namely, FISTA) is used in many applications because it is simpler to implement.

It is known that $\boldsymbol{\alpha}^*$ is an optimal solution of (2) if and only if $\boldsymbol{\alpha}^* = T_L(\boldsymbol{\alpha}^*)$. More specifically, the necessary and sufficient optimality condition for $\boldsymbol{\alpha}^*$ to be an optimal solution of (2) is

$$\exists \gamma_\alpha \in \partial g(\boldsymbol{\alpha}^*) \quad \text{s.t.} \quad \langle \nabla f(\boldsymbol{\alpha}^*) + \gamma_\alpha,\ \boldsymbol{\alpha} - \boldsymbol{\alpha}^* \rangle \ge 0, \quad \forall \boldsymbol{\alpha} \in \mathbb{R}^d. \tag{4}$$

On the other hand, from the definition of $T_L(\boldsymbol{\beta})$, we have

$$\exists \gamma_\beta \in \partial g(T_L(\boldsymbol{\beta})) \quad \text{s.t.} \quad \langle \nabla f(T_L(\boldsymbol{\beta})) + L\big(T_L(\boldsymbol{\beta}) - \boldsymbol{\beta}\big) + \gamma_\beta,\ \boldsymbol{\alpha} - T_L(\boldsymbol{\beta}) \rangle \ge 0, \quad \forall \boldsymbol{\alpha} \in \mathbb{R}^d, \tag{5}$$

for any $\boldsymbol{\beta} \in \mathbb{R}^d$. The term $L\big(T_L(\boldsymbol{\beta}) - \boldsymbol{\beta}\big)$ in (5) can be seen as the residual of the optimality condition (4). Thus it would be a natural criterion to terminate the APG if $L\|T_L(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k\|_2 < \epsilon$ with a small constant $\epsilon > 0$.

Despite having a strong iteration complexity result, the APG method may still not be efficient enough for practical purpose. In the following, we describe several well-known strategies to make the APG method practically efficient.

2.2.1 Backtracking Strategy

We assume that L is greater than or equals to the Lipschitz constant L_f of $\nabla f(\boldsymbol{\alpha})$. However it is advantageous to use a smaller value for L whenever possible since the constant L plays the role of a step size as in a gradient descent method; fixing L to be the Lipschitz constant L_f is usually too conservative (see Table 5 in Section 5). Thus we adopt the following backtracking strategy (Beck and Teboulle, 2009) after Step 1 with arbitrary given constants $\eta_u > 1$ and $L_0 > 0$:

'bt': While
$$F(\boldsymbol{\alpha}^k) > Q_{L_k}(\boldsymbol{\alpha}^k; \boldsymbol{\beta}^k), \tag{6}$$
update $L_k \leftarrow \eta_u L_k$ and $\boldsymbol{\alpha}^k \leftarrow T_{L_k}(\boldsymbol{\beta}^k)$. Set $L_{k+1} \leftarrow L_k$.

The inequality (6) in 'bt' ensures that the complexity result $F(\boldsymbol{\alpha}^k) - F(\boldsymbol{\alpha}^*) \le O(1/k^2)$ still holds. We note that the inequality $F(\boldsymbol{\alpha}^k) \le Q_{L_k}(\boldsymbol{\alpha}^k; \boldsymbol{\beta}^k)$ is satisfied if $L_k \ge L_f$, i.e., it is a weaker condition than (3).

2.2.2 Decreasing Strategy for L_k

Beck and Teboulle (2009) designed the backtracking strategy 'bt' so that the values of L_k is non-decreasing. In fact, the convergence analysis of (Beck and Teboulle, 2009) requires the value of L_k to be non-decreasing. However, it is advantageous to decrease the value of L_k whenever possible since the constant $\frac{1}{L_k}$ gives a larger step size.

To allow L_k to decrease, Scheinberg et al. (2014) modifies the APG method so that $\{t_k\}_{k=1}^{\infty}$ satisfies $t_k/L_k \ge t_{k+1}(t_{k+1} - 1)/L_{k+1}$ $(\forall k \ge 1)$ and the sequences $\{\boldsymbol{\alpha}^k\}_{k=0}^{\infty}$ and $\{\boldsymbol{\beta}^k\}_{k=1}^{\infty}$ are generated along with $\{t_k\}_{k=1}^{\infty}$. To be specific, let us introduce the following simple step to decrease the value of L_k, where $\eta_d > 1$ is a given constant.

'dec': Set $L_{k+1} \leftarrow L_k/\eta_d$.

The modified APG of (Scheinberg et al., 2014) can be described as in Algorithm 1. It differs from the original APG in that the updates of t_k and $\boldsymbol{\beta}^k$ are added to 'bt' and Step 2 is modified. The convergence of Algorithm 1 is shown as follows.

Proposition 1 (from Scheinberg et al., 2014) *Let S^* be the set of optimal solutions of (2). For any $\boldsymbol{\alpha}^* \in S^*$, the sequence $\{\boldsymbol{\alpha}^k\}_{k=0}^{\infty}$ generated by Algorithm 1 satisfies the following inequality:*

$$F(\boldsymbol{\alpha}^k) - F(\boldsymbol{\alpha}^*) \le \frac{2\eta_u L_f \|\boldsymbol{\alpha}^0 - \boldsymbol{\alpha}^*\|_2^2}{k^2}, \quad \forall k \ge 1.$$

Algorithm 1 An Accelerated Proximal Gradient Method with Non-Monotonic Backtracking

Input: f, ∇f, g, $\mathrm{prox}_{g,L}$, $\epsilon > 0$, $L_1 = L_0 > 0$, $\eta_u > 1$, $\eta_d > 1$, $k_{max} > 0$, $\boldsymbol{\beta}^1 = \boldsymbol{\alpha}^0$
Output: $\boldsymbol{\alpha}^k$
Initialize: $t_1 \leftarrow 1,\ t_0 \leftarrow 0$
for $k = 1, \ldots, k_{max}$ **do**
  $\boldsymbol{\alpha}^k \leftarrow T_{L_k}(\boldsymbol{\beta}^k) = \mathrm{prox}_{g,L_k}\left(\boldsymbol{\beta}^k - \frac{1}{L_k}\nabla f(\boldsymbol{\beta}^k)\right)$ # Step 1
  while $F(\boldsymbol{\alpha}^k) > Q_{L_k}(\boldsymbol{\alpha}^k; \boldsymbol{\beta}^k)$ **do**
    $L_k \leftarrow \eta_u L_k$ # 'bt'
    $t_k \leftarrow \frac{1+\sqrt{1+4(L_k/L_{k-1})t_{k-1}^2}}{2}$
    $\boldsymbol{\beta}^k \leftarrow \boldsymbol{\alpha}^{k-1} + \frac{t_{k-1}-1}{t_k}(\boldsymbol{\alpha}^{k-1} - \boldsymbol{\alpha}^{k-2})$
    $\boldsymbol{\alpha}^k \leftarrow T_{L_k}(\boldsymbol{\beta}^k) = \mathrm{prox}_{g,L_k}\left(\boldsymbol{\beta}^k - \frac{1}{L_k}\nabla f(\boldsymbol{\beta}^k)\right)$
  end while
  if $\|L_k(T_{L_k}(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k)\| < \epsilon$ **then**
    break
  end if
  $L_{k+1} \leftarrow L_k/\eta_d$ # 'dec'
  $t_{k+1} \leftarrow \frac{1+\sqrt{1+4(L_{k+1}/L_k)t_k^2}}{2}$ # Step 2'
  $\boldsymbol{\beta}^{k+1} \leftarrow \boldsymbol{\alpha}^k + \frac{t_k-1}{t_{k+1}}(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1})$ # Step 3
end for

2.2.3 Restarting Strategy

The value $\frac{t_k-1}{t_{k+1}} \in [0,1)$ in Step 3 determines the amount of *momentum* in $\frac{t_k-1}{t_{k+1}}(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1})$. When the value $\frac{t_k-1}{t_{k+1}}$ is close to 1, i.e., the momentum is high, the sequences of solutions $\{\boldsymbol{\alpha}^k\}_{k=0}^{\infty}$ and $\{\boldsymbol{\beta}^k\}_{k=1}^{\infty}$ would overshoot and oscillate around the optimal solution $\boldsymbol{\alpha}^*$. In order to avoid the oscillation and further speed up the convergence, O'Donoghue and Candès (2015) introduced an adaptive restarting strategy:

're': If $\nabla f(\boldsymbol{\beta}^k)^\top(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}) + g(\boldsymbol{\alpha}^k) - g(\boldsymbol{\alpha}^{k-1}) > 0$,
then update $t_{k+1} \leftarrow 1$, $t_k \leftarrow 0$, $\boldsymbol{\beta}^{k+1} \leftarrow \boldsymbol{\alpha}^{k-1}$, and $\boldsymbol{\alpha}^k \leftarrow \boldsymbol{\alpha}^{k-1}$.

Roughly, the APG method resets the momentum back to zero and restarts from the previous point $\boldsymbol{\alpha}^{k-1}$ if the direction of motion $\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}$ seems to cause the (approximated) objective value to increase, which may be a sign of overshooting. Note that the computational cost of 're' is inexpensive since $\nabla f(\boldsymbol{\beta}^k)$ has already been computed at Step 1. O'Donoghue and Candès (2015) also provided a heuristic convergence analysis for their restarting scheme ('re') when f is a strongly convex (i.e., there exists a constant $\mu > 0$ such that $f(\boldsymbol{\alpha}) - \frac{\mu}{2}\|\boldsymbol{\alpha} - \boldsymbol{\alpha}^*\|_2^2$ is convex) quadratic function and $g = 0$. However, the convergence of 're' for a general strongly convex objective function is unknown (not to mention for the general non-strongly convex function in our problem).

There are several other restarting schemes (Nesterov, 2013; Lin and Xiao, 2015; Su et al., 2014). They have guaranteed convergence in the case that f is strongly convex.

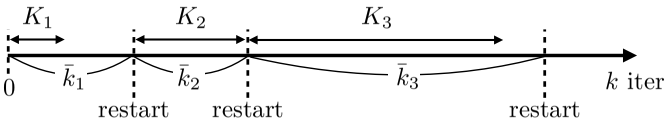


Figure 1: Illustration of Maintaining Top-Speed Strategy ('mt'). A prohibition period of restart for K_i iteration are imposed after the i-th restart occurs. If the condition $\nabla f(\boldsymbol{\beta}^k)^\top(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}) < 0$ is satisfied after the period passed, then the $(i+1)$-th restart occurs. The next prohibition period is doubled, i.e., $K_{i+1} = 2K_i$. $\bar{k}_i (\geq K_i)$ denotes the number of iteration taken between the $(i-1)$-th and i-th restart, which will be used to convergence analysis in Section 4.2.

Nesterov (2013) proposed a restarting method, which has asymptotic linear convergence if g is strongly convex. Lin and Xiao (2015) showed that a similar restart technique can achieve the same convergence rate as Nesterov's scheme if f is strongly convex. Su et al. (2014) modeled the APG method as a second-order ordinary differential equation (ODE). They provided a *speed restarting* framework that ensures a linear convergence of the ODE with respect to time in the case that f is strongly convex and $g = 0$. To the best of our knowledge, none of the restarting schemes have convergence guarantees for a general non-strongly convex function.

2.2.4 Maintaining Top-Speed Strategy

The restarting strategy cancels out high momentum and prevents overshooting. In the neighborhood of an optimum, however, maintaining high momentum may be effective rather than restarting APG (see Figures 6 and 8 in Section 5), because large overshooting may not occur. Thus we put a prohibition period of restart for K_i iteration after the i-th restart occurs, where K_i increases as $K_i = 2K_{i-1}$ $(i = 2, 3, \ldots)$ and $K_1 \geq 2$. See Figure 1 for illustration. Precisely, letting $k = k_{re}$ be the iteration count at which the last restart occurs, we introduce the following step:

'mt': If $k \leq k_{re} + K_i$, then skip 're'. If restart occurs, then update $k_{re} \leftarrow k$, $K_{i+1} \leftarrow 2K_i$, and $i \leftarrow i + 1$.

While a similar strategy is taken in the experiment of (Monteiro et al., 2016), its convergence analysis is not provided. One of our contributions in this paper is to show that in fact the modification 'mt' of the restarting strategy can ensure the convergence rate of $O\big((\log k/k)^2\big)$ under a mild assumption. We will elaborate it in Section 4.2.

3. New Unified Formulation

In this paper, we focus on the following ℓ_2-regularized loss minimization problem:

$$\min_{\boldsymbol{w},b}\ \mathcal{L}(\widetilde{A}^\top \boldsymbol{w} - \boldsymbol{a}b) + \frac{1}{2C}\|\boldsymbol{w}\|_2^2, \tag{7}$$

where $\mathcal{L} : \mathbb{R}^m \to \mathbb{R}$ is a proper, closed, and convex function; $\widetilde{A} \in \mathbb{R}^{n\times l}$, $\boldsymbol{a} \in \mathbb{R}^l$, and $C > 0$ is a parameter. We will later show examples such that

- $l = m, \quad \widetilde{A} = \widetilde{X} := [y_1\boldsymbol{x}_1, y_2\boldsymbol{x}_2, \ldots, y_m\boldsymbol{x}_m], \quad \boldsymbol{a} = \boldsymbol{y}, \quad$ and
- $l = n, \quad \widetilde{A} = I, \quad \boldsymbol{a} = \boldsymbol{0}.$

Our algorithm to be described later can be applied to a more general loss-regularized model with $p \in (1, \infty)$:

$$\min_{\boldsymbol{w},b} \left\{ \mathcal{L}(\widetilde{A}^\top \boldsymbol{w} - \boldsymbol{a}b) \mid \|\boldsymbol{w}\|_p \le \lambda \right\}, \tag{8}$$

where $\lambda > 0$ is a parameter. We note that the condition $p \in (1, \infty)$ is required for our unified classification algorithm because the smoothness of the dual norm of $\|\cdot\|_p$ is required to apply the APG method to the dual problem of (8). However, if $p \in \{1, \infty\}$ and $\mathcal{L}$ is smooth, the APG method can be applied to the primal problem (8). See Appendix D for ℓ_1-regularized problems, i.e., the case of $p = 1$.

3.1 Dual Formulation

The dual problems of (7) is given by

$$\min_{\boldsymbol{\alpha}} \left\{ \mathcal{L}^*(-\boldsymbol{\alpha}) + \frac{C}{2}\|\widetilde{A}\boldsymbol{\alpha}\|_2^2 \mid \boldsymbol{\alpha}^\top \boldsymbol{a} = 0 \right\}, \tag{9}$$

where $\mathcal{L}^*(\boldsymbol{\alpha}) = \sup_{\boldsymbol{z}}\{\boldsymbol{\alpha}^\top \boldsymbol{z} - \mathcal{L}(\boldsymbol{z})\}$ is the convex conjugate of $\mathcal{L}$.

The dual problem of (8) is

$$\min_{\boldsymbol{\alpha}} \ \{ \mathcal{L}^*(-\boldsymbol{\alpha}) + \lambda\|\widetilde{A}\boldsymbol{\alpha}\|_p^* \mid \boldsymbol{\alpha}^\top \boldsymbol{a} = 0 \}, \tag{10}$$

where $\|\boldsymbol{z}\|_p^* = \sup_{\|\boldsymbol{w}\|_p \le 1}\{\boldsymbol{z}^\top \boldsymbol{w}\}$ is the dual norm of $\|\cdot\|_p$. It is known that $\|\cdot\|_p^* = \|\cdot\|_q$, where $q = p/(p-1)$. The problem (10) can be seen as the Fenchel dual of (8), whose derivation is shown in Appendix A. We note that the norms $\|\cdot\|_p$ and $\|\cdot\|_q^*$ are smooth for $p \in (1, \infty)$.

As shown later, in all classification models, $\mathcal{L}^*$ is the sum of a smooth function and an indicator function of a simple set S for which the projection P_S can be computed efficiently. Thus we can apply the APG method as an acceleration of the gradient projection method to the dual problems (9) and (10). By this construction, we can ensure the fast convergence rate of $F(\boldsymbol{\alpha}^k) - F(\boldsymbol{\alpha}^*) \le O(1/k^2)$ for various classification models in a unified way. In other word, the APG method can obtain a solution with desired accuracy $\epsilon > 0$ in $O(1/\sqrt{\epsilon})$ iteration.

There is an existing work (Zhou et al., 2010) which applies the APG method to the primal C-SVM, i.e., (7) with the hinge loss function. However, their method requires $O(1/\epsilon)$ iteration to obtain a solution with the desired accuracy $\epsilon > 0$ because it approximates the hinge loss by a smoothed function and making the approximation tighter requires extra iterations (Nesterov, 2005).

3.2 Relation to Existing Binary Classification Models

In this section, we show some specific examples of $\mathcal{L}$ and $\mathcal{L}^*$.

3.2.1 C-SVM

Let $\widetilde{A} = \widetilde{X}$ and $\boldsymbol{a} = \boldsymbol{y}$ in (7). Let $\ell(z) = \max\{0, 1 - z\}$ be the hinge loss and $\mathcal{L}(\boldsymbol{z}) = \sum_{i=1}^m \ell(z_i)$. Then the primal problem (7) is known as the standard C-SVM (Cortes and Vapnik, 1995). Since

$$\mathcal{L}^*(-\boldsymbol{\alpha}) = \sum_{i=1}^m \ell^*(-\alpha_i), \quad \text{where} \quad \ell^*(-\alpha) = \begin{cases} -\alpha & \text{if } \alpha \in [0,1] \\ +\infty & \text{otherwise,} \end{cases}$$

the dual problem (9) can be reduced to

$$\min_{\boldsymbol{\alpha}} \ \frac{C}{2}\|\widetilde{X}\boldsymbol{\alpha}\|_2^2 - \boldsymbol{\alpha}^\top \boldsymbol{e} + \delta_{S_C}(\boldsymbol{\alpha}), \tag{11}$$

where $S_C = \{\boldsymbol{\alpha} \in \mathbb{R}^m \mid \boldsymbol{\alpha}^\top \boldsymbol{y} = 0, \ \boldsymbol{0} \le \boldsymbol{\alpha} \le \boldsymbol{e}\}$.

3.2.2 ℓ_2-SVM

Let $\widetilde{A} = \widetilde{X}$ and $\boldsymbol{a} = \boldsymbol{y}$ in (7). Let $\ell(z) = (\max\{0, 1 - z\})^2$ be the truncated squared loss and $\mathcal{L}(\boldsymbol{z}) = \sum_{i=1}^m \ell(z_i)$. Then the primal problem (7) is known as the ℓ_2-SVM. Since

$$\mathcal{L}^*(-\boldsymbol{\alpha}) = \sum_{i=1}^m \ell^*(-\alpha_i), \quad \text{where} \quad \ell^*(-\alpha) = \begin{cases} \frac{\alpha^2}{4} & \text{if } \alpha \ge 0 \\ +\infty & \text{otherwise,} \end{cases}$$

the dual problem (9) can be reduced to

$$\min_{\boldsymbol{\alpha}} \ \frac{C}{2}\|\widetilde{X}\boldsymbol{\alpha}\|_2^2 + \frac{\|\boldsymbol{\alpha}\|_2^2}{4} + \delta_{S_{\ell_2}}(\boldsymbol{\alpha}), \tag{12}$$

where $S_{\ell_2} = \{\boldsymbol{\alpha} \in \mathbb{R}^m \mid \boldsymbol{\alpha}^\top \boldsymbol{y} = 0, \ \boldsymbol{\alpha} \ge \boldsymbol{0}\}$.

3.2.3 Logistic regression

Let $\widetilde{A} = \widetilde{X}$ and $\boldsymbol{a} = \boldsymbol{y}$ in (7). Let $\ell(z) = \log(1 + \exp(-z))$ be the logistic loss and $\mathcal{L}(\boldsymbol{z}) = \sum_{i=1}^m \ell(z_i)$. Then the primal problem (7) is the logistic regression (Cox, 1958). Since

$$\mathcal{L}^*(-\boldsymbol{\alpha}) = \sum_{i=1}^m \ell^*(-\alpha_i), \quad \text{where} \quad \ell^*(-\alpha) = \begin{cases} \alpha\log(\alpha) + (1-\alpha)\log(1-\alpha) & (0 < \alpha < 1) \\ 0 & (\alpha = 0, 1) \\ +\infty & \text{otherwise,} \end{cases}$$

the dual problem (9) can be reduced to

$$\min_{\boldsymbol{\alpha}} \ \frac{C}{2}\|\widetilde{X}\boldsymbol{\alpha}\|_2^2 + \sum_{i:\alpha_i > 0} \alpha_i \log(\alpha_i) + \sum_{i:\alpha_i < 1} (1-\alpha_i)\log(1-\alpha_i) + \delta_{S_{\mathrm{LR}}}(\boldsymbol{\alpha}), \tag{13}$$

where $S_{\mathrm{LR}} = \{\boldsymbol{\alpha} \in \mathbb{R}^m \mid \boldsymbol{\alpha}^\top \boldsymbol{y} = 0, \ \boldsymbol{0} \le \boldsymbol{\alpha} \le \boldsymbol{e}\}$.

We should note that the gradient of the second and third terms in (13) is not Lipschitz continuous on S_{LR} and not defined at $\alpha_i = 0, 1$ $(i \in M)$. However, since it is known that

its optimal solution $\boldsymbol{\alpha}^*$ satisfies $\mathbf{0} < \boldsymbol{\alpha}^* < \boldsymbol{e}$ (Yu et al., 2011), we can instead solve (13) by replacing $S_{\rm LR}$ by $S_{\rm LR}(\xi) := \{\boldsymbol{\alpha} \in \mathbb{R}^m \mid \boldsymbol{\alpha}^\top \boldsymbol{y} = 0,\ \xi \boldsymbol{e} \le \boldsymbol{\alpha} \le (1-\xi)\boldsymbol{e}\}$, where $\xi > 0$ is a small constant. The second and third terms in (13) are differentiable over $S_{\rm LR}(\xi)$. We will later show numerically that the resulting decision hyperplane is robust to a small deviation of ξ (see Table 6 in Section 5).

3.2.4 ν-SVM

Let $\widetilde{A} = \widetilde{X}$ and $\boldsymbol{a} = \boldsymbol{y}$ in (7). Let

$$\mathcal{L}(\boldsymbol{z}) = \min_{\rho} \Big\{ -\rho + \frac{1}{m\nu} \sum_{i=1}^{m} \max\{\rho - z_i, 0\} \Big\},$$

where $\nu \in (0, 1]$ is a given parameter. The function $\mathcal{L}(\boldsymbol{z})$, known as the conditional value-at-risk (CVaR) (Rockafellar and Uryasev, 2000), is the expected value of the upper ν-tail distribution. Gotoh and Takeda (2005) pointed out that minimizing CVaR is equivalent to ν-SVM (Schölkopf et al., 2000), which is also known to be equivalent to C-SVM.

Since

$$\mathcal{L}^*(-\boldsymbol{\alpha}) = \begin{cases} 0 & \Big(\boldsymbol{e}^\top \boldsymbol{\alpha} = 1 \text{ and } \boldsymbol{\alpha} \in \Big[0, \frac{1}{m\nu}\Big]^m\Big) \\ +\infty & \text{otherwise,} \end{cases}$$

the dual problem is given by

$$\min_{\boldsymbol{\alpha}} \ \frac{C}{2} \|\widetilde{X}\boldsymbol{\alpha}\|_2^2 + \delta_{S_\nu}(\boldsymbol{\alpha}), \tag{14}$$

where $S_\nu = \{\boldsymbol{\alpha} \in \mathbb{R}^m \mid \boldsymbol{\alpha}^\top \boldsymbol{y} = 0,\ \boldsymbol{\alpha}^\top \boldsymbol{e} = 1,\ \mathbf{0} \le \boldsymbol{\alpha} \le \frac{1}{m\nu}\boldsymbol{e}\}$ is the intersection of the hyperplane $\{\boldsymbol{\alpha} \mid \boldsymbol{\alpha}^\top \boldsymbol{y} = 0\}$ and the truncated simplex $\{\boldsymbol{\alpha} \mid \boldsymbol{\alpha}^\top \boldsymbol{e} = 1,\ \mathbf{0} \le \boldsymbol{\alpha} \le \frac{1}{m\nu}\boldsymbol{e}\}$. Note that any positive value for C does not change the optimal solution of (14), and therefore, we can set $C = 1$. There exists a valid range $(\nu_{min}, \nu_{max}] \subseteq (0, 1]$ for ν, where ν_{min} is the infimum of $\nu > 0$ such that $\widetilde{X}\boldsymbol{\alpha}^* \neq \mathbf{0}$, and $\nu_{max} := \frac{2\max\{m_+, m_-\}}{m}$ is the maximum of $\nu \le 1$ for which $S_\nu \neq \phi$. Since the parameter ν takes a value in the finite range $(\nu_{min}, \nu_{max}]$, choosing the parameter value for ν-SVM is often easier than that for C-SVM.

3.2.5 Distance Weighted Discrimination (DWD)

Let $\widetilde{A} = \widetilde{X}$, $\boldsymbol{a} = \boldsymbol{y}$, $\lambda = 1$ and $p = 2$ in (8). For a positive parameter ν, let $\ell(z)$ be the function defined by

$$\begin{aligned} \ell(z) &= \min_{\rho} \Big\{ \frac{1}{\rho} + \frac{1}{m\nu}(\rho - z) \mid \rho \ge z,\ \rho \ge \sqrt{m\nu} \Big\} \\ &= \begin{cases} \frac{1}{z} & \text{if } z \ge \sqrt{m\nu} \\ \frac{2\sqrt{m\nu} - z}{m\nu} & \text{otherwise} \end{cases}. \end{aligned}$$

Consider $\mathcal{L}(\boldsymbol{z}) = \sum_{i=1}^m \ell(z_i)$. This loss function is used in the distance weighted discrimination (DWD) (Marron et al., 2007) which is proposed as an alternative to ν-SVM. Since we have

$$\mathcal{L}^*(-\boldsymbol{\alpha}) = \sum_{i=1}^m \ell^*(-\alpha_i), \quad \text{where} \quad \ell^*(-\alpha) = \begin{cases} -2\sqrt{\alpha} & \big(0 \le \alpha \le \frac{1}{m\nu}\big) \\ +\infty & \text{otherwise,} \end{cases}$$

the dual problem (9) can be equivalently reduced to

$$\min_{\boldsymbol{\alpha}} \Big\{ \|\widetilde{X}\boldsymbol{\alpha}\|_2 - 2\sum_{i=1}^m \sqrt{\alpha_i} + \delta_{S_{\rm DWD}}(\boldsymbol{\alpha}) \Big\}, \tag{15}$$

where $S_{\rm DWD} = \big\{\boldsymbol{\alpha} \in \mathbb{R}^m \mid \boldsymbol{\alpha}^\top \boldsymbol{y} = 0,\ \mathbf{0} \le \boldsymbol{\alpha} \le \frac{1}{m\nu}\boldsymbol{e}\big\}$.

As in the case of the logistic regression, the second term in (15) is not differentiable at $\alpha_i = 0$ ($i \in M$). However, since it is known that there exists an optimal solution $\boldsymbol{\alpha}^*$ such that $\alpha_i^* > 0$ ($i \in M$), we can solve (15) by replacing $S_{\rm DWD}$ by $S_{\rm DWD}(\xi) = \big\{\boldsymbol{\alpha} \in \mathbb{R}^m \mid \boldsymbol{\alpha}^\top \boldsymbol{y} = 0,\ \xi\boldsymbol{e} \le \boldsymbol{\alpha} \le \frac{1}{m\nu}\boldsymbol{e}\big\}$ where $\xi > 0$ is a small constant. The second term in (15) is differentiable over $S_{\rm DWD}(\xi)$.

3.2.6 Extended Fisher's discriminant analysis (MM-FDA)

The well-known Fisher's discriminant analysis (FDA) uses a decision function to maximize the ratio of the variance between the classes to the variance within the classes. Let $\bar{\boldsymbol{x}}_o$ and Σ_o, $o \in \{+, -\}$, be the mean vectors and the positive definite covariance matrices of $\boldsymbol{x}_i$, $i \in M_o$, respectively. Then FDA is formulated as follows:

$$\max_{\boldsymbol{w}} \ \frac{\big(\boldsymbol{w}^\top(\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-)\big)^2}{\boldsymbol{w}^\top(\Sigma_+ + \Sigma_-)\boldsymbol{w}}.$$

Its optimal solution is given by

$$\boldsymbol{w}^* = (\Sigma_+ + \Sigma_-)^{-1}(\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-).$$

Let $\widetilde{A} = I$ and $\boldsymbol{a} = \mathbf{0}$ in (7). Here we consider the mean-standard deviation type of risk corresponding to FDA:

$$\mathcal{L}(\boldsymbol{w}) = -\boldsymbol{w}^\top(\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-) + \kappa\sqrt{\boldsymbol{w}^\top(\Sigma_+ + \Sigma_-)\boldsymbol{w}},$$

where $\kappa > 0$ is a parameter. Since

$$\begin{aligned} \mathcal{L}^*(-\boldsymbol{\alpha}) &= \sup_{\boldsymbol{w}} \Big\{ -\boldsymbol{\alpha}^\top \boldsymbol{w} + \boldsymbol{w}^\top(\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-) - \kappa\sqrt{\boldsymbol{w}^\top(\Sigma_+ + \Sigma_-)\boldsymbol{w}} \Big\} \\ &= \sup_{\boldsymbol{w}} \min_{\|\boldsymbol{u}\|_2 \le \kappa} \Big\{ -\boldsymbol{\alpha}^\top \boldsymbol{w} + \boldsymbol{w}^\top(\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-) + \boldsymbol{w}^\top(\Sigma_+ + \Sigma_-)^{1/2}\boldsymbol{u} \Big\} \\ &= \min_{\boldsymbol{u}} \big\{ \delta_{S_{\rm FDA}}(\boldsymbol{u}) \mid \boldsymbol{\alpha} = (\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-) + (\Sigma_+ + \Sigma_-)^{1/2}\boldsymbol{u} \big\}, \end{aligned}$$

where $S_{\rm FDA} = \big\{\boldsymbol{u} \in \mathbb{R}^n \mid \|\boldsymbol{u}\|_2 \le \kappa\big\}$, the dual problem (9) can be reduced to

$$\min_{\boldsymbol{\alpha}, \boldsymbol{u}} \Big\{ \delta_{S_{\rm FDA}}(\boldsymbol{u}) + \frac{C}{2}\|\boldsymbol{\alpha}\|_2^2 \mid \boldsymbol{\alpha} = (\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-) + (\Sigma_+ + \Sigma_-)^{1/2}\boldsymbol{u} \Big\},$$

which is equivalent to

$$\min_{\boldsymbol{u}} \left\{ \delta_{S_{\mathrm{FDA}}}(\boldsymbol{u}) + \left\| (\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-) + (\Sigma_+ + \Sigma_-)^{1/2}\boldsymbol{u} \right\|_2^2 \right\}. \tag{16}$$

Takeda et al. (2013) showed that

$$\kappa_{\max} := \left| \begin{array}{ll} \inf_{\boldsymbol{u},\kappa} & \kappa \\ \text{s.t.} & \min_{\boldsymbol{u}} \left\{ \left\| (\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-) + (\Sigma_+ + \Sigma_-)^{1/2}\boldsymbol{u} \right\|_2 + \delta_{S_{\mathrm{FDA}}}(\boldsymbol{u}) \right\} = 0 \end{array} \right.$$

is equivalent to FDA. Hence (16) can be seen as an extension of FDA. We will refer it as MM-FDA in this paper.

3.2.7 Maximum Margin Minimax Probability Machine (MM-MPM)

Let $\widetilde{A} = I$ and $\boldsymbol{a} = \boldsymbol{0}$ in (7). We consider a class-wise mean-standard deviation type of risk:

$$\mathcal{L}(\boldsymbol{w}) = \Big(-\boldsymbol{w}^\top \bar{\boldsymbol{x}}_+ + \kappa \sqrt{\boldsymbol{w}^\top \Sigma_+ \boldsymbol{w}} \Big) - \Big(-\boldsymbol{w}^\top \bar{\boldsymbol{x}}_- + \kappa \sqrt{\boldsymbol{w}^\top \Sigma_- \boldsymbol{w}} \Big).$$

Similar to MM-FDA, we have

$$\begin{aligned} \mathcal{L}^*(\boldsymbol{w}) &= \sup_{\boldsymbol{w}} \Big\{ -\boldsymbol{\alpha}^\top \boldsymbol{w} + \Big(\boldsymbol{w}^\top \bar{\boldsymbol{x}}_+ - \kappa \sqrt{\boldsymbol{w}^\top \Sigma_+ \boldsymbol{w}} \Big) - \Big(\boldsymbol{w}^\top \bar{\boldsymbol{x}}_- - \kappa \sqrt{\boldsymbol{w}^\top \Sigma_- \boldsymbol{w}} \Big) \Big\} \\ &= \sup_{\boldsymbol{w}} \min_{\substack{\|\boldsymbol{u}_+\|_2 \le \kappa \\ \|\boldsymbol{u}_-\|_2 \le \kappa}} \Big\{ -\boldsymbol{\alpha}^\top \boldsymbol{w} + \big(\boldsymbol{w}^\top \bar{\boldsymbol{x}}_+ + \boldsymbol{w}^\top \Sigma_+^{1/2} \boldsymbol{u}_+ \big) - \big(\boldsymbol{w}^\top \bar{\boldsymbol{x}}_- + \boldsymbol{w}^\top \Sigma_-^{1/2} \boldsymbol{u}_- \big) \Big\} \\ &= \min_{\boldsymbol{u}_+, \boldsymbol{u}_-} \left\{ \delta_{S_{\mathrm{MPM}}}(\boldsymbol{u}_+, \boldsymbol{u}_-) \mid \boldsymbol{\alpha} = \big(\bar{\boldsymbol{x}}_+ + \Sigma_+^{1/2} \boldsymbol{u}_+ \big) - \big(\bar{\boldsymbol{x}}_- + \Sigma_-^{1/2} \boldsymbol{u}_- \big) \right\}, \end{aligned}$$

where $S_{\mathrm{MPM}} = \big\{ (\boldsymbol{u}_+, \boldsymbol{u}_-) \in \mathbb{R}^n \times \mathbb{R}^n \mid \|\boldsymbol{u}_+\|_2 \le \kappa,\ \|\boldsymbol{u}_-\|_2 \le \kappa \big\}$. Thus the dual problem (9) can be reduced to

$$\min_{\boldsymbol{\alpha}, \boldsymbol{u}_+, \boldsymbol{u}_-} \Big\{ \delta_{S_{\mathrm{MPM}}}(\boldsymbol{u}_+, \boldsymbol{u}_-) + \frac{C}{2} \|\boldsymbol{\alpha}\|_2^2 \mid \boldsymbol{\alpha} = \big(\bar{\boldsymbol{x}}_+ + \Sigma_+^{1/2} \boldsymbol{u}_+ \big) - \big(\bar{\boldsymbol{x}}_- + \Sigma_-^{1/2} \boldsymbol{u}_- \big) \Big\},$$

which is equivalent to

$$\min_{\boldsymbol{u}_+, \boldsymbol{u}_-} \left\{ \delta_{S_{\mathrm{MPM}}}(\boldsymbol{u}_+, \boldsymbol{u}_-) + \left\| \big(\bar{\boldsymbol{x}}_+ + \Sigma_+^{1/2} \boldsymbol{u}_+ \big) - \big(\bar{\boldsymbol{x}}_- + \Sigma_-^{1/2} \boldsymbol{u}_- \big) \right\|_2^2 \right\}. \tag{17}$$

The last problem (17) is equivalent to the dual of the maximum margin minimax probability machine (MM-MPM) (Nath and Bhattacharyya, 2007).

4. A Fast APG method for Unified Binary Classification

In this section, we first provide an efficient vector projection computation in order to apply the APG method to the unified formulation of the binary classification models. After that, we develop a fast APG (FAPG) method and show its convergence.

4.1 Vector Projection Computation

When applying the APG method to a constrained optimization problem over S, the projection P_S onto S appears at Step 1. We need to change the computation of the projection P_S depending on the definition of S. In this section, we provide efficient projection computations for S_C, S_{ℓ_2}, $S_{\mathrm{LR}}(\xi)$, S_ν, $S_{\mathrm{DWD}}(\xi)$, S_{MPM}, and S_{FDA}.

4.1.1 Bisection Method for Projection

Many models shown in Section 3 have a linear equality and box constraints. Here we consider the set $S = \{\boldsymbol{\alpha} \in \mathbb{R}^m \mid \boldsymbol{\alpha}^\top \boldsymbol{e} = r,\ l \le \alpha_i \le u\ (\forall i \in M)\}$ and provide an efficient algorithm for computing the projection $P_S(\bar{\boldsymbol{\alpha}})$:

$$\min_{\boldsymbol{\alpha}} \Big\{ \frac{1}{2} \|\boldsymbol{\alpha} - \bar{\boldsymbol{\alpha}}\|_2^2 \mid \boldsymbol{\alpha}^\top \boldsymbol{e} = r, \quad l \le \alpha_i \le u, \quad (i \in M) \Big\}. \tag{18}$$

We assume that (18) is feasible. One way to solve (18) is to use breakpoint search algorithms (e.g., (Helgason et al., 1980; Kiwiel, 2008; Duchi et al., 2008)). They are exact algorithms and have linear time complexity of $O(m)$. Helgason et al. (1980); Kiwiel (2008) developed the breakpoint search algorithms for the continuous quadratic knapsack problem (CQKP) which involves the projection (18) as a special case. By integrating the breakpoint search algorithm and the red-black tree data structure, Duchi et al. (2008) developed an efficient update algorithm of the gradient projection method when the gradient is sparse.

In this paper, we provide a numerical algorithm for (18). Although its worst case complexity is $O\Big(m \log\big(\frac{\bar{\alpha}_{\max} - \bar{\alpha}_{\min}}{\epsilon'}\big)\Big)$, where $\bar{\alpha}_{\max} = \max\{\bar{\alpha}_i \mid i \in M\}$ and $\bar{\alpha}_{\min} = \min\{\bar{\alpha}_i \mid i \in M\}$, it often requires less time to compute a solution $\hat{\boldsymbol{\alpha}}$ such that $\|\hat{\boldsymbol{\alpha}} - P_S(\bar{\boldsymbol{\alpha}})\|_\infty < \epsilon' = u$ in practice, where $u \approx 2.22 \times 10^{-16}$ is the IEEE 754 double precision. Note that the error u can occur even when using the breakpoint search algorithm because u is the supremum of the relative error due to rounding in the floating point number system. Thus, it is sufficient to choose $\epsilon' = u$ as the stopping tolerance for our algorithm in practice. [2]

The difference between the algorithms is that the breakpoint search algorithms use binary search, whereas our algorithm uses bisection to solve an equation. It is known that the projection (18) can be reduced to finding the root of an one dimensional monotone function.

Lemma 2 (e.g. Helgason et al., 1980) *Suppose that the problem* (18) *is feasible. Let*

$$\alpha_i(\theta) = \min\Big\{ \max\{l, \bar{\alpha}_i - \theta\}, u \Big\}, \quad i \in M$$

and let

$$h(\theta) = \sum_{i \in M} \alpha_i(\theta).$$

The following statements hold:

2. Another practical way is to decrease ϵ', i.e., to improve the accuracy of the projection progressively, as APG iterates. The APG method with the inexact computation also share the same iteration complexity $O(1/k^2)$ as the exact counterpart, as shown in (Jiang et al., 2012).

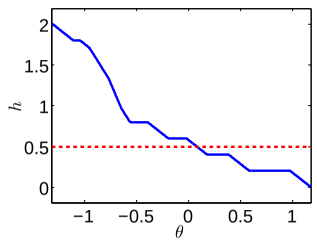


Figure 2: Illustration of the function $h(\theta) := \sum_{i\in M} \alpha_i(\theta)$. The function h is piecewise linear.

1. *$h(\theta)$ is a continuous, non-increasing, and piecewise linear function which has breakpoints at $\bar{\alpha}_i - u$ and $\bar{\alpha}_i - l$ $(i \in M)$.*

2. *There exists $\theta^* \in (\bar{\alpha}_{\min} - \frac{r}{m}, \bar{\alpha}_{\max} - \frac{r}{m})$ such that $h(\theta^*) = r$.*

3. *Let $\hat{\alpha}_i = \alpha_i(\theta^*)$, $i \in M$. Then $\hat{\boldsymbol{\alpha}}$ is an optimal solution of* (18).

An example of $h(\theta)$ is illustrated in Figure 2. To solve $h(\theta) = r$, the breakpoint search uses the binary search to find two adjacent breakpoints that contain a solution θ^* between them. Then the exact solution θ^* can be obtained by linear interpolation of the two breakpoints.

Instead of binary search, we employ the bisection method to solve the equation $h(\theta) = r$.

Bisection Algorithm for Projection

Step 1. Set $\theta^u = \bar{\alpha}_{\max} - \frac{r}{m}$ and $\theta^l = \bar{\alpha}_{\min} - \frac{r}{m}$.

Step 2. Set $\hat{\theta} = (\theta^u + \theta^l)/2$

Step 3. Compute $h(\hat{\theta})$.

Step 4. If $h(\hat{\theta}) = r$, then terminate with $\hat{\theta}^* = \hat{\theta}$.
Else if $h(\hat{\theta}) < r$, then set $\theta^u = \hat{\theta}$.
Else if $h(\hat{\theta}) > r$, then set $\theta^l = \hat{\theta}$.

Step 5. If $|\theta^u - \theta^l| < \epsilon'$, then terminate with $\hat{\theta}^* = \hat{\theta}$. Else, go to Step 2.

All steps require at most $O(m)$ operations and the maximum number of iteration is $\lceil \log(\frac{\bar{\alpha}_{\max} - \bar{\alpha}_{\min}}{\epsilon'}) \rceil$. Thus, the computational complexity of the bisection algorithm is at most $O\big(m \log(\frac{\bar{\alpha}_{\max} - \bar{\alpha}_{\min}}{\epsilon'})\big)$.

As the bisection method narrows the interval (θ^l, θ^u), some of the terms $\alpha_i(\theta)$, $i \in M$, can be set to l, $\bar{\alpha}_i - \theta$, or u for $\theta \in (\theta^l, \theta^u)$. By exploiting the property of $\alpha_i(\theta)$, we can refine the computation of $h(\hat{\theta})$ in Step 3 by avoiding recalculation of certain sums. Moreover, we can often obtain an exact solution. Let us divide M into the following three disjoint sets

for given θ^l and θ^u satisfying $\theta^l < \theta^u$:

$$\begin{aligned} U &:= \{i \in M \mid \bar{\alpha}_i - \theta^u \geq u \quad (\text{i.e., } \alpha_i(\theta) = u, \quad \forall \theta \in [\theta^l, \theta^u])\} \\ L &:= \{i \in M \mid \bar{\alpha}_i - \theta^l \leq l \quad (\text{i.e., } \alpha_i(\theta) = l, \quad \forall \theta \in [\theta^l, \theta^u])\} \\ I &:= M \backslash (U \cup L). \end{aligned}$$

If $\bar{\alpha}_i - \theta^u \geq l$ and $\bar{\alpha}_i - \theta^l \leq u$ $\forall i \in I$, then

$$\theta = \Big(|U|u + |L|l + \sum_{i\in I} \alpha_i - r\Big)/|I| \tag{19}$$

is an exact solution. If $I \neq \phi$, then we also divide I into the following three disjoint sets for given $\hat{\theta} \in (\theta^l, \theta^u)$:

$$\begin{aligned} I^U &= \{i \in I \mid \bar{\alpha}_i - \hat{\theta} \geq u, \quad (\text{i.e., } \alpha_i(\theta) = u \quad \forall \theta \in [\theta^l, \hat{\theta}])\} \\ I^L &= \{i \in I \mid \bar{\alpha}_i - \hat{\theta} \leq l, \quad (\text{i.e., } \alpha_i(\theta) = l \quad \forall \theta \in [\hat{\theta}, \theta^u])\} \\ I^C &= I \backslash (I^U \cup I^L) \end{aligned}$$

Then we have

$$h(\hat{\theta}) = \sum_{i\in M} \alpha_i(\hat{\theta}) = \underbrace{|U|u}_{s_u} + \underbrace{|I^U|u}_{\Delta s_u} + \underbrace{|L|l}_{s_l} + \underbrace{|I^L|l}_{\Delta s_l} + \underbrace{\sum_{i\in I^C} \alpha_i}_{s_c} - |I^C|\hat{\theta}.$$

By storing the value of each terms, we can reduce the number of additions. The resulting algorithm can be described as Algorithm 2.

Compared to the existing breakpoint search algorithms, our bisection method can avoid the computation of breakpoints and the comparison between them. In many cases, our algorithm can obtain the exact solution by (19). It is often faster than the breakpoint search algorithms in practice (see Table 2 in Section 5).

Remark 3 *Mimicking (Kiwiel, 2008), we can divide the set M more finely and reduce the recalculation of certain sums as shown in Appendix B. However, Algorithm 2 allows for simpler implementation and runs faster on our computer systems.*

In the followings, we use Algorithm 2 to compute the projection onto S_C, S_{ℓ_2}, $S_{\mathrm{LR}}(\xi)$, $S_{\mathrm{DWD}}(\xi)$ and S_ν.

4.1.2 Projection for C-SVM, Logistic Regression, ℓ_2-SVM, DWD

The projection for C-SVM, logistic regression, ℓ_2-SVM, and DWD can be formulated as follows:

$$\min_{\boldsymbol{\alpha}} \Big\{ \frac{1}{2}\|\boldsymbol{\alpha} - \bar{\boldsymbol{\alpha}}\|_2^2 \mid \boldsymbol{y}^\top \boldsymbol{\alpha} = 0, \quad l \leq \alpha_i \leq u \quad (i \in M) \Big\}, \tag{20}$$

where $(l, u) = (0, 1)$ for C-SVM, $(l, u) = (0, \infty)$ for ℓ_2-SVM, $(l, u) = (\xi, 1 - \xi)$ for logistic regression, and $(l, u) = \big(\xi, \frac{1}{m\nu}\big)$ for DWD. By transforming the variable $\boldsymbol{\alpha}$ and the vector $\bar{\boldsymbol{\alpha}}$ to

$$\beta_i = \begin{cases} \alpha_i & \text{if } y_i = 1 \\ -\alpha_i + l + u & \text{otherwise} \end{cases} \quad \text{and} \quad \bar{\beta}_i = \begin{cases} \bar{\alpha}_i & \text{if } y_i = 1 \\ -\bar{\alpha}_i + l + u & \text{otherwise,} \end{cases}$$

Algorithm 2 Bisection Algorithm for (18)

INPUT: $\bar{\boldsymbol{\alpha}},\ r,\ \boldsymbol{l},\ \boldsymbol{u},\ \epsilon' > 0$ OUTPUT: $\boldsymbol{\alpha}$

INITIALIZE: $I \leftarrow M,\ s_u \leftarrow s_l \leftarrow s_c \leftarrow 0,\ \theta^u \leftarrow \bar{\alpha}_{\max} - \frac{r}{m},\ \theta^l \leftarrow \bar{\alpha}_{\min} - \frac{r}{m}$ # Step 1

while $|\theta^u - \theta^l| > \epsilon'$ **do**

  $\hat{\theta} \leftarrow \frac{\theta^u + \theta^l}{2}$ # Step 2

  $\hat{u} \leftarrow u + \hat{\theta}, \quad \hat{l} \leftarrow l + \hat{\theta}$

  $I^U \leftarrow \{i \in I \mid \bar{\alpha}_i \geq \hat{u}\}, \quad I^L \leftarrow \{i \in I \mid \bar{\alpha}_i \leq \hat{l}\}$ # Step 3

  $I^C \leftarrow I \backslash (I^U \cup I^L)$

  $\Delta s_u \leftarrow |I^U| u, \quad \Delta s_l \leftarrow |I^L| l, \quad s_c \leftarrow \sum_{i \in I^C} \alpha_i$

  $val \leftarrow s_u + \Delta s_u + s_l + \Delta s_l + s_c - |I^C| \hat{\theta}$

  if $val < r$ **then** # Step 4

    $\theta^u \leftarrow \hat{\theta}, \quad I \leftarrow I^C \cup I^L$

    $s_u \leftarrow s_u + \Delta s_u$

  else if $val > r$ **then**

    $\theta^l \leftarrow \hat{\theta}, \quad I \leftarrow I^C \cup I^U$

    $s_l \leftarrow s_l + \Delta s_l$

  else

    break

  end if

  if $\bar{\alpha}_i - \theta^u \geq l$ and $\bar{\alpha}_i - \theta^l \leq u \quad \forall i \in I$ **then**

    $\hat{\theta} \leftarrow (s_u + s_l + s_c - r)/|I|$

    break

  end if

end while

$\alpha_i \leftarrow \alpha_i(\hat{\theta}),\ \forall i \in M$ # Step 5

respectively, the problem (20) can be reduced to

$$\min_{\boldsymbol{\beta}} \Big\{ \frac{1}{2} \|\boldsymbol{\beta} - \bar{\boldsymbol{\beta}}\|_2^2 \mid \boldsymbol{e}^\top \boldsymbol{\beta} = (l+u) m_-, \quad l \leq \beta_i \leq u \quad (i \in M) \Big\}.$$

The last problem can be solved by Algorithm 2.

4.1.3 Projection for ν-SVM

In applying the APG method to ν-SVM, we shall be concerned with the projection $P_{S_\nu}(\bar{\boldsymbol{\alpha}})$:

$$\min_{\boldsymbol{\alpha}} \Big\{ \frac{1}{2} \|\boldsymbol{\alpha} - \bar{\boldsymbol{\alpha}}\|_2^2 \ \Big|\ \boldsymbol{\alpha}^\top \boldsymbol{y} = 0,\ \boldsymbol{\alpha}^\top \boldsymbol{e} = 1,\ \boldsymbol{0} \leq \boldsymbol{\alpha} \leq \frac{1}{m\nu} \boldsymbol{e} \Big\}. \tag{21}$$

Let $\boldsymbol{\alpha}_+$ and $\boldsymbol{\alpha}_-$ be the subvectors of $\boldsymbol{\alpha}$ corresponding to the label $+1$ and -1, respectively. Let $\boldsymbol{e}_+$ and $\boldsymbol{e}_-$ be subvectors of $\boldsymbol{e}$ with size m_+ and m_-. From the fact that the conditions $\boldsymbol{\alpha}^\top \boldsymbol{y} = 0$ and $\boldsymbol{\alpha}^\top \boldsymbol{e} = 1$ are equivalent to $\boldsymbol{\alpha}_o^\top \boldsymbol{e}_o = \frac{1}{2}$ $(o \in \{+,-\})$, the problem (21) can be decomposed into the following two problems:

$$\min_{\boldsymbol{\alpha}_o} \Big\{ \frac{1}{2} \|\boldsymbol{\alpha}_o - \bar{\boldsymbol{\alpha}}_o\|_2^2 \ \Big|\ \boldsymbol{e}_o^\top \boldsymbol{\alpha} = \frac{1}{2},\ \boldsymbol{0} \leq \boldsymbol{\alpha}_o \leq \frac{1}{m\nu} \boldsymbol{e}_o \Big\},\ o \in \{+,-\}. \tag{22}$$

Then, Algorithm 2 can be applied to solve (22).

4.1.4 Projection for MM-MPM and MM-FDA

The projection $P_{S_{\rm FDA}}(\bar{\boldsymbol{\alpha}})$ of $\bar{\boldsymbol{\alpha}}$ onto $S_{\rm FDA} = \big\{ \boldsymbol{\alpha} \in \mathbb{R}^n \mid \|\boldsymbol{\alpha}\|_2 \leq \kappa \big\}$ is easy to compute; We have

$$P_{S_{\rm FDA}}(\bar{\boldsymbol{\alpha}}) = \min \Big\{ 1, \frac{\kappa}{\|\bar{\boldsymbol{\alpha}}\|_2} \Big\} \bar{\boldsymbol{\alpha}}.$$

Similarly, the projection $P_{S_{\rm MPM}}(\bar{\boldsymbol{\alpha}}_+, \bar{\boldsymbol{\alpha}}_-)$ for MM-MPM can be computed as follows:

$$P_{S_{\rm MPM}}(\bar{\boldsymbol{\alpha}}_+, \bar{\boldsymbol{\alpha}}_-) = \Big(\min \Big\{ 1, \frac{\kappa}{\|\bar{\boldsymbol{\alpha}}_+\|_2} \Big\} \bar{\boldsymbol{\alpha}}_+, \min \Big\{ 1, \frac{\kappa}{\|\bar{\boldsymbol{\alpha}}_-\|_2} \Big\} \bar{\boldsymbol{\alpha}}_- \Big).$$

4.2 Convergence Analysis of a Fast APG Method with Speed-up Strategies

We have shown several strategies to speed-up the APG method in Section 2. While the convergence proof has been shown for the backtracking 'bt' and decreasing 'dec' strategy, the convergence for the restart 're' and maintaining top-speed 'mt' strategy is unknown. In this section, we develop a fast APG (FAPG) method by combining all these techniques and show its convergence.

4.2.1 A Fast APG with Stabilization

In addition to the techniques in Section 2, we also employ the following practical stabilization strategy, where it does *not* affect our convergence analysis.

Decreasing L ('dec') and restarting strategies ('re') shown in Section 2 are effective to speed-up the practical convergence of the APG method, especially in early iterations and near the optimal solution, respectively (see Figures 6 and 10 in Section 5). However, they have opposing effects. Decreasing L_k enlarges the stepsize $\frac{1}{L_k}$, which extends $\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}$.

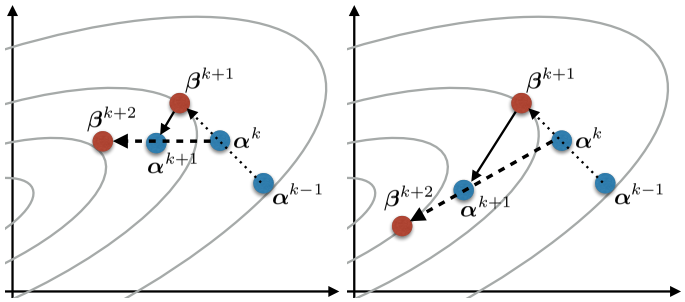


Figure 3: Effects of the value of L_k to the momentum $\frac{t_k-1}{t_{k+1}}(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1})$ (left: L_k is large, right: L_k is small). A smaller value of L_k gives a larger step size at Step 1 as illustrated by the solid lines. This results in high momentum at Step 3 as illustrated by the dashed lines.

This inherently induces high momentum (as illustrated in Figure 3). On the other hand, the restarting strategy cancels out high momentum. Hence combining them triggers the restart frequently and makes the APG method unstable. In order to avoid the instability, it would be necessary to reduce the η_d (rate of decreasing L_k) near the optimum. The restart would occur when the sequences approach the optimum. Therefore we take the following strategy to reduce the value of η_d with a constant $\delta \in (0, 1)$:

'st': Update $\eta_d \leftarrow \delta \cdot \eta_d + (1 - \delta) \cdot 1$ when the restart occurs.

Algorithm 3 is our FAPG method which combines these strategies. The difference from Algorithm 1 is the "if block" after Step 3.

4.2.2 Convergence Analysis

Let $\bar{k}_i$ denotes the number of iteration taken between the $(i-1)$-th and the i-th restart (see Figure 1 for illustration). In the followings, we denote $\boldsymbol{\alpha}^k$ as $\boldsymbol{\alpha}^{(j,k_{j+1})}$ if $k = \sum_{i=1}^{j} \bar{k}_i + k_{j+1}$ ($\bar{k}_{j+1} \geq k_{j+1} \geq 0$). In other word, $\boldsymbol{\alpha}^{(i,k_{i+1})}$ denotes a point obtained at the k_{i+1}-th iteration counting from the i-th restart. Note that $\boldsymbol{\alpha}^{(i+1,0)} = \boldsymbol{\alpha}^{(i,\bar{k}_{i+1}-1)}$ ($\forall i \geq 0$). We will frequently switch the notations $\boldsymbol{\alpha}^k$ and $\boldsymbol{\alpha}^{(i,k_{i+1})}$ for notational convenience.

The following lemma is useful to evaluate the function F.

Lemma 4 (from Beck and Teboulle, 2009) *If $F(T_L(\boldsymbol{\beta})) \leq Q_L(T_L(\boldsymbol{\beta}); \boldsymbol{\beta})$, then*

$$F(T_L(\boldsymbol{\beta})) - F(\boldsymbol{\alpha}) \leq \frac{L}{2}\{\|\boldsymbol{\beta} - \boldsymbol{\alpha}\|_2^2 - \|T_L(\boldsymbol{\beta}) - \boldsymbol{\alpha}\|_2^2\}, \quad \forall \boldsymbol{\alpha} \in \mathbb{R}^d.$$

Using Lemma 4, we obtain the following key lemma.

Algorithm 3 A Fast Accelerated Proximal Gradient (FAPG) Method with Speeding-Up Strategies

Input: f, ∇f, g, $\mathrm{prox}_{g,L}$, $\epsilon > 0$, $L_1 = L_0 > 0$, $\eta_u > 1$, $\eta_d > 1$, $k_{max} > 0$, $\boldsymbol{\beta}^1 = \boldsymbol{\alpha}^0 = \boldsymbol{\alpha}^{-1}$, $K_1 \geq 2$, $\delta \in (0, 1)$
Output: $\boldsymbol{\alpha}^k$
Initialize: $t_1 \leftarrow 1$, $t_0 \leftarrow 0$, $i \leftarrow 1$, $k_{re} \leftarrow 0$
for $k = 1, \ldots, k_{max}$ **do**
  $\boldsymbol{\alpha}^k \leftarrow T_{L_k}(\boldsymbol{\beta}^k) = \mathrm{prox}_{g,L_k}\left(\boldsymbol{\beta}^k - \frac{1}{L_k}\nabla f(\boldsymbol{\beta}^k)\right)$ # Step 1
  while $F(\boldsymbol{\alpha}^k) > Q_{L_k}(\boldsymbol{\alpha}^k; \boldsymbol{\beta}^k)$ **do**
    $L_k \leftarrow \eta_u L_k$ # 'bt'
    $t_k \leftarrow \frac{1+\sqrt{1+4(L_k/L_{k-1})t_{k-1}^2}}{2}$
    $\boldsymbol{\beta}^k \leftarrow \boldsymbol{\alpha}^{k-1} + \frac{t_{k-1}-1}{t_k}(\boldsymbol{\alpha}^{k-1} - \boldsymbol{\alpha}^{k-2})$
    $\boldsymbol{\alpha}^k \leftarrow T_{L_k}(\boldsymbol{\beta}^k) = \mathrm{prox}_{g,L_k}\left(\boldsymbol{\beta}^k - \frac{1}{L_k}\nabla f(\boldsymbol{\beta}^k)\right)$
  end while
  if $\|L_k(T_{L_k}(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k)\| < \epsilon$ **then**
    break
  end if
  $L_{k+1} \leftarrow L_k/\eta_d$ # 'dec'
  $t_{k+1} \leftarrow \frac{1+\sqrt{1+4(L_{k+1}/L_k)t_k^2}}{2}$ # Step 2'
  $\boldsymbol{\beta}^{k+1} \leftarrow \boldsymbol{\alpha}^k + \frac{t_k-1}{t_{k+1}}(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1})$ # Step 3
  if $k > k_{re} + K_i$ and $\langle \nabla f(\boldsymbol{\beta}^k), \boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}\rangle + g(\boldsymbol{\alpha}^k) - g(\boldsymbol{\alpha}^{k-1}) > 0$ **then**
    $k_{re} \leftarrow k$, $K_{i+1} \leftarrow 2K_i$, $i \leftarrow i+1$ # 'mt'
    $\eta_d \leftarrow \delta \cdot \eta_d + (1-\delta) \cdot 1$ # 'st'
    $t_{k+1} \leftarrow 1$, $t_k \leftarrow 0$, $\boldsymbol{\beta}^{k+1} \leftarrow \boldsymbol{\alpha}^{k-1}$, $\boldsymbol{\alpha}^k \leftarrow \boldsymbol{\alpha}^{k-1}$ # 're'
  end if
end for

Lemma 5 *Let the sequence* $\{\boldsymbol{\alpha}^k\}_{k=1}^{\infty} (\equiv \{\boldsymbol{\alpha}^{(i,k_{i+1})}\})$ *be generated by Algorithm 3. The following inequality holds:*

$$F(\boldsymbol{\alpha}^k) \leq F(\boldsymbol{\alpha}^0), \quad \forall k \geq 1.$$

Moreover,

$$F(\boldsymbol{\alpha}^{(i,k_{i+1})}) \leq F(\boldsymbol{\alpha}^{(i,0)}), \quad \forall i \geq 0 \quad \textit{and} \quad \forall k_{i+1} \in \{0, 1, 2, \ldots, \bar{k}_{i+1}\},$$

and

$$F(\boldsymbol{\alpha}^{(i+1,0)}) \leq F(\boldsymbol{\alpha}^{(i,0)}), \quad \forall i \geq 0.$$

Proof First, assume that the restart does not occur (i.e., the steps 're', 'st', and 'mt' are not executed) until the k-th iteration. From Lemma 4, we have

$$F(\boldsymbol{\alpha}^1) - F(\boldsymbol{\alpha}^0) \leq \frac{L_1}{2}\{\|\boldsymbol{\beta}^1 - \boldsymbol{\alpha}^0\|_2^2 - \|\boldsymbol{\alpha}^1 - \boldsymbol{\alpha}^0\|_2^2\} = -\frac{L_1}{2}\|\boldsymbol{\alpha}^1 - \boldsymbol{\alpha}^0\|_2^2 \leq 0. \tag{23}$$

For all $n = 1, 2, \ldots$, we also have

$$\begin{aligned} F(\boldsymbol{\alpha}^{n+1}) - F(\boldsymbol{\alpha}^n) &\leq \frac{L_{n+1}}{2}\{\|\boldsymbol{\beta}^{n+1} - \boldsymbol{\alpha}^n\|_2^2 - \|\boldsymbol{\alpha}^{n+1} - \boldsymbol{\alpha}^n\|_2^2\} \\ &= \frac{L_{n+1}}{2}\Big\{\Big(\frac{t_n - 1}{t_{n+1}}\Big)^2\|\boldsymbol{\alpha}^n - \boldsymbol{\alpha}^{n-1}\|_2^2 - \|\boldsymbol{\alpha}^{n+1} - \boldsymbol{\alpha}^n\|_2^2\Big\}. \end{aligned}$$

Summing over $n = 1, 2, \ldots, k-1$, we obtain

$$\begin{aligned} &F(\boldsymbol{\alpha}^k) - F(\boldsymbol{\alpha}^1) \\ &\leq \frac{1}{2}\left\{L_2\Big(\frac{t_1 - 1}{t_2}\Big)^2\|\boldsymbol{\alpha}^1 - \boldsymbol{\alpha}^0\|_2^2 + \sum_{n=2}^{k-1}\Big(L_{n+1}\Big(\frac{t_n - 1}{t_{n+1}}\Big)^2 - L_n\Big)\|\boldsymbol{\alpha}^n - \boldsymbol{\alpha}^{n-1}\|_2^2 - L_k\|\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}\|_2^2\right\}. \end{aligned}$$

Note that $t_1 - 1 = 0$ and $\Big(L_{n+1}\Big(\frac{t_n - 1}{t_{n+1}}\Big)^2 - L_n\Big) \leq 0$ for all $n \geq 1$ because

$$\frac{t_n - 1}{t_{n+1}} = \frac{2(t_n - 1)}{1 + \sqrt{1 + 4(L_{n+1}/L_n)t_n^2}} \leq \frac{2(t_n - 1)}{\sqrt{4(L_{n+1}/L_n)t_n^2}} = \sqrt{\frac{L_n}{L_{n+1}}}\frac{t_n - 1}{t_n}.$$

Thus we have

$$F(\boldsymbol{\alpha}^k) - F(\boldsymbol{\alpha}^1) \leq 0.$$

Similarly, since the restart does not occur from $\boldsymbol{\alpha}^{(i,0)}$ to $\boldsymbol{\alpha}^{(i,\bar{k}_{i+1})}$ $(\forall i \geq 0)$, we have

$$F(\boldsymbol{\alpha}^{(i,k_{i+1})}) \leq F(\boldsymbol{\alpha}^{(i,0)}), \quad \forall k_{i+1} \in \{0, 1, \ldots, \bar{k}_{i+1}\},$$

and hence by definition,

$$F(\boldsymbol{\alpha}^{(i+1,0)}) = F(\boldsymbol{\alpha}^{(i,\bar{k}_{i+1}-1)}) \leq F(\boldsymbol{\alpha}^{(i,0)}).$$

■

Lemma 5 states that the initial function value $F(\boldsymbol{\alpha}^{(i,0)})$ of each outer iteration gives an upper bound of the subsequent function values and hence the sequence $\{F(\boldsymbol{\alpha}^{(i,0)})\}_{i=1}^{\infty}$ is non-increasing as illustrated in Figure 4.

Now we are ready to show the convergence result of Algorithm 3.

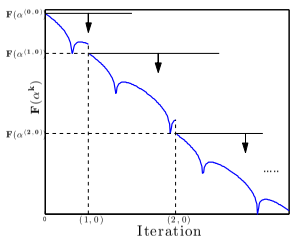


Figure 4: Illustration of the sequence $\{F(\boldsymbol{\alpha}^k)\}_{k=0}^{\infty}$ generated by Algorithm 3. Lemma 5 ensures that the function values $F(\boldsymbol{\alpha}^{(i,0)})$ at restarted points are non-increasing and gives upper bounds of the subsequent function values.

Theorem 6 *Consider the sequence* $\{\boldsymbol{\alpha}^k\}_{k=0}^{\infty} (\equiv \{\boldsymbol{\alpha}^{(i,k_{i+1})}\})$ *generated by Algorithm 3. Let* S^* *be the set of optimal solutions and* $B := \{\boldsymbol{\alpha} \mid F(\boldsymbol{\alpha}) \leq F(\boldsymbol{\alpha}^0)\}$ *be the level set. Assume that there exists a finite* R *such that*

$$R \geq \sup_{\boldsymbol{\alpha}^* \in S^*}\sup_{\boldsymbol{\alpha} \in B}\|\boldsymbol{\alpha} - \boldsymbol{\alpha}^*\|_2.$$

Then we have

$$F(\boldsymbol{\alpha}^k) - F(\boldsymbol{\alpha}^*) \leq 2\eta_u L_f R^2\Big(\frac{\log_2(k+2)}{k - \log_2(k+2)}\Big)^2, \quad \forall k \geq 3.$$

Proof From Lemma 5, we have $\boldsymbol{\alpha}^k \in B$ for all $k \geq 1$. Let $\boldsymbol{\alpha}^k = \boldsymbol{\alpha}^{(j,k_{j+1})}$. We assume $k_{j+1} \geq 1$ without loss of generality. From Proposition 1, we have

$$F(\boldsymbol{\alpha}^{(j,k_{j+1})}) - F(\boldsymbol{\alpha}^*) \leq \frac{2\eta_u L_f\|\boldsymbol{\alpha}^{(j,0)} - \boldsymbol{\alpha}^*\|_2^2}{k_{j+1}^2}$$

Moreover, for all $1 \leq i \leq j$, Lemma 5 leads to

$$\begin{aligned} F(\boldsymbol{\alpha}^{(j,k_{j+1})}) - F(\boldsymbol{\alpha}^*) &\leq F(\boldsymbol{\alpha}^{(j,0)}) - F(\boldsymbol{\alpha}^*) \\ &\leq F(\boldsymbol{\alpha}^{(i,0)}) - F(\boldsymbol{\alpha}^*) \\ &= F(\boldsymbol{\alpha}^{(i-1,\bar{k}_i-1)}) - F(\boldsymbol{\alpha}^*) \\ &\leq \frac{2\eta_u L_f\|\boldsymbol{\alpha}^{(i-1,0)} - \boldsymbol{\alpha}^*\|_2^2}{(\bar{k}_i - 1)^2} \\ &\leq \frac{2\eta_u L_f R^2}{(\bar{k}_i - 1)^2}. \end{aligned}$$

Note that $\bar{k}_i \geq K_i \geq 2$. Thus we obtain

$$F(\boldsymbol{\alpha}^{(j,k_{j+1})}) - F(\boldsymbol{\alpha}^*) \leq \frac{2\eta_u L_f R^2}{(\max\{\bar{k}_1 - 1, \bar{k}_2 - 1, \ldots, \bar{k}_j - 1, k_{j+1}\})^2} \leq \frac{2\eta_u L_f R^2}{(\max\{\bar{k}_1, \bar{k}_2, \ldots, \bar{k}_j, k_{j+1}\} - 1)^2}$$

From

$$k \geq \sum_{i=1}^{j} K_i = \frac{K_1(2^j - 1)}{2 - 1} \geq 2^{j+1} - 2,$$

the number of restart j is at most $\log_2(k+2) - 1$. Hence we have

$$\max\{\bar{k}_1, \bar{k}_2, \ldots, \bar{k}_j, k_{j+1}\} \geq \frac{k}{j+1} \geq \frac{k}{\log_2(k+2)}, \tag{24}$$

which leads to

$$F(\boldsymbol{\alpha}^{(j,k_{j+1})}) - F(\boldsymbol{\alpha}^*) \leq 2\eta_u L_f R^2 \Big(\frac{\log_2 k}{k - \log_2(k+2)}\Big)^2, \quad \forall k \geq 3.$$

■

The above theorem ensures the convergence of Algorithm 3 for C-SVM, ν-SVM, MM-MPM, MM-FDA, and ℓ_2-SVM because the level set B is bounded for any initial point $\boldsymbol{\alpha}^0 \in \mathrm{dom}(g)$ since S_C, S_ν, S_{FDA}, and S_{MPM} are bounded and the objective function of ℓ_2-SVM is strongly convex. Similarly, the algorithm is convergent for the logistic regression problem (13) if S_{LR} is replaced by $\{\boldsymbol{\alpha} \in \mathbb{R}^m \mid \boldsymbol{\alpha}^\top \boldsymbol{y} = 0,\ \xi \boldsymbol{e} \leq \boldsymbol{\alpha} \leq \boldsymbol{e}\}$ for some small constant $\xi > 0$. To guarantee the convergence of Algorithm 3 for the distance weighted discrimination problem (15), we need to replace S_{DWD} by $S_{\mathrm{DWD}}(\xi) := \big\{\boldsymbol{\alpha} \in \mathbb{R}^m \mid \boldsymbol{\alpha}^\top \boldsymbol{y} = 0,\ \xi \boldsymbol{e} \leq \boldsymbol{\alpha} \leq \frac{1}{m\nu}\boldsymbol{e}\big\}$ for some small constant $\xi > 0$ and make the assumption that $\|\widetilde{X}\boldsymbol{\alpha}\|_2 > 0$ for all $\boldsymbol{\alpha} \in S_{\mathrm{DWD}}(\xi)$.

The iteration complexity $O\big((\log(k)/k)^2\big)$ shown in Theorem 6 can be improved to $O\big((W(k)/k)^2\big)$ by finding a better lower bound of (24), where the Lambert's W function $W(\cdot)$ (Corless et al., 1996) is the inverse function of $x\exp(x)$ and diverges slower than $\log(\cdot)$.

Theorem 7 *Consider the sequence $\{\boldsymbol{\alpha}^k\}_{k=0}^{\infty} (\equiv \{\boldsymbol{\alpha}^{(i,k_{i+1})}\})$ generated by Algorithm 3. Let S^* be the set of optimal solutions and $B := \{\boldsymbol{\alpha} \mid F(\boldsymbol{\alpha}) \leq F(\boldsymbol{\alpha}^0)\}$ be the level set. Assume that there exists a finite R such that*

$$R \geq \sup_{\boldsymbol{\alpha}^* \in S^*} \sup_{\boldsymbol{\alpha} \in B} \|\boldsymbol{\alpha} - \boldsymbol{\alpha}^*\|_2.$$

Then we have

$$F(\boldsymbol{\alpha}^k) - F(\boldsymbol{\alpha}^*) \leq 2\eta_u L_f R^2 \Big(\frac{W(2k\ln 2)}{k\ln 2 - W(2k\ln 2)}\Big)^2, \quad \forall k \geq 3.$$

Proof See Appendix C. ■

4.3 A Practical FAPG

In this section, we develop a practical version of the FAPG method which reduces the computation cost of FAPG at each iterations.

4.3.1 Heuristic Backtracking Procedure

Although the APG method of (Scheinberg et al., 2014), which is employed in Algorithm 1 and modified in Algorithm 3, can guarantee the convergence, its principal drawback is the extra computational cost at the backtracking step. Recall that

$$Q_{L_k}(\boldsymbol{\alpha}^k; \boldsymbol{\beta}^k) = f(\boldsymbol{\beta}^k) + \langle \nabla f(\boldsymbol{\beta}^k),\ \boldsymbol{\alpha}^k - \boldsymbol{\beta}^k \rangle + \frac{L_k}{2}\|\boldsymbol{\alpha}^k - \boldsymbol{\beta}^k\|_2^2 + g(\boldsymbol{\alpha}^k).$$

To check the condition $F(\boldsymbol{\alpha}^k) \leq Q_{L_k}(\boldsymbol{\alpha}^k; \boldsymbol{\beta}^k)$, we need to recompute $f(\boldsymbol{\beta}^k)$ and $\nabla f(\boldsymbol{\beta}^k)$ since $\boldsymbol{\beta}^k$ is updated at each loop of the backtracking.

On the other hand, since the original APG of (Beck and Teboulle, 2009) does not update $\boldsymbol{\beta}^k$ during the backtracking step, we can reduce the computation cost by storing the value of $f(\boldsymbol{\beta}^k)$ and $\nabla f(\boldsymbol{\beta}^k)$. Hence we follow the strategy of (Beck and Teboulle, 2009), i.e., we remove the steps for updating t_k and $\boldsymbol{\beta}^k$ subsequent to 'bt' from Algorithm 3 and restore Step 2 as $t_{k+1} \leftarrow \frac{1+\sqrt{1+4t_k^2}}{2}$.

4.3.2 Skipping Extra Computations

The backtracking step ('bt') involves extra computation of the function value $F(\boldsymbol{\alpha}^k)$. Checking the termination criteria $L_k\|T_{L_k}(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k\|_2 < \epsilon$ also involves the extra computation of $\nabla f(\boldsymbol{\alpha}^k)$ which is needed for $T_{L_k}(\boldsymbol{\alpha}^k)$. These computation costs can be significant (see Tables 3 and 9 in Section 5). Thus we compute the 'bt' and $L_k\|T_{L_k}(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k\|_2$ in every 10 and 100 iterations, respectively.

Instead of checking the condition $L_k\|T_{L_k}(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k\|_2 < \epsilon$, we check whether $L_k\|\boldsymbol{\alpha}^k - \boldsymbol{\beta}^k\|_2 < \epsilon$ at each iteration. The reason for doing so is that computing $L_k\|\boldsymbol{\alpha}^k - \boldsymbol{\beta}^k\|_2$ is much cheaper than computing $T_{L_k}(\boldsymbol{\alpha}^k)$. It follows from (4) and (5) that if $\boldsymbol{\alpha}^k = \boldsymbol{\beta}^k$, then $\boldsymbol{\alpha}^k$ is an optimal solution. Moreover, $L_k(\boldsymbol{\alpha}^k - \boldsymbol{\beta}^k)$ represents a residual of a sufficient optimality condition for $\boldsymbol{\alpha}^k$ (and necessary and sufficient optimality condition for $\boldsymbol{\beta}^k$).

Finally, our practical FAPG is described as in Algorithm 4. We note that FAPG can be applied not only to the unified formulation shown in Section 3, but also to the optimization problem of the form (2). See Appendix D in which we demonstrate the performance of FAPG for ℓ_1-regularized classification models.

4.4 Computation of Primal Solution from Dual Solution

Various classification models can be solved in a unified way by applying the vector projection method and the FAPG method to the dual formulation (9) or (10). However, the primal solution $(\boldsymbol{w}, b)$ of (7) or (8) is still required to do classification tasks. In this section, we provide a method to compute the primal solution $(\hat{\boldsymbol{w}}, \hat{b})$ which corresponds to the dual solution $\hat{\boldsymbol{\alpha}}$.

Algorithm 4 A Practical Fast Accelerated Proximal Gradient Method

Input: f, ∇f, g, $\mathrm{prox}_{g,L}$, $\epsilon > 0$, $L > 0$, $\eta_u > 1$, $\eta_d > 1$ $\delta \in (0,1)$, $k_{max} > 0$, $K_1 \geq 2$, $\boldsymbol{\beta}^1 = \boldsymbol{\alpha}^0$
Output: $\boldsymbol{\alpha}^k$
Initialize: $t_1 \leftarrow 1$, $i \leftarrow 1$, $k_{re} \leftarrow 0$
for $k = 1, \ldots, k_{max}$ **do**
  $\boldsymbol{\alpha}^k \leftarrow T_{L_k}(\boldsymbol{\beta}^k) = \mathrm{prox}_{g,L_k}\left(\boldsymbol{\beta}^k - \frac{1}{L}\nabla f(\boldsymbol{\beta}^k)\right)$ # Step 1
  if k mod 10 == 1 **then**
    while $F(\boldsymbol{\alpha}^k) > Q_{L_k}(\boldsymbol{\alpha}^k; \boldsymbol{\beta}^k)$ **do**
      $L_k \leftarrow \eta L_k$; $\boldsymbol{\alpha}^k \leftarrow T_{L_k}(\boldsymbol{\beta}^k) \leftarrow \mathrm{prox}_{g,L_k}\left(\boldsymbol{\beta}^k - \frac{1}{L_k}\nabla f(\boldsymbol{\beta}^k)\right)$ # 'bt'
    end while
  end if
  if $\|L(\boldsymbol{\alpha}^k - \boldsymbol{\beta}^k)\| < \epsilon$ or ((k mod 100 == 1) and ($\|L(T_L(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k)\| < \epsilon$)) **then**
    break
  end if
  $L_{k+1} \leftarrow L_k/\eta_d$ # 'dec'
  $t_{k+1} \leftarrow \frac{1+\sqrt{1+4t_k^2}}{2}$ # Step 2
  $\boldsymbol{\beta}^{k+1} \leftarrow \boldsymbol{\alpha}^k + \frac{t_k - 1}{t_{k+1}}(\boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1})$ # Step 3
  if $k > k_{re} + K_i$ and $\langle \nabla f(\boldsymbol{\beta}^k),\ \boldsymbol{\alpha}^k - \boldsymbol{\alpha}^{k-1}\rangle + g(\boldsymbol{\alpha}^k) - g(\boldsymbol{\alpha}^{k-1}) > 0$ **then**
    $k_{re} \leftarrow k$, $K_{i+1} \leftarrow 2K_i$, $i \leftarrow i+1$. # 'mt'
    $\eta_d \leftarrow \delta \cdot \eta_d + (1-\delta)\cdot 1$. # 'st'
    $t_{k+1} \leftarrow 1$, $\boldsymbol{\beta}^{k+1} \leftarrow \boldsymbol{\alpha}^{k-1}$, $\boldsymbol{\alpha}^k \leftarrow \boldsymbol{\alpha}^{k-1}$ # 're'
  end if
end for

4.4.1 Computation of $\boldsymbol{w}$

The primal vector $\hat{\boldsymbol{w}}$ in (8) corresponding to the dual solution $\hat{\boldsymbol{\alpha}}$ (and $\hat{\boldsymbol{u}}$ for MM-FDA and MM-MPM) can be obtained as

$$\hat{\boldsymbol{w}} = \underset{\|\boldsymbol{w}\|_p \leq 1}{\mathrm{argmax}}\, \boldsymbol{w}^\top \boldsymbol{z}, \quad \text{with} \quad \hat{w}_i = \frac{\mathrm{sign}(z_i)|z_i|^{q-1}}{\|\boldsymbol{z}\|_q^{q-1}} \quad \forall\, i \in M, \tag{25}$$

where $q = p/(p-1)$ and $\boldsymbol{z} = \widetilde{A}\hat{\boldsymbol{\alpha}}$ (i.e., $\boldsymbol{z} = \widetilde{X}\hat{\boldsymbol{\alpha}}$ for C-SVM, ℓ_2-SVM, ν-SVM, logistic regression, and DWD; $\boldsymbol{z} = \hat{\boldsymbol{\alpha}} = (\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-) + (\Sigma_+ + \Sigma_-)^{1/2}\hat{\boldsymbol{u}}$ for MM-FDA; and $\boldsymbol{z} = \hat{\boldsymbol{\alpha}} = (\bar{\boldsymbol{x}}_+ + \Sigma_+^{1/2}\hat{\boldsymbol{u}}_+) - (\bar{\boldsymbol{x}}_- + \Sigma_-^{1/2}\hat{\boldsymbol{u}}_-)$ for MM-MPM).

4.4.2 Computation of b

There are some issues on computing an optimal bias term b. For C-SVM, ℓ_2-SVM, ν-SVM, DWD, and logistic regression, the bias term b is derived from the Lagrange multiplier corresponding to the constraint $\boldsymbol{\alpha}^\top \boldsymbol{y} = 0$. However, it is often difficult to compute the corresponding Lagrange multiplier. In addition, MM-FDA and MM-MPM does not provide a specific way to compute the bias term. Thus we need to estimate an appropriate value of b.

One of the efficient ways is to estimate b as the minimum solution of training error under a given $\hat{\boldsymbol{w}}$, i.e.,

$$\hat{b} \in \underset{b}{\mathrm{argmin}}\ \rho(b) := \sum_{i=1}^{m} \ell\big(y_i(\boldsymbol{x}_i^\top \hat{\boldsymbol{w}} - b)\big), \quad \text{where} \quad \ell(z) = \begin{cases} 1 & (z < 0) \\ 0 & (\text{otherwise}). \end{cases} \tag{26}$$

Let $\zeta_i = \boldsymbol{x}_i^\top \hat{\boldsymbol{w}}$ $(i = 1, 2, \ldots, m)$. Let σ be the permutation such that $\zeta_{\sigma(1)} \leq \zeta_{\sigma(2)} \leq \ldots \leq \zeta_{\sigma(m)}$. If $b < \zeta_{\sigma(1)}$, then $\boldsymbol{x}_i^\top \hat{\boldsymbol{w}} - b > 0$ $(\forall i \in M)$, i.e., all samples are predicted as positive $\hat{y} = +1$. Then we have m_- misclassified samples and thus $\rho(b) = m_-$. If $b \in (\zeta_{\sigma(1)}, \zeta_{\sigma(2)})$, then only $\boldsymbol{x}_{\sigma(1)}$ is predicted as negative $\hat{y} = -1$. Thus we have $\rho(b) = m_- + y_{\sigma(1)}$. Similarly, if $b \in (\zeta_{\sigma(k)}, \zeta_{\sigma(k+1)})$, then we have $\rho(b) = m_- + \sum_{i=1}^{k} y_{\sigma(i)}$. Thus, by letting $k^* \in \mathrm{argmin}_k \sum_{i=1}^{k} y_{\sigma(i)}$, an arbitrary constant $\hat{b} \in (\zeta_{\sigma(k^*)}, \zeta_{\sigma(k^*+1)})$ is an optimal solution of the problem (26). The minimization algorithm is as follows:

Step 1. Compute $\zeta_i = \hat{\boldsymbol{w}}^\top \boldsymbol{x}_i$ $(i = 1, 2, \ldots, m)$.

Step 2. Sort ζ_i as $\zeta_{\sigma(1)} \leq \zeta_{\sigma(2)} \leq \ldots \leq \zeta_{\sigma(m)}$.

Step 3. Find $k^* = \mathrm{argmin}_k \sum_{i=1}^{k} y_{\sigma(i)}$.

Step 4. Compute $\hat{b} = (\zeta_{(k^*)} + \zeta_{(k^*+1)})/2$.

Its computational complexity is $O\big(m(n + \log m)\big)$.

4.4.3 Duality Gap

The component b obtained above is not necessarily the primal optimal solution. Moreover, $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{u}}$ are close to optimal, but may not be exactly optimal due to e.g., numeric overflow.

We still obtain a primal solution $(\hat{\boldsymbol{w}}, \hat{b})$ with some optimality gap guarantee. Since $(\hat{\boldsymbol{w}}, \hat{b})$ and $\hat{\boldsymbol{\alpha}}$ are the primal and dual feasible solutions, respectively, we can obtain the duality gap by

$$\left\{\mathcal{L}(\widetilde{A}^\top \hat{\boldsymbol{w}} - \boldsymbol{a}\hat{b}) + \frac{1}{2C}\|\hat{\boldsymbol{w}}\|_2^2)\right\} - \left\{\mathcal{L}^*(-\hat{\boldsymbol{\alpha}}) + \frac{C}{2}\|\widetilde{A}\hat{\boldsymbol{\alpha}}\|_2^2\right\}$$

for the formulations of (7) and (9); and

$$\mathcal{L}(\widetilde{A}^\top \hat{\boldsymbol{w}} - \boldsymbol{a}\hat{b}) - \left\{\mathcal{L}^*(-\hat{\boldsymbol{\alpha}}) + \lambda\|\widetilde{A}\hat{\boldsymbol{\alpha}}\|_p^*\right\}$$

for the formulations of (8) and (10). Therefore, the obtained primal solution $(\hat{\boldsymbol{w}}, \hat{b})$ has a guarantee that the gap between its objective value and the optimal value is at most the duality gap.

5. Numerical Experiment

In this section we demonstrate the performance of our proposed FAPG algorithm. We run the numerical experiments on a Red Hat Enterprise Linux Server release 6.4 (Santiago) with Intel Xeon Processor E5-2680 (2.7GHz) and 64 GB of physical memory. We implemented the practical FAPG method (Algorithm 4) in MATLAB R2013a and the bisection method (Algorithm 2) in C++. The C++ code was called from MATLAB via MEX files.

We conducted the experiments using artificial datasets and benchmark datasets from LIBSVM Data (Chang and Lin, 2011). The artificial datasets were generated as follows. Positive samples $\{\boldsymbol{x}_i \in \mathbb{R}^n \mid i \in M_+\}$ and negative samples $\{\boldsymbol{x}_i \in \mathbb{R}^n \mid i \in M_-\}$ were distributed with n-dimensional standard normal distributions $\mathcal{N}_n(\boldsymbol{0}, I_n)$ and $\mathcal{N}_n(\frac{10}{\sqrt{n}}\boldsymbol{e}, SS^\top)$, respectively, where the elements of the $n \times n$ matrix S are i.i.d. random variables following the standard normal distribution $\mathcal{N}(0,1)$. The marginal probability of the label was assumed to be same, i.e. $P(y=+1) = P(y=-1) = \frac{1}{2}$. After generating the samples, we scaled them so that each input vector $\boldsymbol{x}_i$ $(\forall i \in M)$ was in $[-1,1]^n$ for the purpose of computational stability, following LIBSVM (Chang and Lin, 2011). On the other hand, we scaled the benchmark datasets, that are not scaled by Chang and Lin (2011), so that $\boldsymbol{x}_i \in [0,1]^n$, $(\forall i \in [m])$ in order to leverage their sparsity. The details of benchmark datasets are shown in Table 1.

5.1 Projection Algorithms

Before presenting the performance of our practical FAPG algorithm, we compared the performance of our bisection algorithm (Algorithm 2) against the breakpoint search algorithm (Kiwiel, 2008, Algorithm 3.1) with random pivoting. Both algorithms were implemented in C++. We generated $\mathbb{R}^n$-valued random vectors $\tilde{\boldsymbol{\alpha}}$ with uniformly distributed elements and computed the projections $P_{S_\nu}(\tilde{\boldsymbol{\alpha}})$ of $\tilde{\boldsymbol{\alpha}}$ onto S_ν, where $S_\nu := \left\{\boldsymbol{\alpha} \mid \boldsymbol{e}_o^\top \boldsymbol{\alpha}_o = \frac{1}{2},\ o \in \{+,-\},\ \boldsymbol{0} \le \boldsymbol{\alpha} \le \frac{1}{m\nu}\boldsymbol{e}\right\}$ with $\nu = 0.5$. The bisection algorithm used the accuracy of $\epsilon' \approx 2.22 \times 10^{-16}$ (i.e., IEEE 754 double precision). Table 2 reports the average and standard deviation of the computation times and the number of iterations of 20 trials. As we can see from Table 2, the bisection method was faster and more stable (in the sense that the deviations are smaller) than the breakpoint search algorithm. This explains why we

data	m (	m_+,	m_-)	n	range	density	source
a8a	22,696 (	5,506,	17,190)	123	$[0,1]^n$	0.113	(Bache and Lichman, 2013)
a9a	32,561 (	7,841,	24,720)	123	$[0,1]^n$	0.113	(Bache and Lichman, 2013)
australian	690 (	307,	383)	14	$[-1,1]^n$	0.874	(Bache and Lichman, 2013)
breast-cancer	683 (	444,	239)	10	$[-1,1]^n$	1.000	(Bache and Lichman, 2013)
cod-rna	59,535 (	39,690,	19,845)	8	$[0,1]^n$	0.999	(Uzilov et al., 2006)
colon-cancer	62 (	40,	22)	2,000	$[0,1]^n$	0.984	(Alon et al., 1999)
covtype	581,012 (	297,711,	283,301)	54	$[0,1]^n$	0.221	(Bache and Lichman, 2013)
diabetes	768 (	500,	268)	8	$[-1,1]^n$	0.999	(Bache and Lichman, 2013)
duke	44 (	21,	23)	7,129	$[0,1]^n$	0.977	(West et al., 2001)
epsilon	400,000 (	199,823,	200,177)	2,000	$[-0.15,0.16]^n$	1.000	(Sonnenburg et al., 2008)
fourclass	862 (	307,	555)	2	$[-1,1]^n$	0.996	(Ho and Kleinberg, 1996)
german.numer	1,000 (	300,	700)	24	$[-1,1]^n$	0.958	(Bache and Lichman, 2013)
gisette	6,000 (	3,000,	3,000)	5,000	$[-1,1]^n$	0.991	(Guyon et al., 2005)
heart	270 (	120,	150)	13	$[-1,1]^n$	0.962	(Bache and Lichman, 2013)
ijcnn1	35,000 (	3,415,	31,585)	22	$[-0.93,1]^n$	0.591	(Prokhorov, 2001)
ionosphere	351 (	225,	126)	34	$[-1,1]^n$	0.884	(Bache and Lichman, 2013)
leu	38 (	11,	27)	7,129	$[0,1]^n$	0.974	(Golub et al., 1999)
liver-disorders	345 (	145,	200)	6	$[-1,1]^n$	0.991	(Bache and Lichman, 2013)
madelon	2,000 (	1,000,	1,000)	500	$[0,1]^n$	0.999	(Guyon et al., 2005)
mushrooms	8,124 (	3,916,	4,208)	112	$[0,1]^n$	0.188	(Bache and Lichman, 2013)
news20.binary	19,996 (	9,999,	9,997)	1,355,191	$[0,1]^n$	3.36E-04	(Keerthi and DeCoste, 2005)
rcv1-origin	20,242 (	10,491,	9,751)	47,236	$[0,0.87]^n$	0.002	(Lewis et al., 2004)
real-sim	72,309 (	22,238,	50,071)	20,958	$[0,1]^n$	0.002	(McCallum)
skin-nonskin	245,057 (	50,859,	194,198)	3	$[0,1]^n$	0.983	(Bache and Lichman, 2013)
sonar	208 (	97,	111)	60	$[-1,1]^n$	1.000	(Bache and Lichman, 2013)
splice	1,000 (	517,	483)	60	$[-1,1]^n$	1.000	(Bache and Lichman, 2013)
svmguide1	3,089 (	1,089,	2,000)	4	$[0,1]^n$	0.997	(Hsu et al., 2003)
svmguide3	1,243 (	947,	296)	22	$[0,1]^n$	0.805	(Hsu et al., 2003)
url	2,396,130 (	1,603,985,	792,145)	3,231,961	$[0,1]^n$	3.54E-05	(Ma et al., 2009)
w7a	24,692 (	740,	23,952)	300	$[0,1]^n$	0.039	(Platt, 1998)
w8a	49,749 (	1,479,	48,270)	300	$[0,1]^n$	0.039	(Platt, 1998)

Table 1: Details of Datasets. We have scaled the datasets that are highlighted in boldface type.

		msec.(#iter.)			
		Breakpoint		Bisection	
dim. n	range	ave.	std.	ave.	std.
100,000	$[0,10]^n$	8.3 (25.7)	2.0 (5.4)	4.9 (31.1)	0.6 (4.7)
100,000	$[0,1000]^n$	9.0 (27.3)	1.4 (4.1)	5.5 (27.1)	1.2 (3.2)
1,000,000	$[0,10]^n$	94.2 (22.6)	16.2 (4.3)	49.9 (32.0)	3.4 (3.1)
1,000,000	$[0,1000]^n$	99.1 (26.5)	15.6 (3.2)	53.4 (30.0)	4.4 (1.5)

Table 2: Runtime of Projection Algorithms

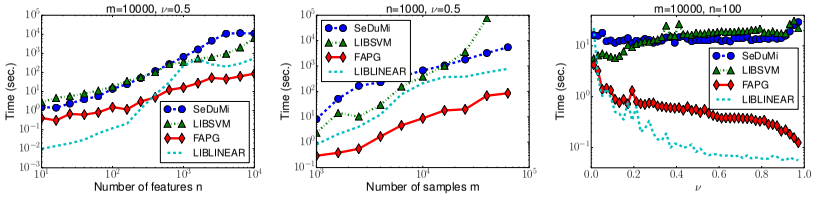


Figure 5: Computation Time for ν-SVM

have chosen to use the bisection methods in Section 4.1 to perform the projection steps in Algorithm 4.

5.2 ν-SVM

As mentioned in Section 3.2, the standard C-SVM (11) and ν-SVM (14) are equivalent. Here we chose ν-SVM to solve because choosing the parameter of ν-SVM is easier than that of C-SVM. We solved the ν-SVM (14) via our FAPG method, SeDuMi (Sturm, 1999), and LIBSVM (Chang and Lin, 2011). SeDuMi is a general purpose optimization solver implementing an interior point method for large-scale second-order cone problems such as (14). LIBSVM implements the sequential minimal optimization (SMO) (Platt, 1998) which is specialized for learning ν-SVM. For reference, we also compared the FAPG method with LIBLINEAR (Fan et al., 2008) which implements a highly optimized stochastic dual coordinate descent method (Hsieh et al., 2008) for C-SVM[3] (Cortes and Vapnik, 1995) and is known to be quite an efficient method; we note that it may not be a fair comparison because LIBLINEAR omits the bias term b of C-SVM from the calculations, [4] i.e., it solves a less complex model than the ν-SVM (14) in order to speed up the computation.

We terminate the algorithms if the violation of the KKT optimality condition is less than $\epsilon = 10^{-6}$. The heuristic option in LIBSVM was set to "off" in order to speed up its convergence for large datasets. In the FAPG method, (η_u, η_d, δ) were set to $(1.1, 1.1, 0.8)$. L_0 was set to the maximum value in the diagonal elements of $\widetilde{X}^\top \widetilde{X}$ (i.e., the coefficient matrix of the quadratic form). The initial point $\boldsymbol{\alpha}^0$ was set to the *center* $\boldsymbol{\alpha}^c$ of S_ν, i.e. $\alpha^c_i = \frac{1}{2m_o}$, $i \in M_o$, $o \in \{+, -\}$.

5.2.1 Scalability

First, we compare the computation time with respect to the size of the datasets and parameters using artificial datasets. The results are shown in Figure 5. The left panel shows the computation time with respect to the dimension n of the features for $m = 10000$ and $\nu = 0.5$. The FAPG method has a clear advantage when the dimension n is high, say

3. C-SVM (with the bias term b) is known to lead to the same decision function as ν-SVM if ν and C are set properly (Schölkopf et al., 2000). The value of C corresponding to ν can be computed by LIBSVM.
4. Although LIBLINEAR can virtually deal with the bias term b by augmenting the dimension of the samples, the best performance of the resulting model tends to be lower than the one of ν-SVM as reported in (Kitamura et al., 2014).

Function	% Time	Time	# Evals.	Time/Eval.
$\nabla f(\boldsymbol{\alpha})$	78.1%	14.909	1375	0.0108
$f(\boldsymbol{\alpha})$	10.8%	2.053	369	0.0056
$P_{S_\nu}(\boldsymbol{\alpha})$	6.9%	1.307	1453	0.0009
Total Runtime:		19.080		

Table 3: Runtime Breakdown of the FAPG Method (sec.)

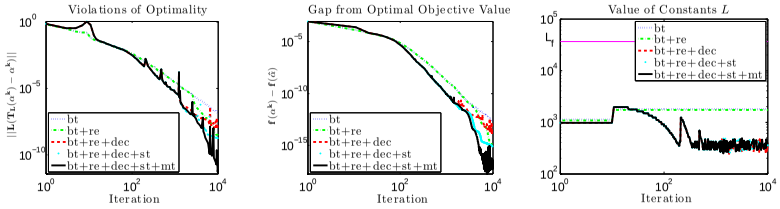


Figure 6: Effect of various acceleration strategies for the APG Method.

$n \geq 10^3$. The middle panel shows the computation time with respect to the number m of the samples for $n = 1000$ and $\nu = 0.5$. LIBSVM did not converge within a week for $m = 63000$. SeDuMi, LIBLINEAR, and the APG method were scalable for the increased number of samples m. The right panel illustrates the computation time with respect to the parameter ν for $m = 10000$ and $n = 100$. We may observe that the FAPG method (and LIBLINEAR) is very efficient when ν is larger than 0.1. This can be attributed to the fact that larger ν shrinks the feasible region S_ν and shorten the distance between the initial point $\boldsymbol{\alpha}^0 = \boldsymbol{\alpha}^c$ and the optimal solution $\boldsymbol{\alpha}^*$.

5.2.2 Runtime Breakdown

Table 3 shows the runtime breakdown of the FAPG method for the artificial dataset with $(m, n, \nu) = (10000, 1000, 0.5)$. The computation of the gradient $\nabla f(\boldsymbol{\alpha})$ and the function $f(\boldsymbol{\alpha})$ was the most time-consuming parts in the FAPG method. Since the computations of 'bt' (and $\|L_k(T_{L_k}(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k)\|$) involve the extra evaluation of $f(\boldsymbol{\alpha}^k)$ (and/or $\nabla f(\boldsymbol{\alpha}^k)$), computing them only every 100 iteration (and 10 iteration, respectively) as in Algorithm 4 would be effective to reduce the total runtime. Our projection algorithm was efficient enough in the sense that its runtime was marginal compared to the runtime of the other parts.

5.2.3 Effect of Each Acceleration Strategy

Figure 6 shows the running history of the FAPG method with various acceleration strategies for the artificial dataset with $(m, n, \nu) = (10000, 1000, 0.5)$.

The left panel depicts the violations of the optimality, i.e. the values of $\|L_k(T_{L_k}(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k)\|$. The FAPG method with 'bt+re' restarted at $k = 2594$, where the sharp decrease occurred. 'dec' was effective to reduce $L_k\|T_{L_k}(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k\|$ in the early iterations. The FAPG method with 'bt+re+dec' seems to be unstable near the optimum (say $k \geq 10^3$), but

the one with 'bt+re+dec+st' converged stably. Moreover, 'bt+re+dec+st+mt' obtained much more accurate solution than others. We note that several spikes of the values occurred when using 'dec' because it sometimes leads a small value of L_k that violates the condition (6). However, the violation was swiftly recovered in every 10 iterations by 'bt' and did not affect the speed of convergence so much.

The middle panel illustrated the gap in objective value $f(\boldsymbol{\alpha}^k) - f(\hat{\boldsymbol{\alpha}})$ where we regarded $\hat{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^{10000}$ of 'bt+re+dec+st+mt' as an optimal solution. The gap behaved similar to the violations of the optimality $\|L_k(T_{L_k}(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k)\|$. At $k = 1000$, the objective value $f(\boldsymbol{\alpha}^k)$ reached $f(\hat{\boldsymbol{\alpha}})$ within a relative error 0.0003% though the violation of optimality was greater than $\epsilon = 10^{-6}$. Thus, a larger tolerance, say $\epsilon = 10^{-5}$, may also lead to a reasonable solution in practice.

The right panel illustrated the value of constant L_k of the FAPG at each iteration and the value of Lipschitz constant L_f; L_f is known to be the largest eigenvalue of $\widetilde{X}^\top \widetilde{X}$. While $L_f = 3.61 \times 10^4$, the FAPG method with 'bt+re+dec+st+mt' leads to a much smaller average value of 3.68×10^2 for L_k, while the maximum value for L_k is 1.95×10^3.

5.2.4 Benchmark Results

We also conducted experiments using the benchmark datasets. C was set to 10. The computation time is shown in Table 4. When using linear kernel, the FAPG method outperformed LIBSVM and SeDuMi for large datasets where $m \geq 50000$. We indicated the smallest computation time among the three methods except for LIBLINEAR by boldface because the optimal solutions of LIBLINEAR are different from those of the other three methods (note that LIBLINEAR solves a simpler type of SVM, i.e., C-SVM without the bias term b). However, the FAPG method had an advantage over LIBLINEAR for many datasets where $n \geq 2000$.

When using the RBF kernel, SeDuMi broke down for datasets with $m \geq 10000$ as in the case of the artificial datasets. The FAPG method run out of memory for $m \geq 50000$ since it requires the $m \times m$ dense kernel matrix K to compute the gradient $\nabla f(\boldsymbol{\alpha})$. The FAPG can avoid the memory shortage by computing the elements of K on demand as LIBSVM does for large datasets. Although the current implementation has room for improvement, our practical FAPG method was still competitive with LIBSVM and had stable and good performance for datasets with large n.

We should remark that the number of iterations taken by the FAPG method with the RBF kernel tends to be smaller than the one with the linear kernel. However, when using the RBF kernel, the computational complexity of $\nabla f(\boldsymbol{\alpha})$ changes from $O(mn)$ to $O(m^2)$. Hence the total runtime tended to increase except for "gisette" whose n and m have the same order of magnitude.

In summary, the FAPG method showed better performance than specialized algorithms designed for learning SVM, such as LIBSVM and LIBLINEAR, in many datasets. Taking into account of the generality (i.e., applicability to other models) of FAPG, one could argue that it is a very efficient method.

Table 5 shows the values taken by the parameter L_k. We can see that our practical FAPG method (with backtracking strategy and decreasing strategy for L_k) keep the values of L_k to be much smaller than the Lipschitz constant L_f.

				Linear				RBF		
data	m	n	ν	SeDuMi	LIBSVM	FAPG (iter)	LIBLIN	SeDuMi	LIBSVM	FAPG (iter)
a8a	22,696	123	0.368	16.82	32.31	**1.70** (563)	0.06	–	**68.24**	72.88(343)
a9a	32,561	123	0.365	25.42	66.11	**4.84** (665)	0.07	–	**138.89**	170.63(390)
australian	690	14	0.348	0.34	**0.12**	0.86 (4056)	0.003	8.03	**0.049**	0.076(243)
breast-cancer	683	10	0.128	0.21	**0.004**	0.05 (253)	0.001	6.72	**0.015**	0.050(144)
cod-rna	59,535	8	0.223	6.71	53.73	**3.39** (588)	0.17	**	267.56	**(**)
colon-cancer	62	2,000	0.078	0.18	**0.01**	0.09 (210)	0.15	0.17	**0.013**	**0.013**(109)
covtype	581,012	54	0.620	288.71	16164.88	**60.75** (701)	1.55	**	–	**(**)
diabetes	768	8	0.533	0.14	**0.02**	0.06 (306)	0.003	8.53	**0.060**	0.093(296)
duke	44	7,129	0.106	0.38	**0.04**	0.11 (317)	0.30	0.09	0.041	**0.017**(120)
epsilon	400,000	2,000	0.500	–	–	**1685.33** (2643)	–	**	–	**(**)
fourclass	862	2	0.543	0.12	**0.01**	0.07 (356)	0.0004	12.34	0.069	**0.068**(150)
german.numer	1,000	24	0.525	**0.26**	**0.26**	0.29 (1107)	0.04	17.13	0.14	**0.10**(236)
gisette	6,000	5,000	0.100	4572.24	57.48	**9.31** (590)	0.41	5586.74	59.07	**4.93**(235)
heart	270	13	0.388	0.14	**0.005**	0.04 (232)	0.001	0.47	**0.008**	0.040(196)
ijcnn1	35,000	22	0.186	7.31	53.57	**5.67** (2000)	0.29	–	**73.83**	236.15(511)
ionosphere	351	34	0.202	0.25	**0.02**	0.22 (1064)	0.34	0.85	**0.012**	0.051(234)
leu	38	7,129	0.070	0.40	**0.03**	0.13 (175)	0.17	0.08	0.034	**0.013**(103)
liver-disorders	345	6	0.731	0.11	**0.01**	0.11 (736)	0.007	0.81	**0.015**	0.037(168)
madelon	2,000	500	0.603	29.15	11.57	**0.32** (510)	87.43	133.85	3.77	**0.24**(108)
mushrooms	8,124	112	0.096	8.74	1.34	**0.56** (435)	0.06	28505.46	**5.74**	19.02(661)
news20.binary	19,996	1,355,191	0.100	–	1333.26	**22.29** (321)	1.57	–	929.85	**51.86**(11)
rcv1-origin	20,242	47,236	0.097	–	587.10	**7.82** (485)	8.96	–	192.16	**97.68**(189)
real-sim	72,309	20,958	0.065	–	6351.62	**12.71** (384)	4.33	**	**1879.90**	**(**)
skin-nonskin	245,057	3	0.233	62.70	726.20	**8.02** (609)	0.09	**	**4425.59**	**(**)
sonar	208	60	0.117	0.29	**0.12**	0.33 (1922)	3.81	0.27	**0.009**	0.060(279)
splice	1,000	60	0.432	0.56	0.25	**0.11** (331)	0.02	15.27	0.18	**0.067**(120)
svmguide1	3,089	4	0.180	0.35	**0.08**	0.15 (394)	0.003	1188.65	**0.42**	1.45(323)
svmguide3	1,243	22	0.408	**0.36**	1.07	0.93 (3248)	1.70	30.17	0.23	**0.21**(430)
url	2,396,130	3,231,961	0.500	–	–	**4853.91** (2521)	–	**	**	**(**)
w7a	24,692	300	0.031	69.90	135.91	**14.13** (4280)	0.81	–	**17.89**	286.27(1208)
w8a	49,749	300	0.031	189.40	143.42	**38.69** (5960)	0.47	–	**124.55**	1423.42(2100)

Table 4: Computation Time for Benchmark Datasets (sec.). '–' means that the algorithm did not converge with in 36000 seconds. '**' means that it had run out of memory. The best results except for LIBLINEAR are indicated by boldface. The underlined results are better than LIBLINEAR.

	Linear			RBF		
	L_k(FAPG)			L_k(FAPG)		
	ave.	max.	L_f	ave.	max.	L_f
a8a	2.99.E+03	1.54.E+04	1.43.E+05	7.00.E+01	2.27.E+02	2.00.E+04
a9a	5.59.E+03	3.38.E+04	2.05.E+05	1.37.E+02	4.99.E+02	2.88.E+04
australian	1.97.E+02	1.21.E+03	2.91.E+03	1.92.E+01	4.69.E+01	3.56.E+02
breast-cancer	9.05.E+01	2.27.E+02	1.68.E+04	1.21.E+01	2.13.E+01	4.52.E+02
cod-rna	1.05.E+03	5.65.E+03	1.24.E+05	–	–	–
colon-cancer	4.02.E+02	6.60.E+02	3.80.E+04	5.91.E-01	9.09.E-01	5.65.E+01
covtype	3.71.E+04	2.37.E+05	4.58.E+06	–	–	–
diabetes	5.21.E+01	1.53.E+02	1.76.E+03	1.64.E+01	4.69.E+01	6.34.E+02
duke	2.82.E+03	4.55.E+03	6.00.E+04	6.89.E-01	9.09.E-01	4.09.E+01
epsilon	2.09.E+04	1.85.E+05	1.40.E+05	–	–	–
fourclass	5.61.E+01	1.87.E+02	2.80.E+02	5.72.E+01	1.03.E+02	5.36.E+02
german.numer	2.16.E+02	1.03.E+03	8.44.E+03	1.95.E+01	4.69.E+01	4.50.E+02
gisette	1.84.E+04	5.71.E+04	2.02.E+07	9.17.E+00	2.13.E+01	3.61.E+03
heart	1.07.E+02	2.53.E+02	7.49.E+02	1.01.E+01	2.13.E+01	1.19.E+02
ijcnn1	1.20.E+03	1.05.E+04	5.89.E+03	1.93.E+02	8.84.E+02	3.12.E+04
ionosphere	2.46.E+02	9.83.E+02	2.14.E+03	1.19.E+01	2.83.E+01	2.24.E+02
leu	1.70.E+03	2.75.E+03	6.05.E+04	6.07.E-01	9.09.E-01	3.46.E+01
liver-disorders	1.27.E+01	3.85.E+01	1.84.E+03	5.04.E+00	9.68.E+00	8.23.E+02
madelon	9.54.E+01	2.87.E+02	2.44.E+05	6.54.E-01	1.00.E+00	1.92.E+03
mushrooms	1.84.E+03	6.34.E+03	2.30.E+05	2.31.E+01	1.03.E+02	1.74.E+04
news20.binary	4.63.E+01	1.03.E+02	1.17.E+03	9.09.E-01	9.09.E-01	2.00.E+04
rcv1-origin	3.20.E+01	8.30.E+01	4.49.E+02	4.42.E-01	9.09.E-01	2.02.E+04
real-sim	6.75.E+01	2.27.E+02	9.21.E+02	–	–	–
skin-nonskin	5.08.E+03	2.82.E+04	2.19.E+05	–	–	–
sonar	2.27.E+02	1.12.E+03	2.68.E+03	4.88.E+00	9.68.E+00	1.51.E+02
splice	3.59.E+02	1.11.E+03	1.74.E+03	6.63.E+00	1.06.E+01	3.57.E+02
svmguide1	3.07.E+01	1.13.E+02	2.47.E+03	1.52.E+01	4.69.E+01	2.92.E+03
svmguide3	1.53.E+02	6.68.E+02	4.92.E+03	1.22.E+01	4.69.E+01	1.15.E+03
url	9.94.E+05	1.10.E+07	1.57.E+08	–	–	–
w7a	5.53.E+03	4.62.E+04	6.52.E+04	4.80.E+01	1.83.E+02	2.31.E+04
w8a	1.04.E+04	1.06.E+05	1.32.E+05	1.14.E+02	4.51.E+02	4.65.E+04

Table 5: Constants for Benchmark Datasets

	Artificial	svmguide1	mushrooms	a8a
$\xi = 10^{-2}$	99.8%	95.5%	99.4%	84.7%
$\xi = 10^{-3}$	100.0%	95.5%	100.0%	84.8%
$\xi = 10^{-4}$	100.0%	95.5%	100.0%	84.8%
$\xi = 10^{-5}$	100.0%	95.5%	100.0%	84.8%
LIBLINEAR	85.2%	90.7%	100.0%	84.7%

Table 6: Classification Accuracies with Varied ξ from 10^{-2} to 10^{-5}.

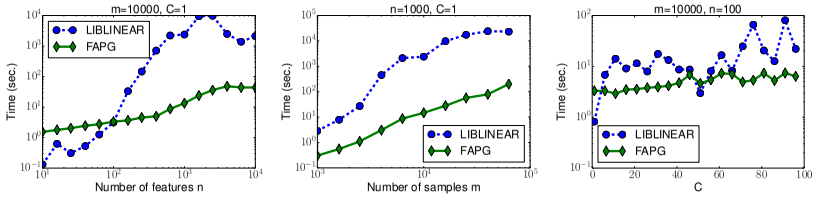


Figure 7: Computation Time for Logistic Regression.

5.3 Logistic Regression

We solved the dual logistic regression (13) via the FAPG method and LIBLINEAR (Fan et al., 2008). LIBLINEAR implements a highly optimized stochastic dual coordinate descent method (Yu et al., 2011) whose subproblems are solved by the Newton method. Note that, as in the case of SVM, LIBLINEAR omits the bias term b from the calculations, i.e., it solves a less complex model than (13). We terminate both algorithms if the violation of the KKT optimality condition is less than $\epsilon = 10^{-6}$. In the FAPG method, (η_u, η_d, δ) were set to $(1.1, 1.1, 0.8)$. L_0 was set to 1. The initial point $\boldsymbol{\alpha}^0$ was set to $\xi \boldsymbol{e}$.

5.3.1 Sensitivity to ξ

Since the gradient ∇f is not defined at $\alpha_i = 0, 1$ $(i \in M)$, we approximately solve the problem by using the constraints $\xi \leq \alpha_i \leq 1 - \xi$ $(i \in M)$, where $\xi > 0$. Table 6 shows the classification accuracy for $\xi = 10^{-2}, 10^{-3}, 10^{-4}$, and 10^{-5}. For all cases, the approximated logistic regression performed better than LIBLINEAR. The classification accuracy of FAPG was identical for $\xi \leq 10^{-3}$, and better than the accuracy with $\xi = 10^{-2}$. We employ $\xi = 10^{-4}$ in the following experiments.

5.3.2 Scalability

First, we compare the computation time with respect to the size of the datasets and parameter using artificial datasets. The results (for the linear kernel) are shown in Figure 7. The left panel shows the computation time with respect to the dimension n of the features for $m = 10000$ and $C = 10$. The FAPG method has a clear advantage when the dimension n is high, say $n \geq 10^2$. The middle panel shows the computation time with respect to the number m of the samples for $n = 1000$ and $C = 10$. The computation time of FAPG grew

Function	% Time	Time	# Evals.	Time/Eval.
$\nabla f(\boldsymbol{\alpha})$	85.4%	10.005	1452	6.89.E-03
$f(\boldsymbol{\alpha})$	12.3%	1.443	340	4.24.E-03
$P_{S_{\mathrm{LR}}}(\boldsymbol{\alpha})$	0.5%	0.058	1473	3.94.E-05
Total Runtime:		11.720		

Table 7: Runtime Breakdown of the FAPG Method (sec.)

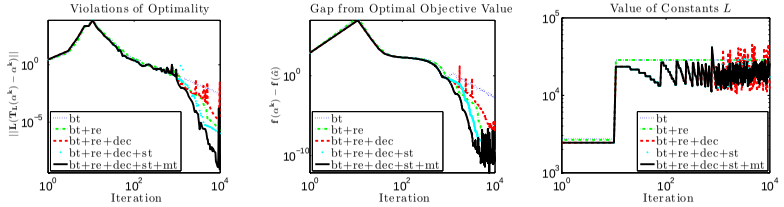


Figure 8: Effect of various acceleration strategies for the FAPG Method.

slower than the one of LIBLINEAR. The right panel illustrates the computation time with respect to the parameter C for $m = 10000$ and $n = 100$. Unlike ν-SVM, the computation time of the FAPG did not change so much depending on the parameter C because it does not affect the area of feasible region. We can observe that the FAPG method is numerically more stable than LIBLINEAR with respect to the changes of the parameter C.

5.3.3 Runtime Breakdown

Table 7 shows the runtime breakdown of the FAPG method for the artificial dataset with $(m, n, C) = (10000, 1000, 10)$. The computation of the gradient $\nabla f(\boldsymbol{\alpha})$ and the function $f(\boldsymbol{\alpha})$ was the most time-consuming parts as in the case of ν-SVM. Thus skipping backtracking step and evaluation of the optimality $\|L_k(\boldsymbol{\alpha}^k - T_{L_k}(\boldsymbol{\alpha}^k))\|$ reduce the computation time significantly because it can avoid the extra computation of f and ∇f.

5.3.4 Effect of Each Acceleration Strategy

Figure 8 shows the running history of the FAPG method with various acceleration strategies for the dataset 'a8a'. C was set to 10.

The left panel depicts the violations of the optimality, i.e. the values of $\|L_k(T_{L_k}(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^k)\|$. 're' had effect on decreasing the values after $k = 762$ at which the first restart occur. 'bt+re+dec' just became unstable compared to 'bt+re' because the gradient of the logistic regression is intrinsically large near the boundary of the feasible region S_{LR}. However, 'bt+re+dec+st' could recover from the instability. It is remarkable that only 'bt+re+dec+st+mt' fell below 10^{-6} in 10000 iterations. Thus, one could argue that 'mt' had a significant effect on reducing the violation. The middle panel illustrated the gap in the objective value $f(\boldsymbol{\alpha}^k) - f(\hat{\boldsymbol{\alpha}})$ where we regarded $\hat{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^{10000}$ of 'bt+re+dec+st+mt' as an optimal solution. 'bt+re+dec+st+mt' decreased the function value faster than others

	m	n	C	FAPG (iter)	LIBLIN
a8a	22696	123	10	19.82 (4939)	9.82
a9a	32561	123	10	35.32 (6155)	57.62
australian	690	14	10	0.45 (1738)	0.01
breast-cancer	683	10	10	0.49 (1904)	0.004
cod-rna	59535	8	10	31.40 (3718)	2.03
colon-cancer	62	2000	10	0.11 (269)	0.16
covtype	581012	54	10	2039.19 (20143)	73.74
diabetes	768	8	10	0.20 (794)	0.01
duke	44	7129	10	0.27 (327)	0.31
epsilon	400000	2000	10	27791.25 (10873)	5225.43
fourclass	862	2	10	0.21 (851)	0.001
german.numer	1000	24	10	0.56 (1741)	0.31
gisette	6000	5000	10	20.30 (1289)	0.54
heart	270	13	10	0.15 (817)	0.02
ijcnn1	35000	22	10	14.57 (2478)	0.51
ionosphere	351	34	10	0.39 (1639)	0.31
leu	38	7129	10	0.11 (148)	0.19
liver-disorders	345	6	10	0.16 (828)	0.01
madelon	2000	500	10	2.23 (910)	1035.26
mushrooms	8124	112	10	2.26 (1224)	0.09
news20.binary	19996	1355191	10	96.24 (1263)	22.15
rcv1-origin	20242	47236	10	14.41 (1293)	11.41
real-sim	72309	20958	10	142.79 (5630)	7.80
skin-nonskin	245057	3	10	247.11 (8373)	3.59
sonar	208	60	10	0.22 (1150)	0.10
splice	1000	60	10	0.66 (1689)	6.02
svmguide1	3089	4	10	0.91 (1313)	0.16
svmguide3	1243	22	10	0.49 (1263)	0.29
url	2396130	3231961	10	– (–)	–
w7a	24692	300	10	17.00 (3400)	10.69
w8a	49749	300	10	51.13 (5279)	42.77

Table 8: Computation Time for Benchmark Datasets (sec.). '–' means that the algorithm did not converge with in 36000 seconds. '**' means that it had run out of memory. The results indicated by underline are better than LIBLINEAR.

and found the minimum solution while oscillation occurred at the end of iteration. The right panel illustrated the value of constant L_k of the FAPG at each iteration.

5.3.5 Benchmark Results

We measured computation time for the benchmark datasets. The parameter C was set to 10 throughout this experiment. The experimental results are shown in Table 8. LIBLINEAR was efficient because it solves a less complex model (without the bias term) than (13). In some cases, however, FAPG solved (13) faster than LIBLINEAR. The computation time and iteration count of FAPG was nearly proportional to the number of samples m. On the other hand, LIBLINEAR took much longer time for 'madeon' and 'splice' although they are relatively small datasets.

Taking these results and Figure 7 into account, one could argue that the FAPG method exhibits stable convergence empirically.

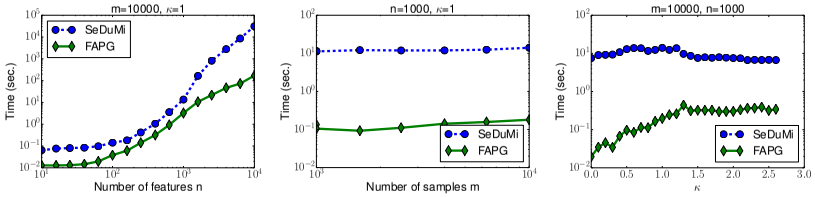


Figure 9: Computation Time for MM-MPM

5.4 MM-MPM

Next, we conducted experiments on MM-MPM (17). To the best of our knowledge, there are no specialized methods for MM-MPM. Thus, we compared the FAPG method only to SeDuMi (Sturm, 1999) which implements an interior point method for the large-scale second-order cone problems such as (17). We terminated the computations if the violation of KKT optimality is less than $\epsilon = 10^{-6}$. In the FAPG method, (η_u, η_d, δ) were set to $(1.1, 1.1, 0.8)$. The value of L_0 was set to the maximum value in the diagonal elements of $\widetilde{\Sigma}^\top\widetilde{\Sigma}$ (i.e., the coefficient matrix of the quadratic form), where $\widetilde{\Sigma} = [\Sigma_+^{1/2}, -\Sigma_-^{1/2}]$. The initial point $\boldsymbol{u}^0 = (\boldsymbol{u}_+^0, \boldsymbol{u}_-^0)$ was set to the origin $\mathbf{0}$.

5.4.1 Scalability

The computation time for the artificial datasets are shown in Figure 9. The left panel shows the results with respect to the number n of features for $m = 10000$ and $\kappa = 1$. The FAPG method has a clear advantage over SeDuMi for large n, say $n \geq 10^3$. The middle panel illustrates the computation time with respect to the number m of samples for $n = 2000$ and $\kappa = 1$. The computation time is nearly independent of the number m of samples because the sizes of matrices $\Sigma_o^{1/2}$ $(o \in \{+, -\})$, which are used for computing the function $f(\boldsymbol{u})$ and the gradient $\nabla f(\boldsymbol{u})$, are $n \times n$. The right panel shows the computation time with respect to the parameter κ for $m = 10000$ and $n = 2000$. We can observe that a larger value of κ leads to more computation time for the FAPG method although it is still far more efficient than SeDuMi. The effect of a larger κ on the FAPG method could be because it gives a larger feasible region $S_{\rm MPM}$, which in turns leads to a larger distance between the initial point $\boldsymbol{u}^0$ and the optimal solution $\boldsymbol{u}^*$ as in the case of ν-SVM (the right panel of Figure 5).

5.4.2 Runtime Breakdown

Table 9 shows the runtime breakdown of the FAPG method for the artificial dataset with $(m, n, \kappa) = (10000, 1000, 1)$. As in the case of ν-SVM (Table 3), the computations of the gradient $\nabla f(\boldsymbol{u})$ and the function value $f(\boldsymbol{u})$ are the most time-consuming parts. Thus, computing 'bt' and $L_k\|T_{L_k}(\boldsymbol{u}^k) - \boldsymbol{u}^k\|$ periodically, which involves the computations of $f(\boldsymbol{u}^k)$ and/or $\nabla f(\boldsymbol{u}^k)$, is effective to reduce the total runtime. The projection $P_{S_{\rm MPM}}(\boldsymbol{u})$ for MM-MPM shown in Section 4.1.4 can be computed highly efficiently.

Function	% Time	Time	# Evals.	Time/Eval.
$\nabla f(\boldsymbol{u})$	75.9%	0.836	339	2.47.E-03
$f(\boldsymbol{u})$	14.2%	0.157	119	1.32.E-03
$P_{S_{\rm MPM}}(\boldsymbol{u})$	3.1%	0.034	383	8.88.E-05
Total Runtime:		1.102		

Table 9: Runtime Breakdown of FAPG Method for MM-MPM (sec.)

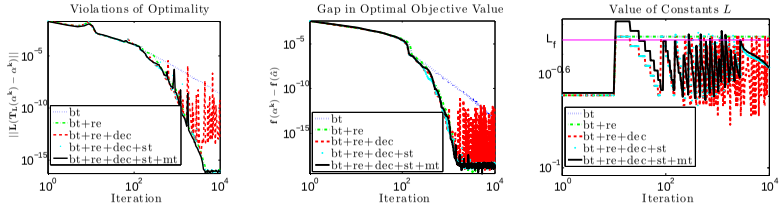


Figure 10: Effect of Each Strategy for the APG Method.

5.4.3 Effect of Each Strategy

Figure 10 illustrates the running history of FAPG with various practical strategies for the artificial dataset with $(m, n, \kappa) = (10000, 1000, 1)$. As in the case of ν-SVM, 're' is effective in reducing the violation of optimality $L_k\|T_{L_k}(\boldsymbol{u}^k) - \boldsymbol{u}^k\|$ and the value of $f(\boldsymbol{u}^k) - f(\hat{\boldsymbol{\alpha}})$. 'dec' seems to make the FAPG method to be unstable, but 'st' can stabilize it. 'bt+re+dec+st' and 'bt+re+dec+st+mt' decreased $L\|T_L(\boldsymbol{u}^k) - \boldsymbol{u}^k\|$ and $f(\boldsymbol{u}^k) - f(\hat{\boldsymbol{\alpha}})$ slightly faster than 'bt+re'.

In the right panel, we can see that the APG method uses values smaller than L_f for L_k in most iterations, where the Lipschitz constant L_f of the gradient $\nabla f(\boldsymbol{u}) = \widetilde{\Sigma}^\top(\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_- + \widetilde{\Sigma}\boldsymbol{u})$ is known to be the largest eigenvalue of the matrix $\widetilde{\Sigma}^\top\widetilde{\Sigma}$ (recall that $\widetilde{\Sigma} = [\Sigma_+^{1/2}, -\Sigma_-^{1/2}]$ and $\boldsymbol{u} = (\boldsymbol{u}_+, \boldsymbol{u}_-)$).

5.4.4 Benchmark Results

Table 10 shows the computational results for the benchmark datasets. We did the experiments by setting $\kappa = \kappa_{max}/2$, but MM-MPM could not be solved for $n \geq 20000$ because the sizes of the $n \times n$ matrices $\Sigma_o^{1/2}$ $(o \in \{+, -\})$ are extremely large.

The FAPG method was much faster than SeDuMi especially when the dimension is high, say $n \geq 2000$. Unlike for ν-SVM (Table 5), the FAPG method for MM-MPM sometimes led to larger values of L_k than the Lipschitz constant L_f. However, the average of the values of L_k is still smaller than L_f.

5.5 Classification Ability

Using the benchmark datasets, we compared the classification ability of classification models: ν-SVM, logistic regression, MM-MPM, and MM-FDA. Each dataset was randomly partitioned into 10 disjoint sets. We investigated the averages of the test accuracy using

					Computation Time		Values		
							L(FAPG)		
data	m	n	κ_{max}	κ	SeDuMi	FAPG (iter)	ave.	max.	L_f
a8a	22,696	123	9.52.E-01	4.76.E-01	0.664	**0.034** (25)	1.12.E+00	1.21.E+00	1.34.E+00
a9a	32,561	123	9.60.E-01	4.80.E-01	0.166	**0.010** (25)	1.12.E+00	1.21.E+00	1.34.E+00
australian	690	14	1.23.E+00	6.17.E-01	0.054	**0.006** (25)	1.85.E+00	2.00.E+00	2.06.E+00
breast-cancer	683	10	2.32.E+00	1.16.E+00	0.038	**0.005** (30)	1.01.E+00	1.10.E+00	1.17.E+00
cod-rna	59,535	8	1.41.E+00	7.05.E-01	0.068	**0.015** (115)	1.69.E-01	2.29.E-01	2.34.E-01
colon-cancer	62	2,000	1.85.E+00	9.24.E-01	311.329	**0.178** (157)	2.56.E+01	4.22.E+01	2.81.E+01
covtype	581,012	54	6.59.E-01	3.29.E-01	0.086	**0.016** (107)	8.31.E-01	1.10.E+00	1.07.E+00
diabetes	768	8	6.88.E-01	3.44.E-01	0.061	**0.003** (18)	3.70.E-01	3.86.E-01	4.79.E-01
duke	44	7,129	1.98.E+00	9.89.E-01	13459.0	**0.660** (291)	1.55.E+02	2.15.E+02	1.97.E+02
epsilon	400,000	2,000	1.16.E+00	5.82.E-01	361.0	**1.186** (217)	3.64.E-01	6.07.E-01	4.77.E-01
fourclass	862	2	7.22.E-01	3.61.E-01	0.069	**0.003** (12)	4.68.E-01	5.47.E-01	6.04.E-01
german.numer	1,000	24	6.52.E-01	3.26.E-01	0.063	**0.005** (33)	2.45.E+00	2.84.E+00	2.89.E+00
gisette	6,000	5,000	8.53.E+00	4.27.E+00	5438.2	**56.292** (1640)	9.40.E+01	2.29.E+02	1.14.E+02
heart	270	13	1.10.E+00	5.48.E-01	0.049	**0.004** (24)	1.85.E+00	1.98.E+00	2.08.E+00
ijcnn1	35,000	22	9.00.E-01	4.50.E-01	0.068	**0.013** (114)	4.06.E-01	6.52.E-01	3.04.E-01
ionosphere	351	34	1.30.E+00	6.48.E-01	0.084	**0.016** (110)	3.49.E+00	4.49.E+00	5.15.E+00
leu	38	7,129	2.11.E+00	1.05.E+00	12971.377	**0.514** (220)	1.37.E+02	2.68.E+02	1.46.E+02
liver-disorders	345	6	4.09.E-01	2.04.E-01	0.075	**0.009** (71)	3.05.E-01	4.08.E-01	3.78.E-01
madelon	2,000	500	6.50.E-01	3.25.E-01	1.964	**0.017** (43)	2.81.E-01	3.28.E-01	3.31.E-01
mushrooms	8,124	112	1.53.E+01	7.66.E+00	0.170	**0.166** (840)	2.38.E+00	3.63.E+00	2.92.E+00
news20.binary	19,996	1,355,191	**	**	**	** (**)	**	**	**
rcv1-origin	20,242	47,236	**	**	**	** (**)	**	**	**
real-sim	72,309	20,958	**	**	**	** (**)	**	**	**
skin-nonskin	245,057	3	1.63.E+00	8.14.E-01	0.066	**0.006** (50)	1.65.E-01	1.98.E-01	2.25.E-01
sonar	208	60	1.29.E+00	6.44.E-01	0.096	**0.020** (116)	3.47.E+00	4.60.E+00	5.07.E+00
splice	1,000	60	1.02.E+00	5.09.E-01	0.071	**0.009** (39)	2.39.E+00	2.84.E+00	2.92.E+00
svmguide1	3,089	4	1.26.E+00	6.29.E-01	0.070	**0.003** (28)	1.06.E-01	1.15.E-01	8.61.E-02
svmguide3	1,243	21	6.24.E-01	3.12.E-01	0.083	**0.012** (105)	5.90.E-01	8.91.E-01	5.82.E-01
url	2,396,130	3,231,961	**	**	**	** (**)	**	**	**
w7a	24,692	300	1.41.E+00	7.03.E-01	0.588	**0.036** (112)	1.42.E+00	2.21.E+00	1.82.E+00
w8a	49,749	300	1.39.E+00	6.97.E-01	0.592	**0.031** (108)	1.45.E+00	2.21.E+00	1.88.E+00

Table 10: Computational Results for MM-MPM with Linear Kernel. The best results are indicated by boldface. '**' means that the algorithm could not be computed due to out of memory.

dataset	ν-SVM (ν)	Logistic (C)	MM-MPM (κ)	MM-FDA (κ)
a8a	84.4% (0.37)	**84.6%** (0.5)	80.7% (0.94)	84.4% (1.32)
a9a	84.7% (0.36)	**84.8%** (0.1)	80.7% (0.95)	84.7% (1.33)
australian	85.7% (0.83)	**87.7%** (0.1)	86.1% (0.25)	87.5% (1.39)
breast-cancer	97.1% (0.07)	97.4% (0.5)	**97.5%** (2.08)	97.4% (0.31)
cod-rna	**93.9%** (0.28)	**93.9%** (4.5)	93.5% (0.42)	93.7% (0.40)
colon-cancer	**88.8%** (0.29)	87.1% (6.0)	87.1% (0.18)	87.1% (1.22)
covtype	**76.3%** (0.59)	75.6% (1.0)	75.6% (0.65)	75.5% (0.92)
diabetes	**77.3%** (0.54)	**77.3%** (3.5)	74.9% (0.61)	76.8% (0.88)
duke	**88.5%** (0.02)	86.0% (5.5)	**88.5%** (0.98)	**88.5%** (1.36)
epsilon	–	–	89.6% (1.04)	**89.7%** (1.48)
fourclass	77.7% (0.68)	78.5% (0.1)	72.7% (0.57)	**78.6%** (0.10)
german.numer	76.7% (0.54)	**77.4%** (0.1)	71.9% (0.39)	77.3% (0.55)
gisette	97.4% (0.11)	97.4% (1.0)	**97.9%** (3.41)	**97.9%** (4.83)
heart	84.1% (0.40)	**84.4%** (0.1)	84.1% (0.76)	84.1% (0.15)
ijcnn1	74.7% (0.19)	91.8% (20.0)	85.9% (0.89)	**91.0%** (0.64)
ionosphere	88.3% (0.21)	**90.0%** (5.0)	86.9% (0.39)	87.8% (0.85)
leu	**94.2%** (0.02)	**94.2%** (5.0)	**94.2%** (2.10)	**94.2%** (0.21)
liver-disorders	**68.1%** (0.76)	67.9% (6.0)	63.8% (0.20)	66.4% (0.52)
madelon	59.3% (0.91)	57.7% (0.1)	59.7% (0.13)	**60.0%** (0.09)
mushrooms	**100.0%** (0.01)	**100.0%** (6.0)	**100.0%** (9.19)	**100.0%** (6.21)
news20.binary	**97.1%** (0.21)	96.5% (20.0)	**	**
rcv1-origin	97.0% (0.11)	**97.1%** (20.0)	**	**
real-sim	97.5% (0.13)	**97.6%** (15.0)	**	**
skin-nonskin	93.7% (0.32)	92.4% (6.0)	93.5% (1.61)	**93.8%** (2.10)
sonar	**79.8%** (0.40)	78.8% (0.5)	**79.8%** (0.64)	77.9% (0.91)
splice	80.9% (0.50)	80.2% (3.0)	80.6% (0.61)	**81.0%** (0.20)
svmguide1	**95.4%** (0.13)	**95.4%** (20.0)	94.4% (1.12)	91.6% (1.56)
svmguide3	**82.5%** (0.41)	82.1% (15.0)	74.3% (0.55)	81.9% (0.69)
url	–	–	**	**
w7a	**98.5%** (0.04)	**98.5%** (4.0)	96.0% (1.26)	98.2% (1.76)
w8a	98.6% (0.04)	**98.7%** (17.5)	96.1% (1.25)	98.3% (1.73)

Table 11: Average Performance of Each Classification Model. The best results are indicated by boldface. '–' means that the cross-validation could not be done within 36000 sec. '**' means that it had run out of memory.

cross-validation over the 10 disjoint sets. We found the best parameter of the each classification model using grid search with cross-validation. The results are reported in Table 11. The model that shows the best performance varies with datasets. This implies the importance of finding a suitable classification model to a dataset in order to achieve a high prediction performance. Our algorithm is useful for the purpose; it provides a unified and efficient framework for solving various classification models.

6. Conclusion

In this work, we presented a unified classification model and provided a general algorithm for the model. We designed a fast accelerated proximal gradient (FAPG) method based on the original APG method in (Beck and Teboulle, 2009) to the model by devising efficient projection computations and effective heuristic acceleration strategies. Our unified

algorithm makes it easy to compare various models, because we can use the same algorithmic framework for the models by only changing the computation of projections. Thus, it provides a practical and useful tool for practitioners who are looking for the best model for a given dataset. Numerical experiments demonstrate the efficiency of our algorithm for large datasets. Indeed, our method often run faster than LIBLINEAR (Fan et al., 2008) especially for large-scale datasets such that $n > 2000$.

In this paper we showed a convergence property of our method for a general non-strongly convex function because our examples are mostly non-strongly convex. As a future work, we would like to see if the linear convergence for FAPG can be established for μ-strongly convex function and provide an adaptive parameter selection scheme even when μ is positive but unknown. In addition, we would like to investigate stochastic variants of our method. Recently, large scale learning has motivated researchers to develop stochastic gradient descent (SGD) methods. SGD approximates the gradient $\nabla\mathcal{L}(\cdot)$ of a loss function by a computationally inexpensive random variable whose expectation coincides with $\nabla\mathcal{L}(\cdot)$. More precisely, it assumes that the loss function is defined as the sum of loss of each sample, i.e., $\mathcal{L}(\widetilde{X}^\top \boldsymbol{w} - \boldsymbol{y}b) = \sum_{i=1}^m \ell(y_i(\boldsymbol{w}^\top \boldsymbol{x}_i - b))$, and approximates the *full* gradient $\nabla\mathcal{L}(\widetilde{X}^\top \boldsymbol{w} - \boldsymbol{y}b)$ by the gradient $\nabla\ell(y_i(\boldsymbol{w}^\top \boldsymbol{x}_i - b))$ of loss of a randomly chosen sample. PG and APG methods can be adapted in SGD, e.g., as (Nitanda, 2014; Defazio et al., 2014) and references there in. In particular, the stochastic dual coordinate ascent (SDCA) method (Shalev-Shwartz and Zhang, 2013, 2016) is a stochastic APG applied to the dual formulation, which is closely related to our unified classification model. While its framework would be useful to develop a stochastic variant of our FAPG presented in this paper, it cannot be directly applied to the unified classification model because ν-SVM, MM-MPM, and MM-FDA do not satisfy the assumption on $\mathcal{L}(\cdot)$: the loss function is defined as the sum of loss of each sample. It is an important future work to relax the assumption for stochastic methods and extend our FAPG to stochastic ones in order to solve even larger scale problems.

Acknowledgments

We would like to express our warm thanks to the reviewers, whose insightful comments have led to many substantial improvements.

Appendix

Appendix A. Derivation of the Dual Problem (10)

The primal problem (8) can be transformed as follows:

$$\begin{aligned}
&\min_{\boldsymbol{w},b}\{\mathcal{L}(\widetilde{A}^\top \boldsymbol{w} - b\boldsymbol{a}) \mid \|\boldsymbol{w}\|_p \le \lambda\} \\
&= \min_{\boldsymbol{w},b,\boldsymbol{z}}\{\mathcal{L}(\boldsymbol{z}) \mid \|\boldsymbol{w}\|_p \le \lambda,\ \boldsymbol{z} = \widetilde{A}^\top \boldsymbol{w} - b\boldsymbol{a}\} \\
&= \min_{\|\boldsymbol{w}\|_p \le \lambda, b, \boldsymbol{z}} \max_{\boldsymbol{\alpha}}\{\mathcal{L}(\boldsymbol{z}) + \boldsymbol{\alpha}^\top(\boldsymbol{z} - \widetilde{A}^\top \boldsymbol{w} + b\boldsymbol{a})\}.
\end{aligned}$$

Its dual problem (10) is derived as follows:

$$\begin{aligned}
&\max_{\boldsymbol{\alpha}} \min_{\|\boldsymbol{w}\|_p \le \lambda, b, \boldsymbol{z}}\{\mathcal{L}(\boldsymbol{z}) + \boldsymbol{\alpha}^\top(\boldsymbol{z} - \widetilde{A}^\top \boldsymbol{w} + b\boldsymbol{a})\} \\
&= -\min_{\boldsymbol{\alpha}}\Big\{\max_{\boldsymbol{z}}\big\{-\boldsymbol{\alpha}^\top \boldsymbol{z} - \mathcal{L}(\boldsymbol{z})\big\} + \max_{\|\boldsymbol{w}\|_p \le \lambda}\big\{\boldsymbol{\alpha}^\top \widetilde{A}^\top \boldsymbol{w}\big\} + \max_{b}\big\{\boldsymbol{\alpha}^\top \boldsymbol{a} b\big\}\Big\} \\
&= -\min_{\boldsymbol{\alpha}}\{\mathcal{L}^*(-\boldsymbol{\alpha}) + \lambda\|\widetilde{A}\boldsymbol{\alpha}\|_p^* \mid \boldsymbol{\alpha}^\top \boldsymbol{a} = 0\}.
\end{aligned}$$

Appendix B. Refined Bisection Algorithm

The computational cost of $h(\hat{\theta})$ in Algorithm 2 can be reduced by dividing the indices set M more finely as in (Kiwiel, 2008). Here we divide M into the following four disjoint sets for given θ^l and θ^u satisfying $\theta^l < \theta^u$:

$$\begin{aligned}
U &:= \{i \in M \mid \bar{\alpha}_i - \theta^u \ge u && \text{(i.e., } \alpha_i(\theta) = u, && \forall\theta \in [\theta^l, \theta^u])\} \\
L &:= \{i \in M \mid \bar{\alpha}_i - \theta^l \le l && \text{(i.e., } \alpha_i(\theta) = l, && \forall\theta \in [\theta^l, \theta^u])\} \\
C &:= \{i \in M \mid \bar{\alpha}_i - \theta^l < u,\ \bar{\alpha}_i - \theta^u > l && \text{(i.e., } \alpha_i(\theta) = \bar{\alpha}_i - \theta, && \forall\theta \in [\theta^l, \theta^u])\} \\
I &:= M\backslash(U \cup C \cup L) && \text{(}[\theta^l, \theta^u] \text{ contains a breakpoint of } \alpha_i(\theta)) &.
\end{aligned}$$

If $I = \phi$, then solving

$$\theta = \Big(|U|u + |L|l + \sum_{i \in C} \bar{\alpha}_i - r\Big)/|C|$$

obtains an exact solution. If $I \neq \phi$, then we also divide I into the following five disjoint sets for given $\hat{\theta} \in (\theta^l, \theta^u)$:

$$\begin{aligned}
I^U &= \{i \in I \mid \bar{\alpha}_i - \hat{\theta} \ge u, && \text{(i.e., } \alpha_i(\theta) = u && \forall\theta \in [\theta^l, \hat{\theta}])\} \\
I^L &= \{i \in I \mid \bar{\alpha}_i - \hat{\theta} \le l, && \text{(i.e., } \alpha_i(\theta) = l && \forall\theta \in [\hat{\theta}, \theta^u])\} \\
I^{C_u} &= \{i \in I \mid \bar{\alpha}_i - \theta^l < u,\ \bar{\alpha}_i - \hat{\theta} > l, && \text{(i.e., } \alpha_i(\theta) = \bar{\alpha}_i - \theta && \forall\theta \in [\theta^l, \hat{\theta}])\} \\
I^{C_l} &= \{i \in I \mid \bar{\alpha}_i - \hat{\theta} < u,\ \bar{\alpha}_i - \theta^u > l, && \text{(i.e., } \alpha_i(\theta) = \bar{\alpha}_i - \theta && \forall\theta \in [\hat{\theta}, \theta^u])\} \\
I^I &= I\backslash(I^U \cup I^{C_u} \cup I^{C_l} \cup I^L)
\end{aligned}$$

Then we have

$$\begin{aligned}
h(\hat{\theta}) = \sum_{i \in M} \alpha_i(\hat{\theta}) = &\underbrace{|U|u}_{s_u} + \underbrace{|I^U|u}_{\Delta s_u} + \underbrace{|L|l}_{s_l} + \underbrace{|I^L|l}_{\Delta s_l} \\
&+ \underbrace{\sum_{i \in I^C} \alpha_i}_{s_c} + \underbrace{\sum_{i \in I^{C_u}} \alpha_i}_{\Delta s_{c_u}} + \underbrace{\sum_{i \in I^{C_l}} \alpha_i}_{\Delta s_{c_l}} + \sum_{i \in I^I} \alpha_i - |C \cup I^{C_u} \cup I^{C_l} \cup I^I|\hat{\theta}.
\end{aligned}$$

By leveraging on the structure of h, the refined bisection method for (18) can be described as Algorithm 5.

Algorithm 5 Refined Bisection Algorithm for (18)

INPUT: $\bar{\boldsymbol{\alpha}}$, r, $\boldsymbol{l}$, $\boldsymbol{u}$, $\epsilon' > 0$ OUTPUT: $\boldsymbol{\alpha}$

INITIALIZE: $I \leftarrow M$, $s_u \leftarrow s_l \leftarrow s_c \leftarrow 0$, $\theta^u \leftarrow \bar{\alpha}_{\max} - \frac{r}{m}$, $\theta^l \leftarrow \bar{\alpha}_{\min} - \frac{r}{m}$ # Step 1

while $|\theta^u - \theta^l| > \epsilon'$ **do**

$\hat{\theta} \leftarrow \frac{\theta^u + \theta^l}{2}$ # Step 2

$\hat{u} \leftarrow u + \hat{\theta}, \quad \hat{l} \leftarrow l + \hat{\theta}$

$I^U \leftarrow \{i \in I \mid \bar{\alpha}_i \geq \hat{u}\}, \quad I^L \leftarrow \{i \in I \mid \bar{\alpha}_i \leq \hat{l}\}$ # Step 3

$u^u \leftarrow u + \theta^u, \quad l^l \leftarrow l + \theta^l$

$I^{C_u} \leftarrow \{i \in I \mid \bar{\alpha}_i > l^l,\ \bar{\alpha}_i < \hat{u}\}, \quad I^{C_l} \leftarrow \{i \in I \mid \bar{\alpha}_i < u^u,\ \bar{\alpha}_i > \hat{l}\}$

$I^I \leftarrow I \backslash (I^U \cup I^{C_u} \cup I^{C_l} \cup I^L)$

$\Delta s_u \leftarrow |I^U| u, \quad \Delta s_l \leftarrow |I^L| l$

$\Delta s_{c_u} \leftarrow \sum_{i \in I^{C_u}} \alpha_i, \quad \Delta s_{c_l} \leftarrow \sum_{i \in I^{C_l}} \alpha_i$

$val \leftarrow s_u + \Delta s_u + s_l + \Delta s_l + s_c + \Delta s_{c_u} + \Delta s_{c_l} + \sum_{i \in I^I} \alpha_i - |C \cup I^{C_u} \cup I^{C_l} \cup I^I| \hat{\theta}$

if $val < r$ **then** # Step 4

$\theta^u \leftarrow \hat{\theta}, \quad I \leftarrow I^I \cup I^{C_l} \cup I^L$

$s_u \leftarrow s_u + \Delta s_u, \quad s_c \leftarrow s_c + \Delta s_{c_u}$

else if $val > r$ **then**

$\theta^l \leftarrow \hat{\theta}, \quad I \leftarrow I^I \cup I^{C_u} \cup I^U$

$s_l \leftarrow s_l + \Delta s_l, \quad s_c \leftarrow s_c + \Delta s_{c_l}$

else

break

end if

if $I == \phi$ **then**

$\hat{\theta} \leftarrow (s_u + s_l + s_c - r)/|C|$

break

end if

end while

$\alpha_i \leftarrow \alpha_i(\hat{\theta}),\ \forall i \in M$ # Step 5

Appendix C. Proof of Theorem 7

We show the iteration complexity $O\big((W(k)/k)^2\big)$ of Algorithm 3, where $W(z)$ is the inverse function of $f(x) = x\exp(x)$ and called Lambert's W function (Corless et al., 1996). $W(z)$ is single-valued for $z \geq 0$, but multi-valued for $z < 0$. For $z \geq 0$, $W(z)$ is monotonically increasing, but diverges slower than $\log(z)$. It is known that $W(z)$ cannot be expressed in terms of elementary functions.

Here we improve the lower bound (24) of

$$\max\{\bar{k}_1, \bar{k}_2, \ldots, \bar{k}_j, k_{j+1}\}.$$

Since $\bar{k}_i \geq K_i \geq 2^i$ for all $i \in \{1, 2, \ldots, j\}$, we have

$$\max\{\bar{k}_1, \bar{k}_2, \ldots, \bar{k}_j, k_{j+1}\} \geq \max\Big\{\frac{k}{j+1}, 2^j\Big\}.$$

$\frac{k}{j+1}$ is monotonically decreasing and 2^j is monotonically increasing with respect to j. Thus the right-hand side is minimized at $j = \hat{j}$ such that

$$\frac{k}{\hat{j}+1} = 2^{\hat{j}}.$$

The above equation implies that

$$\begin{aligned}
&\frac{2k}{\hat{j}+1} = \exp\big((\hat{j}+1)\ln 2\big) \\
&\Rightarrow 2k\ln 2 = (\hat{j}+1)\ln 2 \exp\big((\hat{j}+1)\ln 2\big) \\
&\Rightarrow W(2k\ln 2) = (\hat{j}+1)\ln 2 \\
&\Rightarrow \hat{j} = \frac{W(2k\ln 2)}{\ln 2} - 1,
\end{aligned}$$

which yields

$$\max\Big\{\frac{k}{j+1}, 2^j\Big\} \geq \frac{k\ln 2}{W(2k\ln 2)}.$$

This leads to

$$F(\boldsymbol{\alpha}^{(j,k_{j+1})}) - F(\boldsymbol{\alpha}^*) \leq 2\eta_u L_f R^2 \Big(\frac{W(2k\ln 2)}{k\ln 2 - W(2k\ln 2)}\Big)^2, \quad \forall k \geq 3.$$

Appendix D. Application of FAPG to ℓ_1-regularized models

The ℓ_1-regularized logistic regression:

$$\min_{\boldsymbol{w}} C\sum_{i=1}^m \log(1 + \exp(-y_i \boldsymbol{w}^\top \boldsymbol{x}_i)) + \|\boldsymbol{w}\|_1 \tag{27}$$

and the ℓ_1-regularized ℓ_2-loss SVM:

$$\min_{\boldsymbol{w}} C\sum_{i=1}^m \big(\max\{0, 1 - y_i \boldsymbol{w}^\top \boldsymbol{x}_i\}\big)^2 + \|\boldsymbol{w}\|_1 \tag{28}$$

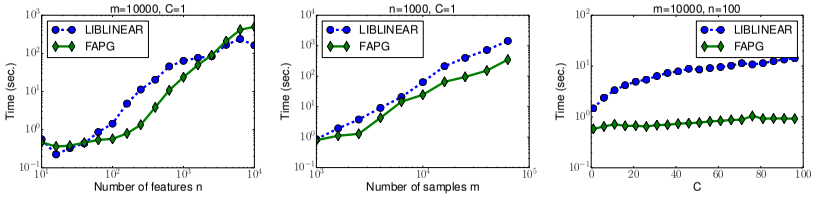


Figure 11: Computation Time for ℓ_1-regularized Logistic Regression (27).

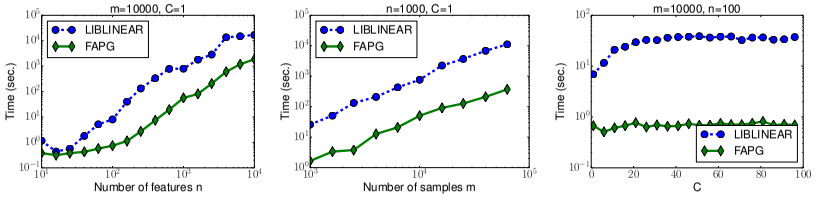


Figure 12: Computation Time for ℓ_1-regularized ℓ_2-loss SVM (28).

have the form of (2), where $C > 0$ is a hyperparameter. Letting $g(\cdot) = \|\cdot\|_1$, we have $\left(\mathrm{prox}_{g,L}(\boldsymbol{w})\right)_i = \mathrm{sign}(w_i)\max\{0, |w_i| - L\}$ $(i = 1, 2, \ldots, m)$ which is known as the soft-thresholding operator.

We applied the practical FAPG (Algorithm 4) and LIBLINEAR (Fan et al., 2008) to (27) and (28) using the artificial datasets which is generated as described in Section 5. We set the initial point $\boldsymbol{w}^0$ to the origin $\mathbf{0}$ and the tolerance ϵ to 10^{-6}. For all datasets, FAPG found solutions with better objective value than LIBLINEAR does. Figures 11 and 12 show the computation time of FAPG and LIBLINEAR. FAPG was highly competitive to and numerically more stable than LIBLINEAR with respect to the changes of the hyperparameter C.

References

U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 96(12):6745–6750, 1999.

K. Bache and M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`. Accessed 6 February 2016.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

C. Bhattacharyya. Second order cone programming formulations for feature selection. *Journal of Machine Learning Research*, 5:1417–1433, 2004.

C.-C. Chang and C.-J. Lin. LIBSVM : A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. *Advances in Computational Mathematics*, 5(1):329–359, 1996.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

D. R. Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, 20(2):215–242, 1958.

A. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems 27*, pages 1646–1654. Curran Associates, Inc., 2014.

J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the L1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, pages 272–279, Helsinki, Finland, 2008.

R.-E. Fan, K.-W. Chang, C.-J Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear : A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. Software available at `http://www.csie.ntu.edu.tw/~cjlin/liblinear`.

T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, 286(5439):531–537, 1999.

J. Gotoh and A. Takeda. A linear classification model based on conditional geometric score. *Pacific Journal of Optimization*, 1(2):277–296, 2005.

I. Guyon, S. Gunn, A. B. Hur, and G. Dror. Result analysis of the nips 2003 feature selection challenge. In *Advances in Neural Information Processing Systems 17*, pages 545–552, 2005.

K. Helgason, J. Kennington, and H. Lall. A polynomially bounded algorithm for a singly constrained quadratic program. *Mathematical Programming*, 18:338–343, 1980.

T.-K. Ho and E. M. Kleinberg. Building projectable classifiers of arbitrary complexity. In *Proceedings of the 13th International Conference on Pattern Recognition*, volume 2, pages 880–885. IEEE, 1996.

C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *Proceedings of the 25th International Conference on Machine learning*, pages 408–415. ACM, 2008.

C.-W. Hsu, C.-C. Chang, and C.-J. Lin. A practical guide to support vector classification. Technical report, Department of Computer Science, National Taiwan University, 2003.

K. Jiang, D.-F. Sun, and K.-C. Toh. An inexact accelerated proximal gradient method for large scale linearly constrained convex sdp. *SIAM Journal on Optimization*, 22:1042–1064, 2012.

S. S. Keerthi and D. DeCoste. A modified finite Newton method for fast solution of large scale linear SVMs. *Journal of Machine Learning Research*, 6:341–361, 2005.

M. Kitamura, A. Takeda, and S. Iwata. Exact SVM training by Wolfe's minimum norm point algorithm. In *2014 IEEE International Workshop on Machine Learning for Signal Processing*, pages 1–6, Sept 2014.

K. C. Kiwiel. Breakpoint searching algorithms for the continuous quadratic knapsack problem. *Mathematical Programming*, 112:473–491, 2008.

D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

Q. Lin and L. Xiao. An adaptive accelerated proximal gradient method and its homotopy continuation for sparse optimization. *Computational Optimization and Applications*, 60 (3):633–674, April 2015.

J. Ma, L. K. Saul, S. Savage, and G. M. Voelker. Identifying suspicious URLs: an application of large-scale online learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 681–688. ACM, 2009.

J. S. Marron, M. J. Todd, and J. Ahn. Distance-weighted discrimination. *Journal of the American Statistical Association*, 102(480):1267–1271, 2007.

A. McCallum. SRAA: simulated/real/aviation/auto UseNet data. URL `http://people.cs.umass.edu/~mccallum/data.html`. Accessed 6 February 2016.

R. D. C. Monteiro, C. Ortiz, and B. F. Svaiter. An adaptive accelerated first-order method for convex optimization. *Computational Optimization and Applications*, 64:31–73, 2016.

J. S. Nath and C. Bhattacharyya. Maximum margin classifiers with specified false positive and false negative error rates. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 35–46. SIAM, 2007.

Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

Y. E. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 152:127–152, 2005.

A. Nitanda. Stochastic proximal gradient descent with acceleration techniques. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1574–1582. Curran Associates, Inc., 2014.

B. O'Donoghue and E. Candès. Adaptive restart for accelerated gradient schemes. *Foundations of Computational Mathematics*, 15:715–732, 2015.

N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1 (3):127–239, 2014.

J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods - Support Vector Learning*. MIT Press, January 1998.

D. Prokhorov. IJCNN 2001 neural network competition. *Slide presentation in IJCNN'01*, 2001. URL `http://www.geocities.com/ijcnn/nnc_ijcnn01.pdf`. Accessed 6 February 2016.

R. T. Rockafellar and S. Uryasev. Optimization of conditional value-at-risk. *Journal of Risk*, 2(3):21–41, 2000.

K. Scheinberg, D. Goldfarb, and X. Bai. Fast first-order methods for composite convex optimization with backtracking. *Foundations of Computational Mathematics*, 14(3):389–417, Jun 2014.

B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.

B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett. New support vector algorithms. *Neural Computation*, 12(5):1207–1245, May 2000.

S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss. *Journal of Machine Learning Research*, 14:567–599, 2013.

S. Shalev-Shwartz and T. Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. *Mathematical Programming*, 155(1):105–145, 2016.

S. Sonnenburg, V. Franc, E. Yom-Tov, and M. Sebag. Pascal large scale learning challenge, 2008. URL `http://largescale.ml.tu-berlin.de`. Accessed 6 February 2016.

J.F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11–12:625–653, 1999. Version 1.05 available from `http://fewcal.kub.nl/sturm`.

W. Su, S. Boyd, and E. J. Candès. A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. In *Advances in Neural Information Processing Systems*, pages 2510–2518, Mar 2014.

A. Takeda, H. Mitsugi, and T. Kanamori. A unified classification model based on robust optimization. *Neural computation*, 25(3):759–804, 2013.

A. V. Uzilov, J. M. Keegan, and D. H. Mathews. Detection of non-coding RNAs on the basis of predicted secondary structure formation free energy change. *BMC bioinformatics*, 7 (1):173, 2006.

M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J. A. Olson, J. R. Marks, and J. R. Nevins. Predicting the clinical status of human breast cancer by using gene expression profiles. *Proceedings of the National Academy of Sciences*, 98(20): 11462–11467, 2001.

H.-F. Yu, F.-L. Huang, and C.-J. Lin. Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, 85(1):41–75, 2011.

T. Zhou, D. Tao, and X. Wu. NESVM: A fast gradient method for support vector machines. In *2010 IEEE 10th International Conference on Data Mining*, pages 679–688, December 2010.

Journal of Machine Learning Research 18 (2017) 1-5 Submitted 7/16; Published 1/17

Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning

Guillaume Lemaître guillaume.lemaitre@inria.fr
Parietal team, Inria, CEA, Université Paris-Saclay
1 Rue Honoré d'Estienne d'Orves, 91120 Palaiseau, France

Fernando Nogueira fmfnogueira@gmail.com
ShoppeAI
488 Wellington Street West, Suite 304, Toronto, Ontario M5V 1E3, Canada

Christos K. Aridas char@upatras.gr
Computational Intelligence Laboratory
Department of Mathematics
University of Patras
GR-26504 Patras, Greece

Editor: Geoff Holmes

Abstract

`imbalanced-learn` is an open-source python toolbox aiming at providing a wide range of methods to cope with the problem of imbalanced dataset frequently encountered in machine learning and pattern recognition. The implemented state-of-the-art methods can be categorized into 4 groups: (i) under-sampling, (ii) over-sampling, (iii) combination of over- and under-sampling, and (iv) ensemble learning methods. The proposed toolbox depends only on `numpy`, `scipy`, and `scikit-learn` and is distributed under MIT license. Furthermore, it is fully compatible with `scikit-learn` and is part of the `scikit-learn-contrib` supported project. Documentation, unit tests as well as integration tests are provided to ease usage and contribution. Source code, binaries, and documentation can be downloaded from `https://github.com/scikit-learn-contrib/imbalanced-learn`.

Keywords: Imbalanced Dataset, Over-Sampling, Under-Sampling, Ensemble Learning, Machine Learning, Python.

1. Introduction

Real world datasets commonly show the particularity to have a number of samples of a given class under-represented compared to other classes. This imbalance gives rise to the "class imbalance" problem (Prati et al., 2009) (or "curse of imbalanced datasets") which is the problem of learning a concept from the class that has a small number of samples.

The class imbalance problem has been encountered in multiple areas such as telecommunication managements, bioinformatics, fraud detection, and medical diagnosis, and has been considered one of the top 10 problems in data mining and pattern recognition (Yang and Wu, 2006; Rastgoo et al., 2016). Imbalanced data substantially compromises the learning process, since most of the standard machine learning algorithms expect balanced class distribution or an equal misclassification cost (He and Garcia, 2009). For this reason, several approaches have been specifically proposed to handle such datasets. Some of these methods have been implemented mainly in R language (Torgo, 2010; Kuhn, 2015; Dal Pozzolo et al., 2013). Up to our knowledge, there is no python toolbox allowing such processing while cutting edge machine learning toolboxes are available (Pedregosa et al., 2011; Sonnenburg et al., 2010).

In this paper, we present the `imbalanced-learn` API, *a python toolbox to tackle the curse of imbalanced datasets in machine learning*. The following sections present the project vision, a snapshot of the API, an overview of the implemented methods, and finally, we conclude this work by including future functionalities for the `imbalanced-learn` API.

2. Project management

Quality assurance In order to ensure code quality, a set of unit tests is provided leading to a coverage of 99 % for the release 0.2 of the toolbox. Furthermore, the code consistency is ensured by following `PEP8` standards and each new contribution is automatically checked through landscape, which provides metrics related to code quality.
Continuous integration To allow both the user and the developer to either use or contribute to this toolbox, Travis CI is used to easily integrate new code and ensure back-compatibility.
Community-based development All the development is performed in a collaborative manner. Tools such as git, GitHub, and gitter are used to ease collaborative programming, issue tracking, code integration, and idea discussions.
Documentation A consistent API documentation is provided using `sphinx` and `numpydoc`. An additional installation guide and examples are also provided and centralized on GitHub[1].
Project relevance At the edition time, the repository is visited no less than 2,000 times per week, attracting about 300 unique visitors per week. Additionally, the toolbox is supported by `scikit-learn` through the `scikit-learn-contrib` projects.

3. Implementation design

```python
from sklearn.datasets import make_classification
from sklearn.decomposition import PCA
from imblearn.over_sampling import SMOTE

# Generate the dataset
X, y = make_classification(n_classes=2, weights=[0.1, 0.9],
                           n_features=20, n_samples=5000)

# Apply the SMOTE over-sampling
sm = SMOTE(ratio='auto', kind='regular')
X_resampled, y_resampled = sm.fit_sample(X, y)
```

Listing 1: Code snippet to over-sample a dataset using SMOTE.

The implementation relies on `numpy`, `scipy`, and `scikit-learn`. Each sampler class implements three main methods inspired from the `scikit-learn` API: (i) `fit` computes several statistics which are later needed to resample the data into a balanced set; (ii)

1. `https://github.com/scikit-learn-contrib/imbalanced-learn`

©2017 Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v18/16-365.html`.

Method	Over-sampling		Under-sampling	
	Binary	Mutli-class	Binary	Multiclass
ADASYN (He et al., 2008)	✓	✗	✗	✗
SMOTE (Chawla et al., 2002; Han et al., 2005; Nguyen et al., 2011)	✓	✗	✗	✗
ROS	✓	✓	✗	✗
CC	✗	✗	✓	✓
CNN (Hart, 1968)	✗	✗	✓	✓
ENN (Wilson, 1972)	✗	✗	✓	✓
RENN	✗	✗	✓	✓
AKNN	✗	✗	✓	✓
NM (Mani and Zhang, 2003)	✗	✗	✓	✓
NCL (Laurikkala, 2001)	✗	✗	✓	✓
OSS (Kubat et al., 1997)	✗	✗	✓	✓
RUS	✗	✗	✓	✓
IHT (Smith et al., 2014)	✗	✗	✓	✗
TL (Tomek, 1976)	✗	✗	✓	✗
BC (Liu et al., 2009)	✗	✗	✓	✗
EE (Liu et al., 2009)	✗	✗	✓	✓
SMOTE + ENN (Batista et al., 2003)	✓	✗	✓	✗
SMOTE + TL (Batista et al., 2003)	✓	✗	✓	✗

`sample` performs the sampling and returns the data with the desired balancing ratio; and (iii) `fit_sample` is equivalent to calling the method `fit` followed by the method `sample`. A class `Pipeline` is inherited from the `scikit-learn` toolbox to automatically combine `samplers`, `transformers`, and `estimators`. Additionally, we provide some specific state-of-the-art metrics to evaluate classification performance.

4. Implemented methods

The `imbalanced-learn` toolbox provides four different strategies to tackle the problem of imbalanced dataset: (i) under-sampling, (ii) over-sampling, (iii) a combination of both, and (iv) ensemble learning. The following subsections give an overview of the techniques implemented.

4.1 Notation and background

Let χ be an imbalanced dataset with χ_{min} and χ_{maj} being the subset of samples belonging to the minority and majority class, respectively. The balancing ratio of the dataset χ is defined as:

$$r_\chi = \frac{|\chi_{min}|}{|\chi_{maj}|} \ , \tag{1}$$

where $|\cdot|$ denotes the cardinality of a set. The balancing process is equivalent to resample χ into a new dataset χ_{res} such that $r_\chi > r_{\chi_{res}}$.

Under-sampling Under-sampling refers to the process of reducing the number of samples in χ_{maj}. The implemented methods can be categorized into 2 groups: (i) fixed under-sampling and (ii) cleaning under-sampling. *Fixed under-sampling* refer to the methods which perform under-sampling to obtain the appropriate balancing ratio $r_{\chi_{res}}$. Contrary to the previous methods, *cleaning under-sampling* do not allow to reach specifically the balancing ratio $r_{\chi_{res}}$, but rather clean the feature space based on some empirical criteria.

Over-sampling Contrary to under-sampling, data balancing can be performed by over-sampling such that new samples are generated in χ_{min} to reach the balancing ratio $r_{\chi_{res}}$.

Combination of over- and under-sampling Over-sampling can lead to over-fitting which can be avoided by applying cleaning under-sampling methods (Prati et al., 2009).

Ensemble learning Under-sampling methods imply that samples of the majority class are lost during the balancing procedure. Ensemble methods offer an alternative to use most of the samples. In fact, an ensemble of balanced sets is created and used to later train any classifier.

5. Future plans and conclusion

In this paper, we shortly presented the foundations of the `imbalanced-learn` toolbox vision and API. As avenues for future works, additional methods based on prototype/instance selection, generation, and reduction will be added as well as additional user guides.

References

G. E. Batista, A. L. Bazzan, and M. C. Monard. Balancing training data for automated annotation of keywords: a case study. In *WOB*, pages 10–18, 2003.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, pages 321–357, 2002.

A. Dal Pozzolo, O. Caelen, S. Waterschoot, and G. Bontempi. Racing for unbalanced methods selection. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 24–31. Springer, 2013.

H. Han, W.-Y. Wang, and B.-H. Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International Conference on Intelligent Computing*, pages 878–887. Springer, 2005.

P. Hart. The condensed nearest neighbor rule. *Information Theory, IEEE Transactions on*, 14(3):515–516, May 1968.

H. He and E. Garcia. Learning from imbalanced data. *Knowledge and Data Engineering, IEEE Transactions on*, 21(9):1263–1284, 2009.

Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1322–1328. IEEE, 2008.

M. Kubat, S. Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *International Conference in Machine Learning*, volume 97, pages 179–186. Nashville, USA, 1997.

M. Kuhn. Caret: classification and regression training. *Astrophysics Source Code Library*, 1:05003, 2015.

J. Laurikkala. *Improving identification of difficult small classes by balancing class distribution*. Springer, 2001.

X.-Y. Liu, J. Wu, and Z.-H. Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2009.

I. Mani and I. Zhang. knn approach to unbalanced data distributions: a case study involving information extraction. In *Proceedings of Workshop on Learning from Imbalanced Datasets*, 2003.

H. M. Nguyen, E. W. Cooper, and K. Kamei. Borderline over-sampling for imbalanced data classification. *International Journal of Knowledge Engineering and Soft Data Paradigms*, 3(1):4–21, 2011.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

R. C. Prati, G. E. Batista, and M. C. Monard. Data mining with imbalanced class distributions: concepts and methods. In *Indian International Conference Artificial Intelligence*, pages 359–376, 2009.

M. Rastgoo, G. Lemaitre, J. Massich, O. Morel, F. Marzani, R. Garcia, and F. Meriaudeau. Tackling the problem of data imbalancing for melanoma classification. In *Bioimaging*, 2016.

M. R. Smith, T. Martinez, and C. Giraud-Carrier. An instance level analysis of data complexity. *Machine learning*, 95(2):225–256, 2014.

S. C. Sonnenburg, S. Henschel, C. Widmer, J. Behr, A. Zien, F. de Bona, A. Binder, C. Gehl, V. Franc, et al. The SHOGUN machine learning toolbox. *Journal of Machine Learning Research*, 11(Jun):1799–1802, 2010.

I. Tomek. Two modifications of CNN. *Systems, Man, and Cybernetics, IEEE Transactions on*, 6:769–772, 1976.

L. Torgo. *Data mining with R: learning with case studies*. Chapman & Hall/CRC, 2010.

D. L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *Systems, Man and Cybernetics, IEEE Transactions on*, (3):408–421, 1972.

Q. Yang and X. Wu. 10 challenging problems in data mining research. *International Journal of Information Technology & Decision Making*, 5(04):597–604, 2006.

Journal of Machine Learning Research 18 (2017) 1-65 Submitted 11/14; Revised 10/16; Published 4/17

Information-Geometric Optimization Algorithms: A Unifying Picture via Invariance Principles

Yann Ollivier YANN.OLLIVIER@LRI.FR
CNRS & LRI (UMR 8623), Université Paris-Saclay
91405 Orsay, France

Ludovic Arnold ARNOLD@LRI.FR
Univ. Paris-Sud, LRI
91405 Orsay, France

Anne Auger ANNE.AUGER@INRIA.FR
Nikolaus Hansen NIKOLAUS.HANSEN@INRIA.FR
Inria & CMAP, Ecole polytechnique
91128 Palaiseau, France

Editor: Una-May O'Reilly

Abstract

We present a canonical way to turn any smooth parametric family of probability distributions on an arbitrary search space X into a continuous-time black-box optimization method on X, the *information-geometric optimization* (IGO) method. Invariance as a major design principle keeps the number of arbitrary choices to a minimum. The resulting *IGO flow* is the flow of an ordinary differential equation conducting the natural gradient ascent of an adaptive, time-dependent transformation of the objective function. It makes no particular assumptions on the objective function to be optimized.

The IGO method produces explicit IGO algorithms through time discretization. It naturally recovers versions of known algorithms and offers a systematic way to derive new ones. In continuous search spaces, IGO algorithms take a form related to natural evolution strategies (NES). The cross-entropy method is recovered in a particular case with a large time step, and can be extended into a smoothed, parametrization-independent maximum likelihood update (IGO-ML). When applied to the family of Gaussian distributions on $\mathbb{R}^d$, the IGO framework recovers a version of the well-known CMA-ES algorithm and of xNES. For the family of Bernoulli distributions on $\{0,1\}^d$, we recover the seminal PBIL algorithm and cGA. For the distributions of restricted Boltzmann machines, we naturally obtain a novel algorithm for discrete optimization on $\{0,1\}^d$. All these algorithms are natural instances of, and unified under, the single information-geometric optimization framework.

The IGO method achieves, thanks to its intrinsic formulation, maximal invariance properties: invariance under reparametrization of the search space X, under a change of parameters of the probability distribution, and under increasing transformation of the function to be optimized. The latter is achieved through an adaptive, quantile-based formulation of the objective.

Theoretical considerations strongly suggest that IGO algorithms are essentially characterized by a minimal change of the distribution over time. Therefore they have minimal loss in diversity through the course of optimization, provided the initial diversity is high. First experiments using restricted Boltzmann machines confirm this insight. As a simple consequence, IGO seems to provide, from information theory, an elegant way to simultaneously explore several valleys of a fitness landscape in a single run.

Keywords: black-box optimization, stochastic optimization, randomized optimization, natural gradient, invariance, evolution strategy, information-geometric optimization

©2017 Yann Ollivier, Ludovic Arnold, Anne Auger and Nikolaus Hansen.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/14-467.html.

1. Introduction

Optimization problems are at the core of many disciplines. Given an objective function $f : X \to \mathbb{R}$, to be optimized on some space X, the goal of black-box optimization is to find solutions $x \in X$ with small (in the case of minimization) value $f(x)$, using the least number of calls to the function f. In a *black-box* scenario, knowledge about the function f is restricted to the handling of a device (e.g., a simulation code) that delivers the value $f(x)$ for any input $x \in X$. The search space X may be finite, discrete infinite, or continuous. However, optimization algorithms are often designed for a specific type of search space, exploiting its specific structure.

One major design principle in general and in optimization in particular is related to *invariance*. Invariance extends performance observed on a single function to an entire associated invariance class, that is, it generalizes from a single problem to a class of problems. Thus it hopefully provides better robustness w.r.t. changes in the presentation of a problem. For continuous search spaces, invariance under translation of the coordinate system is standard in optimization. Invariance under general affine-linear changes of the coordinates has been—we believe—one of the keys to the success of the Nelder-Mead *Downhill Simplex method* (Nelder and Mead, 1965), of quasi-Newton methods (Deuflhard, 2011) and of the *covariance matrix adaptation evolution strategy*, CMA-ES (Hansen and Ostermeier, 2001; Hansen and Auger, 2014). While these relate to transformations in the search space, another important invariance concerns the application of monotonically increasing transformations to f, so that it is indifferent whether the function f, f^3 or $f \times |f|^{-2/3}$ is minimized. This way some non-convex or non-smooth functions can be as "easily" optimised as convex ones. Invariance under f-transformation is not uncommon, e.g., for evolution strategies (Schwefel, 1995) or pattern search methods (Hooke and Jeeves, 1961; Torczon, 1997; Nelder and Mead, 1965); however it has not always been recognized as an attractive feature.

Many stochastic optimization methods have been proposed to tackle black-box optimization. The underlying (often hidden) principle of these stochastic methods is to iteratively update a probability distribution P_θ defined on X, parametrized by a set of parameters θ. At a given iteration, the distribution P_θ represents, loosely speaking, the current belief about where solutions with the smallest values of the function f may lie. Over time, P_θ is expected to concentrate around the minima of f. The update of the distribution involves querying the function with points sampled from the current probability distribution P_θ. Although implicit in the presentation of many stochastic optimization algorithms, this is the explicit setting for the wide family of *estimation of distribution algorithms* (EDA) (Larranaga and Lozano, 2002; Baluja and Caruana, 1995; Pelikan et al., 2002). Updates of the probability distribution often rely on heuristics (nevertheless in Toussaint 2004 the possible interest of information geometry to exploit the structure of probability distributions for designing better grounded heuristics is pointed out). In addition, in the EDA setting we can distinguish two theoretically founded approaches to update P_θ.

First, the *cross-entropy* method consists in taking θ minimizing the Kullback–Leibler divergence between P_θ and the indicator of the best points according to f (de Boer et al., 2005).

Second, one can transfer the objective function f to the space of parameters θ by taking the average of f under P_θ, seen as a function of θ. This average is a new function from an Euclidian space to $\mathbb{R}$ and is minimal when P_θ is concentrated on minima of f. Consequently, θ can be updated by following a gradient descent of this function with respect to θ. This has been done in various situations such as $X = \{0,1\}^d$ and the family of Bernoulli measures (Berny, 2000a) or of Boltzmann machines (Berny, 2002), or on $X = \mathbb{R}^d$ for the family of Gaussian distributions (Berny, 2000b; Gallagher and Frean, 2005) or the exponential family also using second order information (Zhou and Hu, 2014; E. Zhou, 2016).

However, taking the ordinary gradient with respect to θ depends on the precise way a parameter θ is chosen to represent the distribution P_θ, and does not take advantage of the Riemannian metric structure of families of probability distributions. In the context of machine learning, Amari noted the shortcomings of the ordinary gradient for families of probability distributions (Amari, 1998) and proposed instead to use the natural gradient with respect to the Fisher metric (Rao, 1945; Jeffreys, 1946; Amari and Nagaoka, 2000). In the context of optimization, this natural gradient has been used for exponential families on $X = \{0,1\}^d$ (Malagò et al., 2008, 2011) and for the family of Gaussian distributions on $X = \mathbb{R}^d$ with so-called natural evolution strategies (NES, Wierstra et al. 2008; Sun et al. 2009; Glasmachers et al. 2010; Wierstra et al. 2014).

However, none of the previous attempts using gradient updates captures the invariance under increasing transformations of the objective function, which is instead, in some cases, enforced *a posteriori* with heuristic arguments.

Building on these ideas, this paper overcomes the invariance problem of previous attempts and provides a consistent, unified picture of optimization on arbitrary search spaces via invariance principles. More specifically, we consider an arbitrary search space X, either discrete or continuous, and a black-box optimization problem on X, that is, a function $f : X \to \mathbb{R}$ to be minimized. We assume that a family of probability distributions P_θ on X depending on a continuous multicomponent parameter $\theta \in \Theta$ has been chosen. A classical example is to take $X = \mathbb{R}^d$ and to consider the family of all Gaussian distributions P_θ on $\mathbb{R}^d$, with $\theta = (m, C)$ the mean and covariance matrix. Another simple example is $X = \{0,1\}^d$ equipped with the family of Bernoulli measures: $\theta = (\theta_i)_{1 \leqslant i \leqslant d}$ and $P_\theta(x) = \prod \theta_i^{x_i}(1-\theta_i)^{1-x_i}$ for $x = (x_i) \in X$.

From this setting, *information-geometric optimization* (IGO) can be defined in a natural way. At each (continuous) time t, we maintain a value θ^t of the parameter of the distribution. The function f to be optimized is transferred to the parameter space Θ by means of a suitable time-dependent transformation based on the P_{θ^t}*-levels* of f (Definition 3). The *IGO flow*, introduced in Definition 4, follows the natural gradient of the expected value of this function of θ^t in the parameter space Θ, where the natural gradient derives from the Fisher information metric. The IGO flow is thus the flow of an ordinary differential equation in space Θ. This continuous-time gradient flow is turned into a family of explicit *IGO algorithms* by taking an Euler time discretization of the differential equation and approximating the distribution P_{θ^t} by using samples. From the start, the IGO flow is invariant under strictly increasing transformations of f (Proposition 17); we also prove that the sampling procedure

is consistent (Theorem 6). IGO algorithms share their final algebraic form with the *natural evolution strategies* (NES) introduced in the Gaussian setting (Wierstra et al., 2008; Sun et al., 2009; Glasmachers et al., 2010; Wierstra et al., 2014); the latter are thus recovered in the IGO framework as an Euler approximation to a well-defined flow, without heuristic arguments.

The IGO method also has an equivalent description as an *infinitesimal maximum likelihood update* (Theorem 10); this reveals a new property of the natural gradient and does not require a smooth parametrization by θ anymore. This also establishes a specific link (Theorem 12) between IGO, the natural gradient, and the *cross-entropy method* (de Boer et al., 2005).

When we instantiate IGO using the family of Gaussian distributions on $\mathbb{R}^d$, we naturally obtain versions of the well-known *covariance matrix adaptation evolution strategy* (CMA-ES, Hansen and Ostermeier 2001; Hansen and Kern 2004; Jastrebski and Arnold 2006) and of *natural evolution strategies*. With Bernoulli measures on the discrete cube $\{0,1\}^d$, we recover (Proposition 14) the well-known *population-based incremental learning* (PBIL, Baluja and Caruana 1995; Baluja 1994) and the *compact genetic algorithm* (cGA, Harik et al. 1999); this derivation of PBIL or cGA as a natural gradient ascent appears to be new, and emphasizes the common ground between continuous and discrete optimization.

From the IGO framework, it is (theoretically) immediate to build new optimization algorithms using more complex families of distributions than Gaussian or Bernoulli. As an illustration, distributions associated with restricted Boltzmann machines (RBMs) provide a new but natural algorithm for discrete optimization on $\{0,1\}^d$ which is able to handle dependencies between the bits (see also Berny 2002). The probability distributions associated with RBMs are multimodal; combined with the specific information-theoretic properties of IGO that guarantee minimal change in diversity over time, this allows IGO to reach multiple optima at once in a natural way, at least in a simple experimental setup (Appendix B).

The IGO framework is built to achieve maximal *invariance properties*. First, the IGO flow is invariant under reparametrization of the family of distributions P_θ, that is, it only depends on P_θ and not on the way we write the parameter θ (Proposition 18). This invariance under θ-reparametrization is the main idea behind *information geometry* (Amari and Nagaoka, 2000). For instance, for Gaussian measures it should not matter whether we use the covariance matrix or its inverse or a Cholesky factor as the parameter. This limits the influence of encoding choices on the behavior of the algorithm. Second, the IGO flow is invariant under a change of coordinates in the search space X, provided that this change of coordinates globally preserves the family of distributions P_θ (Proposition 19). For instance, for Gaussian distributions on $\mathbb{R}^d$, this includes all affine changes of coordinates. This means that the algorithm, apart from initialization, does not depend on the precise way the data is presented. Last, the IGO flow and IGO algorithms are invariant under applying a strictly increasing function to f (Proposition 17). Contrary to previous formulations using natural gradients (Wierstra et al., 2008; Glasmachers et al., 2010; Akimoto et al., 2010), this invariance is achieved from the start. Such invariance properties mean that we deal with *intrinsic* properties of the objects themselves, and not with the way we encode them as collections of numbers in $\mathbb{R}^d$. It also means, most importantly, that we make a minimal number of arbitrary choices.

In Section 2, we define the IGO flow and the IGO algorithm. We begin with standard facts about the definition and basic properties of the natural gradient, and its connection with Kullback–Leibler divergence and diversity. We then proceed to the detailed description of the algorithm.

In Section 3, we state some first mathematical properties of IGO. These include monotone improvement of the objective function, invariance properties, and the form of IGO for exponential families of probability distributions.

In Section 4 we explain the theoretical relationships between IGO, maximum likelihood estimates and the cross-entropy method. In particular, IGO is uniquely characterized by a weighted log-likelihood maximization property (Theorem 10).

In Section 5, we apply the IGO framework to explicit families of probability distributions. Several well-known optimization algorithms are recovered this way. These include PBIL (Sec. 5.1) using Bernoulli distributions, and versions of CMA-ES and other evolutionary algorithms such as EMNA and xNES (Sec. 5.2) using Gaussian distributions.

Appendix A discusses further aspects, perspectives and implementation matters of the IGO framework. In Appendix B, we illustrate how IGO can be used to design new optimization algorithms, by deriving the IGO algorithm associated with restricted Boltzmann machines. We illustrate experimentally, on a simple bimodal example, the specific influence of the Fisher information matrix on the optimization trajectories, and in particular on the diversity of the optima obtained. Appendix C details a number of further mathematical properties of IGO (such as its invariance properties or the case of noisy objective functions). Appendix D contains the previously omitted proofs of the mathematical statements.

2. Algorithm Description

We now present the outline of the algorithm. Each step is described in more detail in the sections below.

The IGO flow can be seen as an *estimation of distribution algorithm*: at each time t, we maintain a probability distribution P_{θ^t} on the search space X, where $\theta^t \in \Theta$. The value of θ^t will evolve so that, over time, P_{θ^t} gives more weight to points x with better values of the function $f(x)$ to optimize.

A straightforward way to proceed is to transfer f from x-space to θ-space: define a function $F(\theta)$ as the P_θ-average of f and then do a gradient descent for $F(\theta)$ in space Θ (Berny, 2000a, 2002, 2000b; Gallagher and Frean, 2005). This way, θ will converge to a point such that P_θ yields a good average value of f. We depart from this approach in two ways:

- At each time, we replace f with an adaptive transformation of f representing how good or bad observed values of f are *relative to other observations*. This provides invariance under all monotone transformations of f.

- Instead of the vanilla gradient for θ, we use the so-called *natural gradient* given by the Fisher information matrix. This reflects the intrinsic geometry of the space of probability distributions, as introduced by Rao and Jeffreys (Rao, 1945; Jeffreys, 1946) and later elaborated upon by Amari and others (Amari and Nagaoka, 2000). This provides invariance under reparametrization of θ and, importantly, minimizes the change of diversity of P_θ.

The algorithm is constructed in two steps: we first give an "ideal" version, namely, a version in which time t is continuous so that the evolution of θ^t is given by an ordinary differential equation in Θ. Second, the actual algorithm is a time discretization using a finite time step and Monte Carlo sampling instead of exact P_θ-averages.

2.1 The Natural Gradient on Parameter Space

We recall suitable definitions of the vanilla and the natural gradient and motivate using the natural gradient in the context of optimization.

2.1.1 About Gradients and the Shortest Path Uphill

Let g be a smooth function from $\mathbb{R}^d$ to $\mathbb{R}$, to be maximized. We first recall the interpretation of gradient ascent as "the shortest path uphill".

Let $y \in \mathbb{R}^d$. Define the vector z by

$$z = \lim_{\varepsilon \to 0} \arg\max_{z,\, \|z\| \leqslant 1} g(y + \varepsilon z) \ . \tag{1}$$

Then one can check that z is the normalized gradient of g at y: $z_i = \frac{\partial g/\partial y_i}{\|\partial g/\partial y\|}$. (This holds only at points y where the gradient of g does not vanish.)

This shows that, for small δt, the well-known gradient ascent of g given by

$$y_i^{t+\delta t} = y_i^t + \delta t \, \frac{\partial g}{\partial y_i}$$

realizes the largest increase of the value of g, for a given step size $\|y^{t+\delta t} - y^t\|$.

The relation (1) depends on the choice of a norm $\|\cdot\|$ (the gradient of g is given by $\partial g/\partial y_i$ only in an orthonormal basis). If we use, instead of the standard metric $\|y - y'\| = \sqrt{\sum (y_i - y'_i)^2}$ on $\mathbb{R}^d$, a metric $\|y - y'\|_A = \sqrt{\sum A_{ij}(y_i - y'_i)(y_j - y'_j)}$ defined by a positive definite matrix A_{ij}, then the gradient of g with respect to this metric is given by $\sum_j A^{-1}_{ij} \frac{\partial g}{\partial y_i}$. This follows from the textbook definition of gradients by $g(y+\varepsilon z) = g(y) + \varepsilon \langle \nabla g, z \rangle_A + O(\varepsilon^2)$ with $\langle \cdot, \cdot \rangle_A$ the scalar product associated with the matrix A_{ij} (Schwartz, 1992).

It is possible to write the analogue of (1) using the A-norm. We then find that the gradient ascent associated with metric A is given by

$$y^{t+\delta t} = y^t + \delta t \, A^{-1} \, \frac{\partial g}{\partial y}$$

for small δt and maximizes the increment of g for a given A-distance $\|y^{t+\delta t} - y^t\|_A$—it realizes the steepest A-ascent. Maybe this viewpoint clarifies the relationship between gradient and metric: this steepest ascent property can actually be used as a definition of gradients.

In our setting we want to use a gradient ascent in the parameter space Θ of our distributions P_θ. The "vanilla" gradient $\frac{\partial}{\partial \theta_i}$ is associated with the metric $\|\theta - \theta'\| = \sqrt{\sum (\theta_i - \theta'_i)^2}$ and clearly depends on the choice of parametrization θ. Thus this metric, and the direction pointed by this gradient, are not intrinsic, in the sense that they do not depend only on the *distribution* P_θ. A metric depending on θ only through the distributions P_θ can be defined as follows.

2.1.2 Fisher Information and the Natural Gradient on Parameter Space

Let $\theta, \theta' \in \Theta$ be two values of the distribution parameter. A widely used way to define a "distance" between two generic distributions[1] P_θ and $P_{\theta'}$ is the *Kullback–Leibler divergence* from information theory, defined (Kullback, 1997) as

$$\mathrm{KL}(P_{\theta'} \,\|\, P_\theta) = \int_x \ln \frac{P_{\theta'}(\mathrm{d}x)}{P_\theta(\mathrm{d}x)}\, P_{\theta'}(\mathrm{d}x) \ .$$

When $\theta' = \theta + \delta\theta$ is close to θ, under mild smoothness assumptions we can expand the Kullback–Leibler divergence at second order in $\delta\theta$. This expansion defines the Fisher information matrix I at θ (Kullback, 1997):

$$\mathrm{KL}(P_{\theta+\delta\theta} \,\|\, P_\theta) = \frac{1}{2} \sum I_{ij}(\theta)\, \delta\theta_i \delta\theta_j + O(\delta\theta^3) \ . \tag{2}$$

An equivalent definition of the Fisher information matrix is by the usual formulas (Cover and Thomas, 2006)

$$I_{ij}(\theta) = \int_x \frac{\partial \ln P_\theta(x)}{\partial\theta_i} \frac{\partial \ln P_\theta(x)}{\partial\theta_j}\, P_\theta(\mathrm{d}x) = -\int_x \frac{\partial^2 \ln P_\theta(x)}{\partial\theta_i\, \partial\theta_j}\, P_\theta(\mathrm{d}x) \ .$$

The Fisher information matrix defines a (Riemannian) metric on Θ: the distance, in this metric, between two very close values of θ is given by the square root of twice the Kullback–Leibler divergence. Since the Kullback–Leibler divergence depends only on P_θ and not on the parametrization of θ, this metric is intrinsic.

If $g : \Theta \to \mathbb{R}$ is a smooth function on the parameter space, its *natural gradient* (Amari, 1998) at θ is defined in accordance with the Fisher metric as

$$(\widetilde{\nabla}_\theta\, g)_i = \sum_j I^{-1}_{ij}(\theta)\, \frac{\partial g(\theta)}{\partial\theta_j}$$

or more synthetically

$$\widetilde{\nabla}_\theta\, g = I^{-1}\, \frac{\partial g}{\partial\theta} \ .$$

From now on, we will use $\widetilde{\nabla}_\theta$ to denote the natural gradient and $\frac{\partial}{\partial\theta}$ to denote the vanilla gradient.

By construction, the natural gradient descent is intrinsic: it does not depend on the chosen parametrization θ of P_θ, so that it makes sense to speak of the natural gradient ascent of a function $g(P_\theta)$. The Fisher metric is essentially the only way to obtain this property (Amari and Nagaoka, 2000, Section 2.4).

Given that the Fisher metric comes from the Kullback–Leibler divergence, the "shortest path uphill" property of gradients mentioned above translates as follows (see also Amari 1998, Theorem 1):

1. Throughout the text we do not distinguish a probability distribution P, seen as a measure, and its density with respect to some unspecified reference measure $\mathrm{d}x$, and so will write indifferently $P(\mathrm{d}x)$ or $P(x)\mathrm{d}x$. The measure-theoretic viewpoint allows for a unified treatment of the discrete and continuous case.

Proposition 1 *The natural gradient ascent points in the direction $\delta\theta$ achieving the largest change of the objective function, for a given distance between P_θ and $P_{\theta+\delta\theta}$ in Kullback–Leibler divergence. More precisely, let g be a smooth function on the parameter space Θ. Let $\theta \in \Theta$ be a point where $\widetilde{\nabla} g(\theta)$ does not vanish. Then, if*

$$\delta\theta = \frac{\widetilde{\nabla} g(\theta)}{\|\widetilde{\nabla} g(\theta)\|}$$

is the direction of the natural gradient of g (with $\|\cdot\|$ the Fisher norm), we have

$$\delta\theta = \lim_{\varepsilon\to 0} \frac{1}{\varepsilon} \operatorname*{arg\,max}_{\substack{\delta\theta' \text{ such that} \\ \mathrm{KL}(P_{\theta+\delta\theta'} \,\|\, P_\theta) \leqslant \varepsilon^2/2}} g(\theta + \delta\theta').$$

Here we have implicitly assumed that the parameter space Θ is such that no two points $\theta \in \Theta$ define the same probability distribution, and the mapping $P_\theta \mapsto \theta$ is smooth.

2.1.3 Why Use the Fisher Metric Gradient for Optimization? Relationship to Diversity

The first reason for using the natural gradient is its reparametrization invariance, which makes it the only gradient available in a general abstract setting (Amari and Nagaoka, 2000). Practically, this invariance also limits the influence of encoding choices on the behavior of the algorithm (Appendix C.1). The Fisher matrix can be also seen as an *adaptive learning rate* for different components of the parameter vector θ_i: components i with a high impact on P_θ will be updated more cautiously.

Another advantage comes from the relationship with Kullback–Leibler distance in view of the "shortest path uphill" (see also Amari 1998). To minimize the value of some function $g(\theta)$ defined on the parameter space Θ, the naive approach follows a gradient descent for g using the "vanilla" gradient

$$\theta_i^{t+\delta t} = \theta_i^t + \delta t \tfrac{\partial g}{\partial\theta_i}$$

and, as explained above, this maximizes the increment of g for a given increment $\|\theta^{t+\delta t} - \theta^t\|$. On the other hand, the Fisher gradient

$$\theta_i^{t+\delta t} = \theta_i^t + \delta t I^{-1} \tfrac{\partial g}{\partial\theta_i}$$

maximizes the increment of g for a given Kullback–Leibler distance $\mathrm{KL}(P_{\theta^{t+\delta t}} \,\|\, P_{\theta^t})$.

In particular, if we choose an initial value θ^0 such that P_{θ^0} covers the whole space X uniformly (or a wide portion, in case X is unbounded), the Kullback–Leibler divergence between P_{θ^t} and P_{θ^0} is the Shannon entropy of the uniform distribution minus the Shannon entropy of P_{θ^t}, and so this divergence measures the loss of diversity of P_{θ^t} with respect to the uniform distribution.

Proposition 2 *Let $g : \Theta \to \mathbb{R}$ be a regular function of θ and let θ^0 such that P_{θ^0} is the uniform distribution on a finite space X. Let $(\theta^t)_{t\geqslant 0}$ be the trajectory of the gradient ascent of g using the natural gradient. Then for small t we have*

$$\theta^t = \arg\max_\theta \{t \cdot g(\theta) + \mathrm{Ent}(P_\theta)\} + o(t) \tag{3}$$

where Ent *is the Shannon entropy.*

(We have stated the proposition over a finite space to have a well-defined uniform distribution. A short proof, together with the regularity conditions on g, is given in Appendix D.)

So following the natural gradient of a function g, starting at or close to the uniform distribution, amounts to optimizing the function g with *minimal loss of diversity, provided the initial diversity is large.* (This is valid, of course, only at the beginning; once one gets too far from uniform, a better interpretation is minimal *change* of diversity.) On the other hand, the vanilla gradient descent does not satisfy Proposition 2: it optimizes g with minimal change in the numerical values of the parameter θ, which is of little interest.

So arguably this method realizes the best trade-off between optimization and loss of diversity. (Though, as can be seen from the detailed algorithm description below, maximization of diversity occurs only greedily at each step, and so there is no guarantee that after a given time, IGO will provide the highest possible diversity for a given objective function value.)

An experimental confirmation of the positive influence of the Fisher matrix on diversity is given in Appendix B below.

2.2 IGO: Information-Geometric Optimization

We now introduce a quantile-based rewriting of the objective function. From applying the natural gradient on the rewritten objective we derive *information-geometric optimization*.

2.2.1 Quantile Rewriting of f

Our original problem is to minimize a function $f : X \to \mathbb{R}$. A simple way to turn f into a function on Θ is to use the expected value $-\mathbb{E}_{P_\theta} f$ (Berny, 2000a; Wierstra et al., 2008), but expected values can be unduly influenced by extreme values and using them can be rather unstable (Whitley, 1989); moreover $-\mathbb{E}_{P_\theta} f$ is not invariant under increasing transformation of f (this invariance implies we can only *compare* f-values, not sum them up).

Instead, we take an adaptive, quantile-based approach by first replacing the function f with a monotone rewriting $W^f_{\theta^t}$, depending on the current parameter value θ^t, and then following the gradient of $\mathbb{E}_{P_\theta} W^f_{\theta^t}$, seen as a function of θ. A due choice of $W^f_{\theta^t}$ allows to control the range of the resulting values and achieves the desired invariance. Because the rewriting $W^f_{\theta^t}$ depends on θ^t, it might be viewed as an *adaptive* f-transformation.

The monotone rewriting entails that if $f(x)$ is "small" then $W^f_\theta(x) \in \mathbb{R}$ is "large" and vice versa. The quantitative meaning of "small" or "large" depends on $\theta \in \Theta$. To obtain the value of $W^f_\theta(x)$ we compare $f(x)$ to the quantiles of f under the current distribution, as measured by the P_θ-level fraction in which the value of $f(x)$ lies.

Definition 3 *The lower and upper P_θ-f-levels of $x \in X$ are defined as*

$$\begin{aligned} q^<_\theta(x) &= \Pr_{x' \sim P_\theta}(f(x') < f(x)) \\ q^\leqslant_\theta(x) &= \Pr_{x' \sim P_\theta}(f(x') \leqslant f(x)) \ . \end{aligned} \tag{4}$$

Let $w : [0;1] \to \mathbb{R}$ be a non-increasing function, the selection scheme.

The transform $W^f_\theta(x)$ of an objective function $f : X \to \mathbb{R}$ is defined as a function of the P_θ-f-level of x as

$$W^f_\theta(x) = \begin{cases} w(q^\leqslant_\theta(x)) & \text{if } q^\leqslant_\theta(x) = q^<_\theta(x), \\ \frac{1}{q^\leqslant_\theta(x) - q^<_\theta(x)} \int_{q=q^<_\theta(x)}^{q=q^\leqslant_\theta(x)} w(q)\,\mathrm{d}q & \text{otherwise.} \end{cases} \tag{5}$$

The level functions $q : X \to [0,1]$ reflect the probability to sample a better value than $f(x)$. They are monotone in f (if $f(x_1) \leqslant f(x_2)$ then $q^<_\theta(x_1) \leqslant q^<_\theta(x_2)$, and likewise for $q^\leqslant$) and invariant under strictly increasing transformations of f.

A typical choice for w is $w(q) = \mathbb{1}_{q \leqslant q_0}$ for some fixed value q_0, the *selection quantile*. In what follows, we suppose that a selection scheme (weighting scheme) w has been chosen once and for all.

As desired, the definition of W^f_θ is invariant under a strictly increasing transformation of f. For instance, the P_θ-median of f gets remapped to $w(\frac{1}{2})$.

Note that $\mathbb{E}_{x \sim P_\theta} W^f_\theta(x)$ is always equal to $\int_0^1 w$, independently of f and θ: indeed, by definition, the P_θ inverse-quantile of a random point under P_θ is uniformly distributed in $[0;1]$. In the following, our objective will be to maximize the expected value of $W^f_{\theta^t}$ over θ, that is, to maximize

$$\mathbb{E}_{P_\theta} W^f_{\theta^t} = \int W^f_{\theta^t}(x)\, P_\theta(\mathrm{d}x) \tag{6}$$

over θ, where θ^t is fixed at a given step but will adapt over time.

Importantly, $W^f_\theta(x)$ can be estimated in practice: indeed, the P_θ-f-levels $\Pr_{x' \sim P_\theta}(f(x') < f(x))$ can be estimated by taking samples of P_θ and ordering the samples according to the value of f (see below). The estimate remains invariant under strictly increasing f-transformations.

2.2.2 The IGO Gradient Flow

At the most abstract level, IGO is a continuous-time gradient flow in the parameter space Θ, which we define now. In practice, discrete time steps (a.k.a. iterations) are used, and P_θ-integrals are approximated through sampling, as described in the next section.

Let θ^t be the current value of the parameter at time t, and let $\delta t \ll 1$. We define $\theta^{t+\delta t}$ in such a way as to increase the P_θ-weight of points where f is small, while not going too far from P_{θ^t} in Kullback–Leibler divergence. We use the adaptive weights $W^f_{\theta^t}$ as a way to measure which points have large or small values. In accordance with (6), this suggests taking the gradient ascent

$$\theta^{t+\delta t} = \theta^t + \delta t\, \widetilde{\nabla}_\theta \int W^f_{\theta^t}(x)\, P_\theta(\mathrm{d}x) \tag{7}$$

where the natural gradient is suggested by Proposition 1.

Note again that we use $W^f_{\theta^t}$ and not W^f_θ in the integral. So the gradient $\widetilde{\nabla}_\theta$ does not act on the adaptive objective $W^f_{\theta^t}$. If we used W^f_θ instead, we would face a paradox: right after a move, previously good points do not seem so good any more since the distribution has improved. More precisely, $\int W^f_\theta(x)\, P_\theta(\mathrm{d}x)$ is constant and always equal to the average weight $\int_0^1 w$, and so the gradient would always vanish.

Using the log-likelihood trick $\widetilde{\nabla} P_\theta = P_\theta \, \widetilde{\nabla} \ln P_\theta$ (assuming P_θ is smooth), we get an equivalent expression of the update above as an integral under the current distribution P_{θ^t}; this is important for practical implementation. This leads to the following definition.

Definition 4 (IGO flow) *The IGO flow is the set of continuous-time trajectories in space Θ, defined by the ordinary differential equation*

$$\frac{\mathrm{d}\theta^t}{\mathrm{d}t} = \widetilde{\nabla}_\theta \int W^f_{\theta^t}(x)\, P_\theta(x) \mathrm{d}x \tag{8}$$

$$= \int W^f_{\theta^t}(x)\, \frac{\widetilde{\nabla}_\theta P_\theta(x)}{P_{\theta^t}(x)} P_{\theta^t}(x) \mathrm{d}x$$

$$= \int W^f_{\theta^t}(x)\, \widetilde{\nabla}_\theta \ln P_\theta(x)\, P_{\theta^t}(\mathrm{d}x) \tag{9}$$

$$= I^{-1}(\theta^t) \int W^f_{\theta^t}(x)\, \frac{\partial \ln P_\theta(x)}{\partial \theta}\, P_{\theta^t}(\mathrm{d}x) \ . \tag{10}$$

where the gradients are taken at point $\theta = \theta^t$, and I is the Fisher information matrix.

Natural evolution strategies (NES, Wierstra et al. 2008; Glasmachers et al. 2010; Sun et al. 2009; Wierstra et al. 2014) feature a related gradient *descent* with $f(x)$ instead of $W^f_{\theta^t}(x)$. The associated flow would read

$$\frac{\mathrm{d}\theta^t}{\mathrm{d}t} = -\widetilde{\nabla}_\theta \int f(x)\, P_\theta(\mathrm{d}x) \ , \tag{11}$$

where the gradient is taken at θ^t (in the sequel when not explicitly stated, gradients in θ are taken at $\theta = \theta^t$). However, in the end NESs always implement algorithms using sample quantiles (via "nonlinear *fitness shaping*"), as if derived from the gradient ascent of $W^f_{\theta^t}(x)$.

The update (9) is a weighted average of "intrinsic moves" increasing the log-likelihood of some points. We can slightly rearrange the update as

$$\frac{\mathrm{d}\theta^t}{\mathrm{d}t} = \int \overbrace{W^f_{\theta^t}(x)}^{\text{preference weight}} \underbrace{\widetilde{\nabla}_\theta \ln P_\theta(x)}_{\text{intrinsic move to reinforce } x} \overbrace{P_{\theta^t}(\mathrm{d}x)}^{\text{current sample distribution}} \tag{12}$$

$$= \widetilde{\nabla}_\theta \int \underbrace{W^f_{\theta^t}(x) \ln P_\theta(x)}_{\text{weighted log-likelihood}} P_{\theta^t}(\mathrm{d}x) \ , \tag{13}$$

which provides an interpretation for the IGO gradient flow as a gradient ascent optimization of the weighted log-likelihood of the "good points" of the current distribution. In the sense of Theorem 10 below, IGO is in fact the "best" way to increase this log-likelihood.

For exponential families of probability distributions, we will see later that the IGO flow rewrites as a nice derivative-free expression (21).

2.2.3 IGO Algorithms: Time Discretization and Sampling

The above is a mathematically well-defined continuous-time flow in parameter space. Its practical implementation involves three approximations depending on two parameters N and δt:

- the integral under P_{θ^t} is approximated using N samples taken from P_{θ^t};
- the value $W^f_{\theta^t}$ is approximated for each sample taken from P_{θ^t};
- the time derivative $\frac{\mathrm{d}\theta^t}{\mathrm{d}t}$ is approximated by a δt time increment: instead of the continuous-time IGO flow (8) we use its Euler approximation scheme $\theta^{t+\delta t} \approx \theta^t + \delta t \frac{\mathrm{d}\theta^t}{\mathrm{d}t}$, so that the time t of the flow is discretized with a step size δt, which thus becomes the learning rate of the algorithm. (See Corollary 21 for an interpretation of δt as a number of bits of information introduced in the distribution P_{θ^t} at each step.)

We also assume that the Fisher information matrix $I(\theta)$ and $\frac{\partial \ln P_\theta(x)}{\partial \theta}$ can be computed (see discussion below if $I(\theta)$ is unknown).

At each step, we draw N samples $x_1, \dots, x_N$ under P_{θ^t}. To approximate $W^f_{\theta^t}$, we rank the samples according to the value of f. Define $\mathrm{rk}(x_i) = \#\{j \,|\, f(x_j) < f(x_i)\}$ and let the estimated weight of sample x_i be

$$\widehat{w}_i = \frac{1}{N}\, w\left(\frac{\mathrm{rk}(x_i) + 1/2}{N}\right), \tag{14}$$

using the non-increasing selection scheme function w introduced in Definition 3 above. (This is assuming there are no ties in our sample; in case several sample points have the same value of f, we define $\widehat{w}_i$ by averaging the above over all possible rankings of the ties[2].)

Then we can approximate the IGO flow as follows.

Definition 5 (IGO algorithms) *The* IGO algorithm *associated with parametrization θ, sample size N and step size δt is the following update rule for the parameter θ^t. At each step, N sample points $x_1, \dots, x_N$ are drawn according to the distribution P_{θ^t}. The parameter is updated according to*

$$\theta^{t+\delta t} = \theta^t + \delta t \sum_{i=1}^N \widehat{w}_i \ \widetilde{\nabla}_\theta \ln P_\theta(x_i)\Big|_{\theta=\theta^t} \tag{16}$$

$$= \theta^t + \delta t\, I^{-1}(\theta^t) \sum_{i=1}^N \widehat{w}_i \ \frac{\partial \ln P_\theta(x_i)}{\partial \theta}\Big|_{\theta=\theta^t} \tag{17}$$

where $\widehat{w}_i$ is the weight (14) *obtained from the ranked values of the objective function f.*

Equivalently one can fix the weights $w_i = \frac{1}{N}\, w\left(\frac{i-1/2}{N}\right)$ once and for all and rewrite the update as

$$\theta^{t+\delta t} = \theta^t + \delta t\, I^{-1}(\theta^t) \sum_{i=1}^N w_i \ \frac{\partial \ln P_\theta(x_{i:N})}{\partial \theta}\Big|_{\theta=\theta^t} \tag{18}$$

2. A mathematically neater but less intuitive version would be

$$\widehat{w}_i = \frac{1}{\mathrm{rk}^{\leqslant}(x_i) - \mathrm{rk}^{<}(x_i)} \int_{u=\mathrm{rk}^{<}(x_i)/N}^{u=\mathrm{rk}^{\leqslant}(x_i)/N} w(u)\mathrm{d}u \tag{15}$$

with $\mathrm{rk}^{<}(x_i) = \#\{j \,|\, f(x_j) < f(x_i)\}$ and $\mathrm{rk}^{\leqslant}(x_i) = \#\{j \,|\, f(x_j) \leqslant f(x_i)\}$.

where $x_{i:N}$ denotes the sampled point ranked i^{th} according to f, i.e. $f(x_{1:N}) < \ldots < f(x_{N:N})$ (assuming again there are no ties). Note that $\{x_{i:N}\} = \{x_i\}$ and $\{w_i\} = \{\widehat{w}_i\}$.

As will be discussed in Section 5, this update applied to multivariate normal distributions or Bernoulli measures allows us to neatly recover versions of some well-established algorithms, in particular CMA-ES and PBIL. Actually, in the Gaussian context updates of the form (17) have already been introduced (Glasmachers et al., 2010; Akimoto et al., 2010), though not formally derived from a continuous-time flow with inverse quantiles.

When $N \to \infty$, the IGO algorithm using samples approximates the continuous-time IGO gradient flow, see Theorem 6 below. Indeed, the IGO algorithm, with $N = \infty$, is simply the Euler approximation scheme for the ordinary differential equation defining the IGO flow (8). The latter result thus provides a sound mathematical basis for currently used rank-based updates.

2.2.4 IGO Flow versus IGO Algorithms

The IGO *flow* (8) is a well-defined continuous-time set of trajectories in the space of probability distributions P_θ, depending only on the objective function f and the chosen family of distributions. It does not depend on the chosen parametrization for θ (Proposition 18).

On the other hand, there are several IGO *algorithms* associated with this flow. Each IGO algorithm approximates the IGO flow in a slightly different way. An IGO algorithm depends on three further choices: a sample size N, a time discretization step size δt, and a choice of parametrization for θ in which to implement (17).

If δt is small enough, and N large enough, the influence of the parametrization θ disappears and all IGO algorithms are approximations of the "ideal" IGO flow trajectory. However, the larger δt, the poorer the approximation gets.

So for large δt, different IGO algorithms for the same IGO flow may exhibit different behaviors. We will see an instance of this phenomenon for Gaussian distributions: both CMA-ES and the maximum likelihood update (EMNA) can be seen as IGO algorithms, but the latter with $\delta t = 1$ is known to exhibit premature loss of diversity (Section 5.2).

Still, if δt is sufficiently small, two IGO algorithms for the same IGO flow will differ less from each other than from a non-IGO algorithm: at each step the difference is only $O(\delta t^2)$ (Appendix C.1). On the other hand, for instance, the difference between an IGO algorithm and the vanilla gradient ascent is, generally, not smaller than $O(\delta t)$ at each step, i.e., it can be roughly as big as the steps themselves.

2.2.5 Unknown Fisher Matrix

The algorithm presented so far assumes that the Fisher matrix $I(\theta)$ is known as a function of θ. This is the case for Gaussian or Bernoulli distributions. However, for restricted Boltzmann machines as considered below, no analytical form is known. Yet, provided the quantity $\frac{\partial}{\partial\theta} \ln P_\theta(x)$ can be computed or approximated, it is possible to approximate the integral

$$I_{ij}(\theta) = \int_x \frac{\partial \ln P_\theta(x)}{\partial \theta_i} \frac{\partial \ln P_\theta(x)}{\partial \theta_j} \, P_\theta(\mathrm{d}x)$$

using P_θ-Monte Carlo samples for x. These samples may or may not be the same as those used in the IGO update (17): in particular, it is possible to use as many Monte Carlo samples as necessary to approximate I_{ij}, at no additional cost in terms of the number of calls to the black-box function f to optimize.

Note that each Monte Carlo sample x will contribute $\frac{\partial \ln P_\theta(x)}{\partial \theta_i} \frac{\partial \ln P_\theta(x)}{\partial \theta_j}$ to the Fisher matrix approximation. This is a rank-1 non-negative matrix[3]. So, for the approximated Fisher matrix to be invertible, the number of (distinct) samples x needs to be at least equal to, and ideally much larger than, the number of components of the parameter θ: $N_{\text{Fisher}} \geqslant \dim \Theta$.

For exponential families of distributions, the IGO update has a particular form (21) which simplifies this matter somewhat (Section 3.3). In Appendix B this is examplified using restricted Boltzmann machines.

3. First Properties of IGO

In this section we derive some basic properties of IGO and present the IGO flow for exponential families.

3.1 Consistency of Sampling

The first property to check is that when $N \to \infty$, the update rule using N samples converges to the IGO update rule. This is *not* a straightforward application of the law of large numbers, because the estimated weights $\widehat{w}_i$ depend (non-continuously) on the *whole* sample $x_1, \ldots, x_N$, and not only on x_i.

Theorem 6 (Consistency) *When $N \to \infty$, the N-sample IGO update rule* (17)*:*

$$\theta^{t+\delta t} = \theta^t + \delta t \, I^{-1}(\theta^t) \sum_{i=1}^{N} \widehat{w}_i \left. \frac{\partial \ln P_\theta(x_i)}{\partial \theta} \right|_{\theta=\theta^t}$$

converges with probability 1 *to the update rule* (7)*:*

$$\theta^{t+\delta t} = \theta^t + \delta t \, \widetilde{\nabla}_\theta \int W_{\theta^t}^f(x) \, P_\theta(\mathrm{d}x) \ .$$

The proof is given in Appendix D, under mild regularity assumptions. In particular we do not require that w is continuous. Unfortunately, the proof does not provide an explicit sample size above which the IGO algorithm would be guaranteed to stay close to the IGO flow with high probability; presumably such a size would be larger than the typical sample sizes used in practice.

This theorem may clarify previous claims (Wierstra et al., 2008; Akimoto et al., 2010) where rank-based updates similar to (7), such as in NES or CMA-ES, were derived from optimizing the expected value $-\mathbb{E}_{P_\theta} f$. The rank-based weights $\widehat{w}_i$ were then introduced *after* the derivation as a useful heuristic to improve stability. Theorem 6 shows that, for large N, CMA-ES and NES actually follow the gradient flow of the quantity $\mathbb{E}_{P_\theta} W_{\theta^t}^f$: the update can be rigorously derived from optimizing the expected value of the inverse-quantile-rewriting $W_{\theta^t}^f$.

3. The alternative, equivalent formula $I_{ij}(\theta) = -\int_x \frac{\partial^2 \ln P_\theta(x)}{\partial \theta_i \, \partial \theta_j} \, P_\theta(\mathrm{d}x)$ for the Fisher matrix will not necessarily yield non-negative matrices through Monte Carlo sampling.

3.2 Monotonicity: Quantile Improvement

Gradient descents come with a guarantee that the fitness value decreases over time. Here, since we work with probability distributions on X, we need to define a "fitness" of the distribution P_{θ^t}. An obvious choice is the expectation $\mathbb{E}_{P_{\theta^t}} f$, but it is not invariant under f-transformation and moreover may be sensitive to extreme values.

It turns out that the monotonicity properties of the IGO gradient flow depend on the choice of the selection scheme w. For instance, if $w(u) = \mathbb{1}_{u \leqslant 1/2}$, then the median of f under P_{θ^t} improves over time.

Proposition 7 (Quantile improvement) *Consider the IGO flow* (8)*, with the weight* $w(u) = \mathbb{1}_{u \leqslant q}$ *where* $0 < q < 1$ *is fixed. Then the value of the* q*-quantile of* f *improves over time: if* $t_1 \leqslant t_2$ *then* $Q^q_{P_{\theta^{t_2}}}(f) \leqslant Q^q_{P_{\theta^{t_1}}}(f)$*. Here the* q*-quantile value* $Q^q_P(f)$ *of* f *under a probability distribution* P *is defined as the largest number* m *such that* $\Pr_{x\sim P}(f(x) \geqslant m) \geqslant 1 - q$*.*

Assume moreover that the objective function f *has no plateau, i.e. for any* $v \in \mathbb{R}$ *and any* $\theta \in \Theta$ *we have* $\Pr_{x\sim P_\theta}(f(x) = v) = 0$*. Then for* $t_1 < t_2$ *either* $\theta^{t_1} = \theta^{t_2}$ *or* $Q^q_{P_{\theta^{t_2}}}(f) < Q^q_{P_{\theta^{t_1}}}(f)$*.*

The proof is given in Appendix D, together with the necessary regularity assumptions. Note that on a discrete search space, the objective function has only plateaus, and the q-quantile will evolve by successive jumps even as θ evolves continuously.

This property is proved here only for the IGO gradient flow (8) with $N = \infty$ and $\delta t \to 0$. For an IGO algorithm with finite N, the dynamics is random and one cannot expect monotonicity. Still, Theorem 6 ensures that, with high probability, trajectories of a large enough finite population stay close to the infinite-population limit trajectory.

In Akimoto and Ollivier (2013) this result was extended to finite time steps instead of infinitesimal δt, using the IGO-ML framework from Section 4 below.

3.3 The IGO Flow for Exponential Families

The expressions for the IGO update simplify somewhat if the family P_θ happens to be an exponential family of probability distributions (see also Malagò et al. 2008, 2011 for optimization using the natural gradient for exponential families). This covers, for instance, Gaussian or Bernoulli distributions.

Suppose that P_θ can be written as

$$P_\theta(\mathrm{d}x) = \frac{1}{Z(\theta)} \exp\left(\sum \theta_i T_i(x)\right) H(\mathrm{d}x)$$

where $T_1, \dots, T_k$ is a finite family of functions on X, $H(\mathrm{d}x)$ is an arbitrary reference measure on X, and $Z(\theta)$ is the normalization constant. It is well-known (Amari and Nagaoka, 2000, (2.33)) that

$$\frac{\partial \ln P_\theta(x)}{\partial \theta_i} = T_i(x) - \mathbb{E}_{P_\theta} T_i \tag{19}$$

so that (Amari and Nagaoka, 2000, (3.59))

$$I_{ij}(\theta) = \mathrm{Cov}_{P_\theta}(T_i, T_j) \ . \tag{20}$$

By plugging this into the definition of the IGO flow (10) we find:

Proposition 8 *Let* P_θ *be an exponential family parametrized by the natural parameters* θ *as above. Then the IGO flow is given by*

$$\frac{\mathrm{d}\theta}{\mathrm{d}t} = \mathrm{Cov}_{P_\theta}(T, T)^{-1} \, \mathrm{Cov}_{P_\theta}(T, W^f_\theta) \tag{21}$$

where $\mathrm{Cov}_{P_\theta}(T, W^f_\theta)$ *denotes the vector* $(\mathrm{Cov}_{P_\theta}(T_i, W^f_\theta))_i$*, and* $\mathrm{Cov}_{P_\theta}(T, T)$ *the matrix* $(\mathrm{Cov}_{P_\theta}(T_i, T_j))_{ij}$*.*

Note that the right-hand side does not involve derivatives w.r.t. θ any more. This result makes it easy to simulate the IGO flow using, e.g., a Gibbs sampler for P_θ: both covariances in (21) may be approximated by sampling, so that neither the Fisher matrix nor the gradient term need to be known in advance, and no derivatives are involved.

The values of the variables $\bar{T}_i = \mathbb{E}T_i$, namely the expected value of T_i under the current distribution, can often be used as an alternative parametrization for an exponential family (e.g. for a one-dimensional Gaussian, these are the mean μ and the second moment $\mu^2 + \sigma^2$). The IGO flow (9) may be rewritten using these variables, using the relation $\widetilde{\nabla}_{\theta_i} = \frac{\partial}{\partial \bar{T}_i}$ for the natural gradient of exponential families (Proposition 29 in Appendix D). One finds:

Proposition 9 *With the same setting as in Proposition 8, the expectation variables* $\bar{T}_i = \mathbb{E}_{P_\theta} T_i$ *satisfy the following evolution equation under the IGO flow*

$$\frac{\mathrm{d}\bar{T}_i}{\mathrm{d}t} = \mathrm{Cov}(T_i, W^f_\theta) = \mathbb{E}(T_i \, W^f_\theta) - \bar{T}_i \, \mathbb{E}W^f_\theta \ . \tag{22}$$

The proof is given in Appendix D, in the proof of Theorem 12. We shall further exploit this fact in Section 4.

3.3.1 Exponential Families with Latent Variables.

Similar formulas hold when the distribution $P_\theta(x)$ is the marginal of an exponential distribution $P_\theta(x, h)$ over a "hidden" or "latent" variable h, such as the restricted Boltzmann machines of Appendix B.

Namely, with $P_\theta(x) = \frac{1}{Z(\theta)} \sum_h \exp(\sum_i \theta_i T_i(x, h)) \, H(\mathrm{d}x, \mathrm{d}h)$ we have

$$\frac{\partial \ln P_\theta(x)}{\partial \theta_i} = U_i(x) - \mathbb{E}_{P_\theta} U_i \tag{23}$$

where

$$U_i(x) = \mathbb{E}_{P_\theta}(T_i(x, h)|x)$$

is the expectation of $T_i(x, h)$ knowing x. Then the Fisher matrix is

$$I_{ij}(\theta) = \mathrm{Cov}_{P_\theta}(U_i, U_j) \tag{24}$$

and consequently, the IGO flow takes the form

$$\frac{\mathrm{d}\theta}{\mathrm{d}t} = \mathrm{Cov}_{P_\theta}(U, U)^{-1} \, \mathrm{Cov}_{P_\theta}(U, W^f_\theta) \ . \tag{25}$$

3.4 Further Mathematical Properties of IGO

IGO enjoys a number of other mathematical properties that are expanded upon in Appendix C.

- By its very construction, the IGO flow is invariant under a number of transformations of the original problem. First, replacing the objective function f with a strictly increasing function of f does not change the IGO flow (Proposition 17 in Appendix C.1).

  Second, changing the parameterization θ used for the family P_θ (e.g., letting θ be a variance or a standard deviation) results in unchanged trajectories for the distributions P_θ under the IGO flow (Proposition 18 in Appendix C.1).

  Finally, the IGO flow is insensitive to transformations of the original problem by a change of variable in the search space X itself, provided this transformation can be reflected in the family of distributions P_θ (Proposition 19 in Appendix C.1); this covers, for instance, optimizing $f(Ax)$ instead of $f(x)$ in $\mathbb{R}^d$ using Gaussian distributions, where A is any invertible matrix.

  These latter two invariances are specifically due to the natural gradient and are not satisfied by a vanilla gradient descent. f-invariance and X-invariance are directly inherited from the IGO flow by IGO algorithms, but this is only approximately true of θ-invariance, as discussed in Appendix C.1.

- The speed of the IGO flow is bounded by the variance of the weight function w on $[0; 1]$ (Appendix C.2, Proposition 20). This implies that the parameter δt in the IGO flow is not a meaningless variable but is related to the maximal number of bits introduced in P_θ at each step (Appendix C.2, Corollary 21).

- IGO algorithms still make sense when the objective function f is noisy, that is, when each call to f returns a non-deterministic value. This can be accounted for without changing the framework: we prove in Proposition 22 (Appendix C.3) that IGO for noisy f is equivalent to IGO for a non-noisy $\tilde{f}$ defined on a larger space $X \times \Omega$ with distributions $\tilde{P}_\theta$ that are uniform over Ω. Consequently, theorems such as consistency of sampling immediately transfer to the noisy case.

- The IGO flow can be computed explicitly in the simple case of linear functions on $\mathbb{R}^d$ using Gaussian distributions, and its convergence can be proven for linear functions on $\{0, 1\}^d$ with Bernoulli distributions (Appendix C.4).

3.5 Implementation Remarks

We conclude this section by a few practical remarks when implementing IGO algorithms.

3.5.1 Influence of the Selection Scheme w

The selection scheme w directly affects the update rule (18).

A natural choice is $w(u) = \mathbb{1}_{u \leqslant q}$. This results, as shown in Proposition 7, in an improvement of the q-quantile over the course of optimization. Taking $q = 1/2$ springs to mind (giving positive weights to the better half of the samples); however, this is often

not selective enough, and both theory and experiments confirm that for the Gaussian case, efficient optimization requires $q < 1/2$ (see Section 5.2). According to Beyer (2001), on the sphere function $f(x) = \sum_i x_i^2$, the optimal q is about 0.27 if sample size N is not larger than the search space dimension d, and even smaller otherwise (Jebalia and Auger, 2010).

Second, replacing w with $w + c$ for some constant c clearly has no influence on the IGO continuous-time flow (7), since the gradient will cancel out the constant. However, this is not the case for the update rule (18) with a finite sample of size N.

Indeed, adding a constant c to w adds a quantity $c\frac{1}{N}\sum \widetilde{\nabla}_\theta \ln P_\theta(x_i)$ to the update. In expectation, this quantity vanishes because the P_θ-expected value of $\widetilde{\nabla}_\theta \ln P_\theta$ is 0 (because $\int (\widetilde{\nabla}_\theta \ln P_\theta)\, P_\theta = \int \widetilde{\nabla} P_\theta = \widetilde{\nabla} 1 = 0$). So adding a constant to w does not change the expected value of the update, but it may change, e.g., its variance. The empirical average of $\widetilde{\nabla}_\theta \ln P_\theta(x_i)$ in the sample will be $O(1/\sqrt{N})$. So translating the weights results in a $O(1/\sqrt{N})$ change in the update. See also Section 4 in Sun et al. (2009).

Thus, one may be tempted to introduce a well chosen value of c so as to reduce the variance of the update. However, determining an optimal value for c is difficult: the optimal value minimizing the variance actually depends on possible correlations between $\widetilde{\nabla}_\theta \ln P_\theta$ and the function f. The only general result is that one should shift w such that 0 lies within its range. Assuming independence, or dependence with enough symmetry, the optimal shift is when the weights average to 0.

3.5.2 Complexity

The complexity of the IGO algorithm depends much on the computational cost model. In optimization, it is fairly common to assume that the objective function f is very costly compared to any other calculations performed by the algorithm (Moré et al., 1981; Dolan and Moré, 2002). Then the cost of IGO in terms of number of f-calls is N per iteration, and the cost of using inverse quantiles and computing the natural gradient is negligible.

Setting the cost of f aside, the complexity of the IGO algorithm depends mainly on the computation of the (inverse) Fisher matrix. Assume an analytical expression for this matrix is known. Then, with $p = \dim \Theta$ the number of parameters, the cost of storage of the Fisher matrix is $O(p^2)$ per iteration, and its inversion typically costs $O(p^3)$ per iteration. However, depending on the situation and on possible algebraic simplifications, strategies exist to reduce this cost (e.g., Le Roux et al. 2007 in a learning context). For instance, for CMA-ES the cost is $O(Np)$ (Suttorp et al., 2009). More generally, parametrization by expectation parameters (see above), when available, may reduce the cost to $O(Np)$ as well.

If no analytical form of the Fisher matrix is known and Monte Carlo estimation is required, then complexity depends on the particular situation at hand and is related to the best sampling strategies available for a particular family of distributions. For Boltzmann machines, for instance, a host of such strategies are available (Ackley et al., 1985; Salakhutdinov and Murray, 2008; Salakhutdinov, 2009; Desjardins et al., 2010). Still, in such a situation, IGO may be competitive if the objective function f is costly.

3.5.3 Recycling Old Samples

To compute the ranks of samples in (14), it might be advisable to re-use samples from previous iterations, so that a smaller number of samples is necessary, see e.g. Sun et al. (2009).

For $N = 1$, this is indispensable. In order to preserve sampling consistency (Theorem 6) the old samples need to be reweighted using the ratio of their likelihood under the current versus old distribution, as in importance sampling.

In evolutionary computation, elitist selection (also called plus-selection) is a common approach where the all-time best samples are taken into account in each iteration. Elitist selection can be modelled in the IGO framework by using the current all-time best samples in addition to samples from P_θ. Specifically, in the $(\mu + \lambda)$-selection scheme, we set $N = \mu + \lambda$ and let $x_1, \ldots, x_\mu$ be the current all-time μ best points. Then we sample λ new points, $x_{\mu+1}, \ldots, x_N$, from the current distribution P_θ and apply (16) with $w(q) = (N/\mu)\mathbb{1}_{q \leq \mu/N}$.

3.5.4 Initialization

As with other distribution-based optimization algorithms, it is usually a good idea to initialize in such a way as to cover a wide portion of the search space, i.e. θ^0 should be chosen so that P_{θ^0} has large diversity. For IGO algorithms this is particularly effective, since, as explained above, the natural gradient provides minimal change of diversity (greedily at each step) for a given change in the objective function.

4. IGO, Maximum Likelihood, and the Cross-Entropy Method

In this section we generalize the IGO update for settings where the natural gradient may not exist. This generalization reveals a unique IGO algorithm for finite step-sizes δt and a natural link to the cross-entropy method.

4.1 IGO as a Smooth-time Maximum Likelihood Estimate

The IGO flow turns out to be the only way to maximize a *weighted* log-likelihood, where points of the current distribution are slightly reweighted according to f-preferences.

This relies on the following interpretation of the natural gradient as a weighted maximum likelihood update with infinitesimal learning rate. This result singles out, in yet another way, the *natural* gradient among all possible gradients. The proof is given in Appendix D.

Theorem 10 (Natural gradient as ML with infinitesimal weights) *Let $\varepsilon > 0$ and $\theta_0 \in \Theta$. Let $W(x)$ be a function of x and let θ be the solution of*

$$\theta = \arg\max_\theta \left\{ (1-\varepsilon) \underbrace{\int \ln P_\theta(x)\, P_{\theta_0}(\mathrm{d}x)}_{= \text{const} - \mathrm{KL}(P_{\theta_0} \| P_\theta),\ \text{maximal for } \theta = \theta_0} + \varepsilon \int \ln P_\theta(x) \overbrace{W(x)}^{\text{preference weight biasing } P_{\theta_0}} P_{\theta_0}(\mathrm{d}x) \right\} \ . \tag{26}$$

Then, when $\varepsilon \to 0$ we have

$$\theta = \theta_0 + \varepsilon \int \widetilde{\nabla}_\theta \ln P_\theta(x)\ W(x)\, P_{\theta_0}(\mathrm{d}x) + O(\varepsilon^2) \ . \tag{27}$$

Likewise for discrete samples: with $x_1, \ldots, x_N \in X$, let θ be the solution of

$$\theta = \arg\max_\theta \left\{ (1-\varepsilon) \int \ln P_\theta(x)\, P_{\theta_0}(\mathrm{d}x) + \varepsilon \sum_i W(x_i)\ \ln P_\theta(x_i) \right\} \ . \tag{28}$$

Then when $\varepsilon \to 0$ we have

$$\theta = \theta_0 + \varepsilon \sum_i W(x_i)\ \widetilde{\nabla}_\theta \ln P_\theta(x_i) + O(\varepsilon^2) \ . \tag{29}$$

So if $W(x) = W^f_{\theta_0}(x)$ is the weight of the points according to quantilized f-preferences, the weighted maximum log-likelihood necessarily is the IGO flow (9) using the natural gradient—or the IGO update (17) when using samples.

Thus the IGO flow is the unique flow that, continuously in time, slightly changes the distribution to maximize the log-likelihood of points with good values of f. (In addition, IGO continuously updates the weight $W^f_{\theta^t}(x)$ depending on f and on the current distribution, so that we keep optimizing.)

This theorem suggests a way to approximate the IGO flow by enforcing this interpretation for a given non-infinitesimal step size δt, as follows.

Definition 11 (IGO-ML algorithm) *The* IGO-ML algorithm *with step size δt updates the value of the parameter θ^t according to*

$$\theta^{t+\delta t} = \arg\max_\theta \left\{ (1 - \delta t \sum_i \widehat{w}_i) \int \ln P_\theta(x)\, P_{\theta^t}(\mathrm{d}x) \ + \ \delta t \sum_i \widehat{w}_i \ln P_\theta(x_i) \right\} \tag{30}$$

where $x_1, \ldots, x_N$ are sample points drawn according to the distribution P_{θ^t}, and $\widehat{w}_i$ is the weight (14) *obtained from the ranked values of the objective function f.*

The IGO-ML algorithm is obviously independent of the parametrization θ: indeed it only depends on P_θ itself. Furthermore, the IGO-ML update (30) does not even require a smooth parametrization of the distribution anymore (though in this case, a small δt will likely result in stalling: $\theta^{t+\delta t} = \theta^t$ if the set of possible values for θ is discrete).

Like the cross-entropy method below, the IGO-ML algorithm can be applied only when the argmax can be computed.

It turns out that for exponential families, IGO-ML is just the IGO algorithm in a particular parametrization (see Theorem 12).

4.2 The Cross-Entropy Method

Taking $\delta t = 1$ in (30) above corresponds to a full maximum likelihood update; when using the truncation selection scheme w, this is the *cross-entropy method* (CEM). The cross-entropy method can be defined in an optimization setting as follows (de Boer et al., 2005). Like IGO, it depends on a family of probability distributions P_θ parametrized by $\theta \in \Theta$, and a number of samples N at each iteration. Let also $N_e = \lceil qN \rceil$ $(0 < q < 1)$ be a number of *elite* samples.

At each step, the cross-entropy method for optimization samples N points $x_1, \ldots, x_N$ from the current distribution P_{θ^t}. Let $\widehat{w}_i$ be $1/N_e$ if x_i belongs to the N_e samples with the best value of the objective function f, and $\widehat{w}_i = 0$ otherwise. Then the *cross-entropy method* or *maximum likelihoood* update (CEM/ML) for optimization is (de Boer et al., 2005, Algorithm 3.1)

$$\theta^{t+1} = \arg\max_\theta \sum \widehat{w}_i \ln P_\theta(x_i) \tag{31}$$

(assuming the argmax is tractable). This corresponds to $\delta t = 1$ in (30).

A commonly used version of CEM with a smoother update depends on a step size parameter $0 < \alpha \leqslant 1$ and is given (de Boer et al., 2005) by

$$\theta^{t+1} = (1-\alpha)\theta^t + \alpha \arg\max_\theta \sum \widehat{w}_i \ln P_\theta(x_i). \tag{32}$$

The standard CEM/ML update is $\alpha = 1$. For $\alpha = 1$, the standard cross-entropy method is independent of the parametrization θ, whereas for $\alpha < 1$ this is not the case.

Note the difference between the IGO-ML algorithm (30) and the smoothed CEM update (32) with step size $\alpha = \delta t$: the smoothed CEM update performs a weighted average of the parameter value *after* taking the maximum likelihood estimate, whereas IGO-ML uses a weighted average of current and previous likelihoods, *then* takes a maximum likelihood estimate. In general, these two rules can greatly differ, as they do for Gaussian distributions (Section 5.2).

This swapping of averaging makes IGO-ML parametrization-independent whereas the smoothed CEM update is not.

Yet, for exponential families of probability distributions, there exists one particular parametrization θ in which the IGO algorithm and the smoothed CEM update coincide. We now proceed to this construction.

4.3 IGO for Expectation Parameters and Maximum Likelihood

The particular form of IGO for exponential families has an interesting consequence if the parametrization chosen for the exponential family is the set of *expectation parameters*. Let $P_\theta(x) = \frac{1}{Z(\theta)} \exp\left(\sum \theta_j T_j(x)\right) H(\mathrm{d}x)$ be an exponential family as above. The *expectation parameters* are $\bar{T}_j = \bar{T}_j(\theta) = \mathbb{E}_{P_\theta} T_j$, (denoted η_j in Amari and Nagaoka 2000, Eq. 3.56). The notation $\bar{T}$ will denote the collection $(\bar{T}_j)$. We shall use the notation $P_{\bar{T}}$ to denote the probability distribution P parametrized by the expectation parameters.

It is well-known that, in this parametrization, the maximum likelihood estimate for a sample of points $x_1, \ldots, x_N$ is just the empirical average of the expectation parameters over that sample:

$$\arg\max_{\bar{T}} \frac{1}{N} \sum_{i=1}^N \ln P_{\bar{T}}(x_i) = \frac{1}{N} \sum_{i=1}^N T(x_i) \ . \tag{33}$$

In the discussion above, one main difference between IGO and smoothed CEM was whether we took averages before or after taking the maximum log-likelihood estimate. For the expectation parameters $\bar{T}_i$, we see that these operations commute. (One can say that these expectation parameters "linearize maximum likelihood estimates".) After some work we get the following result.

Theorem 12 (IGO, CEM and maximum likelihood) *Let*

$$P_\theta(\mathrm{d}x) = \frac{1}{Z(\theta)} \exp\left(\sum \theta_j T_j(x)\right) H(\mathrm{d}x)$$

be an exponential family of probability distributions, where the T_j are functions of x and H is some reference measure. Let us parametrize this family by the expected values $\bar{T}_j = \mathbb{E} T_j$.

Let us assume the chosen weights $\widehat{w}_i$ sum to 1. For a sample $x_1, \ldots, x_N$, let

$$T_j^* = \sum_i \widehat{w}_i T_j(x_i).$$

Then the IGO update (17) *in this parametrization reads*

$$\bar{T}_j^{t+\delta t} = (1-\delta t)\, \bar{T}_j^t + \delta t\, T_j^*. \tag{34}$$

Moreover these three algorithms coincide:

- *The IGO-ML algorithm* (30).
- *The IGO algorithm* (17) *written in the parametrization $\bar{T}_j$* (34).
- *The smoothed CEM algorithm* (32) *written in the parametrization $\bar{T}_j$, with $\alpha = \delta t$.*

Corollary 13 *For exponential families, the standard CEM/ML update* (31) *coincides with the IGO algorithm in parametrization $\bar{T}_j$ with $\delta t = 1$.*

Beware that the expectation parameters $\bar{T}_j$ are not always the most obvious parameters (Amari and Nagaoka, 2000, Section 3.5). For example, for 1-dimensional Gaussian distributions, the expectation parameters are the mean μ and the second moment $\mu^2 + \sigma^2$, not the mean and variance. When expressed back in terms of mean and variance, the update (34) boils down to $\mu \leftarrow (1-\delta t)\mu + \delta t \mu^*$ and $\sigma^2 \leftarrow (1-\delta t)\sigma^2 + \delta t (\sigma^*)^2 + \delta t (1-\delta t)(\mu^* - \mu)^2$, where μ^* and σ^* denote the mean and standard deviation of the samples x_i.

On the other hand, when using smoothed CEM with mean and variance as parameters, the new variance is $(1-\delta t)\sigma^2 + \delta t (\sigma^*)^2$, which can be significantly smaller for $\delta t \in (0,1)$. This proves, in passing, that the smoothed CEM update in other parametrizations is generally *not* an IGO algorithm (because it can differ at first order in δt).

The case of Gaussian distributions is further exemplified in Section 5.2 below: in particular, smoothed CEM in the (μ, σ) parametrization almost invariably exhibits a reduction of variance, often leading to premature convergence.

For these reasons we think that the IGO-ML algorithm is the sensible way to define an interpolated ML estimate for $\delta t < 1$ in a parametrization-independent way (see however the analysis of a critical δt in Section 5.2). In Appendix A we further discuss IGO and CEM and sum up the differences and relative advantages.

Taking $\delta t = 1$ is a bold approximation choice: the "ideal" continuous-time IGO flow itself, after time 1, does not coincide with the maximum likelihood update of the best points in the sample. Since the maximum likelihood algorithm is known to converge prematurely in some instances (Section 5.2), using the parametrization by expectation parameters with large δt may not be desirable.

The considerable simplification of the IGO update in these coordinates reflects the duality of coordinates $\bar{T}_i$ and θ_i. More precisely, the natural gradient ascent w.r.t. the parameters $\bar{T}_i$ is given by the vanilla gradient w.r.t. the parameters θ_i:

$$\widetilde{\nabla}_{\bar{T}_i} = \frac{\partial}{\partial \theta_i}$$

(Proposition 29 in Appendix D).

5. CMA-ES, NES, EDAs and PBIL from the IGO Framework

In this section we investigate the IGO algorithms for Bernoulli measures and for multivariate normal distributions, and show the correspondence to well-known algorithms. Restricted Boltzmann machines are given as a third, novel example. In addition, we discuss the influence of the parametrization of the distributions.

5.1 PBIL and cGA as IGO Algorithms for Bernoulli Measures

Let us consider on $X = \{0,1\}^d$ a family of Bernoulli measures $P_\theta(x) = p_{\theta_1}(x_1) \times \ldots \times p_{\theta_d}(x_d)$ with $p_{\theta_i}(x_i) = \theta_i^{x_i}(1-\theta_i)^{1-x_i}$, with each $\theta_i \in [0;1]$. As this family is a product of probability measures $p_{\theta_i}(x_i)$, the different components of a random vector y following P_θ are independent and all off-diagonal terms of the Fisher information matrix are zero. Diagonal terms are given by $\frac{1}{\theta_i(1-\theta_i)}$. Therefore the inverse of the Fisher matrix is a diagonal matrix with diagonal entries equal to $\theta_i(1-\theta_i)$. In addition, the partial derivative of $\ln P_\theta(x)$ w.r.t. θ_i is computed in a straightforward manner resulting in

$$\frac{\partial \ln P_\theta(x)}{\partial \theta_i} = \frac{x_i}{\theta_i} - \frac{1-x_i}{1-\theta_i} \ .$$

Let $x_1, \ldots, x_N$ be N samples at step t with distribution P_{θ^t} and let $x_{1:N}, \ldots, x_{N:N}$ be the samples ranked according to f value. The natural gradient update (18) with Bernoulli measures is then

$$\theta_i^{t+\delta t} = \theta_i^t + \delta t\, \theta_i^t(1-\theta_i^t) \sum_{j=1}^N w_j \left(\frac{[x_{j:N}]_i}{\theta_i^t} - \frac{1-[x_{j:N}]_i}{1-\theta_i^t} \right) \tag{35}$$

where $w_j = w((j-1/2)/N)/N$ and $[y]_i$ denotes the i^{th} coordinate of $y \in X$. The previous equation simplifies to

$$\theta_i^{t+\delta t} = \theta_i^t + \delta t \sum_{j=1}^N w_j \left([x_{j:N}]_i - \theta_i^t \right) \ , \tag{36}$$

or, denoting $\bar{w}$ the sum of the weights $\sum_{j=1}^N w_j$,

$$\theta_i^{t+\delta t} = (1 - \bar{w}\delta t)\, \theta_i^t + \delta t \sum_{j=1}^N w_j \, [x_{j:N}]_i \ . \tag{37}$$

The algorithm so obtained coincides with the so-called *population-based incremental learning* algorithm (PBIL, Baluja and Caruana 1995) for $N =$ `NUMBER_SAMPLES` and the appropriate (usually non-negative) weights w_j, as well as with the *compact genetic algorithm* (cGA, Harik et al. 1999) for $N = 2$ and $w_1 = -w_2$. Different variants of PBIL correspond to different choices of the selection scheme w. In cGA, components for which both samples have the same value are unchanged, because $w_1 + w_2 = 0$. We have thus proved the following.

Proposition 14 *The IGO algorithm on $\{0,1\}^d$ using Bernoulli measures parametrized by θ as above, coincides with the* compact Genetic Algorithm *(cGA) when $N = 2$ and $w_1 = -w_2$.*

Moreover, it coincides with Population-Based Incremental Learning *(PBIL) with the following correspondence of parameters. The PBIL algorithm using the μ best solutions, see Baluja and Caruana (1995, Figure 4), is recovered*[4] *using $\delta t =$ LR, $w_j = (1-\text{LR})^{j-1}$ for $j = 1, \ldots, \mu$, and $w_j = 0$ for $j = \mu+1, \ldots, N$.*

If the selection scheme of IGO is chosen as $w_1 = 1$, $w_j = 0$ for $j = 2, \ldots, N$, IGO recovers the PBIL/EGA algorithm with update rule towards the best solution (Baluja, 1994, Figure 4), with $\delta t =$ LR (the learning rate of PBIL) and MUT_PROBABILITY $= 0$ *(no random mutation of θ).*

Interestingly, the parameters θ_i are the expectation parameters described in Section 4: indeed, the expectation of x_i is θ_i. So the formulas above are particular cases of (34). Thus, by Theorem 12, PBIL is both a smoothed CEM in these parameters and an IGO-ML algorithm.

Let us now consider another, so-called "logit" representation, given by the logistic function $P(x_i = 1) = \frac{1}{1+\exp(-\tilde{\theta}_i)}$. This $\tilde{\theta}$ is the exponential parametrization of Section 3.3. We find that

$$\frac{\partial \ln P_{\tilde{\theta}}(x)}{\partial \tilde{\theta}_i} = (x_i - 1) + \frac{\exp(-\tilde{\theta}_i)}{1+\exp(-\tilde{\theta}_i)} = x_i - \mathbb{E} x_i \tag{38}$$

(cf. Eq. 19) and that the diagonal elements of the Fisher information matrix are given by $\exp(-\tilde{\theta}_i)/(1+\exp(-\tilde{\theta}_i))^2 = \operatorname{Var} x_i$ (as per Eq. 20). So the natural gradient update (18) with Bernoulli measures in parametrization $\tilde{\theta}$ reads

$$\tilde{\theta}_i^{t+\delta t} = \tilde{\theta}_i^t + \delta t (1+\exp(\tilde{\theta}_i^t)) \left(-\bar{w} + (1+\exp(-\tilde{\theta}_i^t)) \sum_{j=1}^N w_j [x_{j:N}]_i \right) \ . \tag{39}$$

To better compare the update with the previous representation, note that $\theta_i = \frac{1}{1+\exp(-\tilde{\theta}_i)}$ and thus we can rewrite

$$\tilde{\theta}_i^{t+\delta t} = \tilde{\theta}_i^t + \frac{\delta t}{\theta_i^t(1-\theta_i^t)} \sum_{j=1}^N w_j \left([x_{j:N}]_i - \theta_i^t \right) \ . \tag{40}$$

So the direction of the update is the same as before and is given by the proportion of bits set to 0 or 1 in the best samples, compared to its expected value under the current distribution. The magnitude of the update is different since the parameter $\tilde{\theta}$ ranges from $-\infty$ to $+\infty$ instead of from 0 to 1. We did not find this algorithm in the literature.

These updates also illustrate the influence of setting the sum of weights to 0 or not (Section 3.5). If, at some time, the first bit is equal to 1 both for a majority of good points and for a majority of bad points, then the original PBIL will increase the probability of setting the first bit to 1, which is counterintuitive. If the weights w_i are chosen to sum to 0 this noise effect disappears; otherwise, it disappears only on average.

4. Note that the pseudocode for the algorithm in Baluja and Caruana (1995, Figure 4) is slightly erroneous since it gives smaller weights to better individuals. The error can be fixed by updating the probability in reversed order, looping from `NUMBER_OF_VECTORS_TO_UPDATE_FROM` to 1. This was confirmed by S. Baluja in personal communication. We consider here the corrected version of the algorithm.

5.2 Multivariate Normal Distributions (Gaussians)

Evolution strategies (Rechenberg, 1973; Schwefel, 1995; Beyer and Schwefel, 2002) are black-box optimization algorithms for the continuous search domain, $X \subseteq \mathbb{R}^d$ (for simplicity we assume $X = \mathbb{R}^d$ in the following), which use multivariate normal distributions to sample new solutions. In the context of continuous black-box optimization, *Natural Evolution Strategies* (NES) introduced the idea of using a natural gradient update of the distribution parameters (Wierstra et al., 2008; Sun et al., 2009; Glasmachers et al., 2010; Wierstra et al., 2014). Surprisingly, the well-known *Covariance Matrix Adaption Evolution Strategy* (CMA-ES, Hansen and Ostermeier 1996, 2001; Hansen et al. 2003; Hansen and Kern 2004; Jastrebski and Arnold 2006) also turns out to conduct a natural gradient update of distribution parameters (Akimoto et al., 2010; Glasmachers et al., 2010).

Let $x \in \mathbb{R}^d$. As the most prominent example, we use mean vector $m = \mathbb{E}x$ and covariance matrix $C = \mathbb{E}(x-m)(x-m)^\mathsf{T} = \mathbb{E}(xx^\mathsf{T}) - mm^\mathsf{T}$ to parametrize a normal distribution via $\theta = (m, C)$. The IGO update in (17) or (18) in this parametrization can now be entirely formulated without the (inverse) Fisher matrix, similarly to (34) or (22). The complexity of the update is linear in the number of parameters (size of $\theta = (m, C)$, where $(d^2 - d)/2$ parameters are redundant).

Let us discuss known algorithms that implement updates of this kind.

5.2.1 CMA-ES.

The rank-μ-update CMA-ES implements the equations[5]

$$m^{t+1} = m^t + \eta_\mathrm{m} \sum_{i=1}^{N} \widehat{w}_i (x_i - m^t) \tag{41}$$

$$C^{t+1} = C^t + \eta_\mathrm{c} \sum_{i=1}^{N} \widehat{w}_i ((x_i - m^t)(x_i - m^t)^\mathsf{T} - C^t) \tag{42}$$

where $\widehat{w}_i$ are the weights based on ranked f-values, see (14) and (17).

Proposition 15 *The IGO update* (17) *for Gaussian distributions in the parametrization by mean and covariance matrix* (m, C)*, coincides with the CMA-ES update equations* (41) *and* (42) *with* $\eta_\mathrm{c} = \eta_\mathrm{m}$.

This result is essentially due to Akimoto et al. (2010) and Glasmachers et al. (2010), who showed that the CMA-ES update with $\eta_\mathrm{c} = \eta_\mathrm{m}$ is a natural gradient update[6].

However, in deviation from the IGO algorithm, the learning rates η_m and η_c are assigned different values if $N \ll \dim \Theta$ in CMA-ES[7]. Note that the Fisher information matrix is block-diagonal in m and C (Akimoto et al., 2010), so that application of the different learning rates and of the inverse Fisher matrix commute. Moreover, CMA-ES uses a *path cumulation* method to adjust the step sizes, which is not covered by the IGO framework (see Appendix A).

5. The CMA-ES implements these equations given the parameter setting $c_1 = 0$ and $c_\sigma = 0$ (or $d_\sigma = \infty$, see e.g. Hansen 2009) that disengages the effect of the rank-one update and of step size control and therefore of both so-called evolution paths.

6. In these articles the result has been derived for $\theta \leftarrow \theta + \eta \widetilde{\nabla}_\theta \mathbb{E}_{P_\theta} f$, see (11), leading to $f(x_i)$ in place of $\widehat{w}_i$. No assumptions on f have been used besides that it does not depend on θ. Consequently, by replacing f with $W^f_{\theta^t}$, where θ^t is fixed, the derivation holds equally well for $\theta \leftarrow \theta + \eta \widetilde{\nabla}_\theta \mathbb{E}_{P_\theta} W^f_{\theta^t}$.

7. Specifically, let $\sum |\widehat{w}_i| = 1$, then the settings are $\eta_\mathrm{m} = 1$ and $\eta_\mathrm{c} \approx 1/(d^2 \sum \widehat{w}_i^2)$ (Hansen, 2006b).

5.2.2 Convenient Reparametrizations Over Time.

For practical purposes, at each step it is convenient to work in a representation of θ in which the diagonal Fisher matrix $I(\theta^t)$ has a simple form, e.g., diagonal with simple diagonal entries. It is generally not possible to obtain such a representation for all θ simultaneously. Still it is always possible to find a transformation achieving a diagonal Fisher matrix at a single parameter θ^t, in multiple ways (it amounts to choosing a basis of parameter space which is orthogonal in the Fisher metric). Such a representation is never unique and not intrinsic, yet it still provides a convenient way to write the algorithms.

For CMA-ES, one such representation can be found by sending the current covariant matrix C^t to the identity, e.g., by representing the mean and covariance matrix by $((C^t)^{-1/2} m, (C^t)^{-1/2} C (C^t)^{-1/2})$ instead of (m, C). Then the Fisher matrix $I(\theta^t)$ at (m^t, C^t) becomes diagonal. The next algorithm we discuss, xNES (Glasmachers et al., 2010), exploits this possibility in a logarithmic representation of the covariance matrix.

5.2.3 Natural Evolution Strategies.

Natural evolution strategies (NES, Wierstra et al. 2008; Sun et al. 2009) implement (41) as well, while using a Cholesky decomposition of C as the parametrization for the update of the variance parameters. The resulting update that replaces (42) is neither particularly elegant nor numerically efficient. The more recent xNES (Glasmachers et al., 2010) chooses an "exponential" parametrization that naturally depends on the current parameters. This leads to an elegant formulation where the additive update in exponential parametrization becomes a multiplicative update for C. With $C = AA^\mathsf{T}$, the matrix update reads

$$A \leftarrow A \times \exp\left(\frac{\eta_\mathrm{c}}{2} \sum_{i=1}^{N} \widehat{w}_i \times (z_i z_i^\mathsf{T} - \mathrm{I}_d) \right) \tag{43}$$

where $z_i = A^{-1}(x_i - m)$ and I_d is the identity matrix. From (43) the updated covariance matrix is $C \leftarrow A \times \exp\left(\eta_\mathrm{c} \sum_{i=1}^{N} \widehat{w}_i \times (z_i z_i^\mathsf{T} - \mathrm{I}_d)\right) \times A^\mathsf{T}$.

Compared to (42), the update has the advantage that also negative weights, $\widehat{w}_i < 0$, always lead to a feasible covariance matrix. By default, xNES sets $\eta_\mathrm{m} \neq \eta_\mathrm{c}$ in the same circumstances as in CMA-ES, but contrary to CMA-ES the past evolution path is not taken into account (Glasmachers et al., 2010).

When $\eta_\mathrm{c} = \eta_\mathrm{m}$, xNES is consistent with the IGO flow (8), and implements an IGO algorithm (17) slightly generalized in that it uses a θ^t-dependent parametrization, which represents the current covariance matrix by 0. Namely, we have:

Proposition 16 (exponential IGO update of Gaussians) *Let* (m^t, C^t) *be the current mean and covariance matrix. Let* $C^t = AA^\mathsf{T}$*. Let* θ *be the time-dependent parametrization of the space of Gaussian distributions, which parametrizes the Gaussian distribution* (m, C) *by*

$$\theta = (m, R), \qquad R = \ln(A^{-1} C (A^\mathsf{T})^{-1})$$

where ln *is the logarithm of positive matrices. (Note that the current value* C^t *of* C *is represented as* $R = 0$*.)*

Then the IGO update (17) *in the parametrization* θ *is as follows: the mean* m *is updated as in CMA-ES* (41)*, and the parameter* R *is updated as*

$$R \leftarrow \delta t \sum_{i=1}^{N} \widehat{w}_i \times (A^{-1}(x_i - m)(x_i - m)^\mathsf{T}(A^\mathsf{T})^{-1} - \mathrm{I}_d) \tag{44}$$

thus resulting in the same update as (43) *(with* $\eta_\mathrm{c} = \delta t$*) for the covariance matrix:* $C \leftarrow A \exp(R) A^\mathsf{T}$.

Proof Indeed, by basic differential geometry, if parametrization $\theta' = \varphi(\theta)$ is used, the IGO update for θ' is $D\varphi(\theta^t)$ applied to the IGO update for θ, where $D\varphi$ is the differential of φ. Here, given the update (42) for C, to find the update for R we have to compute the differential of the map $C \mapsto \ln(A^{-1}C(A^\mathsf{T})^{-1})$ taken at $C = AA^\mathsf{T}$: for any matrix M we have $\ln(A^{-1}(AA^\mathsf{T} + \varepsilon M)(A^\mathsf{T})^{-1}) = \varepsilon\, A^{-1}M(A^\mathsf{T})^{-1} + O(\varepsilon^2)$. So to find the update for the variable R we have to apply $A^{-1} \dots (A^\mathsf{T})^{-1}$ to the update (42) for C. ■

5.2.4 Cross-Entropy Method and EMNA

Estimation of distribution algorithms (EDA) and the *cross-entropy method* (CEM, Rubinstein 1999; Rubinstein and Kroese 2004) estimate a new distribution from a censored sample. Generally, the new parameter value can be written as

$$\begin{aligned} \theta_\mathrm{maxLL} &= \arg\max_\theta \sum_{i=1}^{N} \widehat{w}_i \ln P_\theta(x_i) \\ &\longrightarrow_{N\to\infty} \arg\max_\theta \mathbb{E}_{P_{\theta^t}} W^f_{\theta^t} \ln P_\theta \end{aligned} \tag{45}$$

by Theorem 6. Here, the weights $\widehat{w}_i$ are equal to $1/\mu$ for the μ best points (censored or elitist sample) and 0 otherwise. This θ_maxLL maximizes the weighted log-likelihood of $x_1, \dots, x_N$; equivalently, it minimizes the cross-entropy and the Kullback–Leibler divergence to the distribution of the best μ samples[8].

For Gaussian distributions, Equation (45) can be explicitly written in the form

$$m^{t+1} = m^* = \sum_{i=1}^{N} \widehat{w}_i x_i \tag{46}$$

$$C^{t+1} = C^* = \sum_{i=1}^{N} \widehat{w}_i (x_i - m^*)(x_i - m^*)^\mathsf{T} \tag{47}$$

the empirical mean and variance of the elite sample.

8. Let P_w denote the distribution of the weighted samples: $\Pr(x = x_i) = \widehat{w}_i$ and $\sum_i \widehat{w}_i = 1$. Then the cross-entropy between P_w and P_θ reads $\sum_i P_w(x_i) \ln 1/P_\theta(x_i)$ and the KL divergence reads $\mathrm{KL}(P_w \,\|\, P_\theta) = \sum_i P_w(x_i) \ln 1/P_\theta(x_i) - \sum_i P_w(x_i) \ln 1/P_w(x_i)$. Minimization of both terms in θ result in θ_maxLL.

Equations (46) and (47) also define the simplest continuous domain EDA, the *estimation of multivariate normal algorithm* ($\mathrm{EMNA}_\mathrm{global}$, Larranaga and Lozano 2002). Interestingly, (46) and (47) only differ from (41) and (42) (with $\eta_\mathrm{m} = \eta_\mathrm{c} = 1$) in that the new mean m^{t+1} is used instead of m^t in the covariance matrix update (Hansen, 2006b).

The smoothed CEM (32) in this parametrization thus writes

$$m^{t+\delta t} = (1 - \delta t) m^t + \delta t m^* \tag{48}$$

$$C^{t+\delta t} = (1 - \delta t) C^t + \delta t C^* \ . \tag{49}$$

Note that *this is not an IGO algorithm* (i.e., there is no parametrization of the family of Gaussian distributions in which the IGO algorithm coincides with update Eq. 49): indeed, all IGO algorithms coincide at first order in δt when $\delta t \to 0$ (because they recover the IGO flow), while this update for $C^{t+\delta t}$ does not coincide with (42) in this limit, due to the use of m^* instead of m^t. This does not contradict Theorem 12: smoothed CEM is an IGO algorithm only if smoothed CEM is written in the expectation parametrization, which (m, C) is not.

5.2.5 CMA-ES, Smoothed CEM, and IGO-ML

Let us compare IGO-ML (30), rank-μ CMA (41)–(42), and smoothed CEM (48)–(49) in the parametrization with mean and covariance matrix. These algorithms all update the distribution mean in the same way, while the update of the covariance matrix depends on the algorithm. With learning rate δt, these updates are computed to be

$$\begin{aligned} m^{t+1} &= (1 - \delta t)\, m^t + \delta t\, m^* \\ C^{t+1} &= (1 - \delta t)\, C^t + \delta t\, C^* + \delta t (1 - \delta t)^j\, (m^* - m^t)(m^* - m^t)^\mathsf{T} \ , \end{aligned} \tag{50}$$

for different values of j, where m^* and C^* are the mean and covariance matrix computed over the elite sample (with positive weights $\widehat{w}_i$ summing to one) as above. The rightmost term of (50) is reminiscent of the so-called rank-one update in CMA-ES (not included in Eq. 42).

For $j = 0$ we recover the rank-μ CMA-ES update (42), for $j = 1$ we recover IGO-ML, and for $j = \infty$ we recover smoothed CEM (the rightmost term is absent). The case $j = 2$ corresponds to an update that uses m^{t+1} instead of m^t in (42) (with $\eta_\mathrm{m} = \eta_\mathrm{c} = \delta t$). For $0 < \delta t < 1$, the larger j, the smaller C^{t+1}. For $\delta t = 1$, IGO-ML and smoothed CEM/EMNA realize θ_maxLL from (45)–(47).

For $\delta t \to 0$, the update is independent of j at first order in δt if $j < \infty$: this reflects compatibility with the IGO flow of CMA-ES and of IGO-ML, but not of smoothed CEM.

In the default (full) CMA-ES (as opposed to rank-μ CMA-ES), the coefficient preceeding $(m^* - m^t)(m^* - m^t)^\mathsf{T}$ in (50) reads approximately $3\delta t$, where the additional $2\delta t$ originate from the so-called rank-one update and are moreover modulated by a "cumulated path" up to a factor of about $\sqrt{d}$ (Hansen and Auger, 2014).

5.2.6 Critical δt

Let us assume that $\mu < N$ weights are set to $\widehat{w}_i = 1/\mu$ and the remaining weights to zero, so that the selection ratio is $q = \mu/N$.

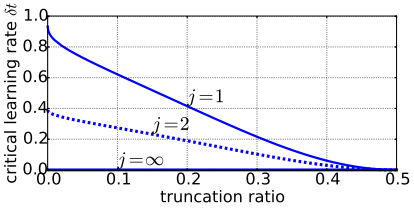


Figure 1: Critical δt versus selection truncation ratio q for three values of j in (50). With δt above the critical δt, the variance decreases systematically when optimizing a linear function, indicating failure of the algorithm. For CMA-ES/NES where $j = 0$, the critical δt for $q < 0.5$ is infinite.

Then there is a critical value of δt depending on this ratio q, such that above this critical δt the algorithms given by IGO-ML and smoothed CEM are prone to premature convergence. Indeed, let f be a linear function on $\mathbb{R}^d$, and consider the variance in the direction of the gradient of f. Assuming further $N \to \infty$ and $q \leq 1/2$, then the variance C^* of the elite sample is smaller than the current variance C^t, by a constant factor. Depending on the precise update for C^{t+1}, if δt is too large, the variance C^{t+1} is going to be smaller than C^t by a constant factor as well. This implies that the algorithm is going to stall, i.e., the variance will go to 0 before the optimum is reached. (On the other hand, the continuous-time IGO flow corresponding to $\delta t \to 0$ does not stall, see Appendix C.4.)

We now study the critical δt (in the limit $N \to \infty$) below which the algorithm does not stall (this is done similarly to the analysis in Appendix C.4 for linear functions). For IGO-ML, ($j = 1$ in Eq. 50, or equivalently for the smoothed CEM in the expectation parameters $(m, C + mm^\mathsf{T})$, see Section 4), the variance increases if and only if δt is smaller than the critical value $\delta t_{\text{crit}} = qb\sqrt{2\pi}e^{b^2/2}$ where b is the percentile function of q, i.e. b is such that $q = \int_b^\infty e^{-x^2/2}/\sqrt{2\pi}$. This value δt_{crit} is plotted as a solid line in Fig. 1. For $j = 2$, δt_{crit} is smaller, related to the above by $\delta t_{\text{crit}} \leftarrow \sqrt{1 + \delta t_{\text{crit}}} - 1$ and plotted as a dashed line in Fig. 1. For CEM ($j = \infty$), the critical δt is zero (reflecting the non-IGO behavior of CEM in this parametrization). For CMA-ES ($j = 0$), the critical δt is infinite for $q < 1/2$. When the selection ratio q is above $1/2$, for all algorithms the critical δt becomes zero.

5.2.7 Gaussian Distributions with Restricted Parametrization

When considering a restricted parametrization of multivariate normal distributions, IGO recovers other known algorithms. In particular for sep-CMA-ES (Ros and Hansen, 2008) and SNES (Schaul et al., 2011), the update has been restricted to the diagonal of the covariance matrix.

5.3 IGO for Restricted Boltzmann Machines, and Multimodal Optimization

As a third example after Bernoulli and Gaussian distributions, we have applied IGO to restricted Boltzmann machines (RBMs, Smolensky 1986; Ackley et al. 1985), which are

families of distributions on the search space $X = \{0, 1\}^d$ extending Bernoulli distributions to allow for dependencies between the bits (see, e.g., Berny 2002 for Boltzmann machines used in an optimization setting). The details are given in Appendix B. We chose this example for two reasons.

First, it illustrates how to obtain a novel algorithm from a family of probability distributions in the IGO framework, in a case when there is no fully explicit expression for the Fisher information matrix.

Second, it illustrates the influence of the natural gradient on diversity and its relevance to multimodal optimization. The RBM probability distributions on $\{0, 1\}^d$ are multimodal, contrary to Bernoulli distributions. Thus, in principle, IGO on such distributions could reach several optima simultaneously. This allows for a simple testing of the entropy-maximizing property of the IGO flow (Proposition 2). Accordingly, for a given improvement on the objective function, we would expect IGO to favor preserving the diversity of P_θ. Indeed on a simple objective function with two optima, we find that the IGO flow converges to a distribution putting weight on both optima, while a descent using the vanilla gradient always only concentrates around one optimum. This might be relevant for multimodal optimization, where simultaneous handling of multiple hypotheses is seen as useful for optimization (Sareni and Krähenbühl, 1998; Das et al., 2011), or in situations in which several valleys appear equally good at first but only one of them contains the true optimum.

We refer to Appendix B for a reminder about Boltzmann machines, for how IGO rolls out in this case, and for the detailed experimental setup for studying the influence of IGO in a multimodal situation.

6. Summary and Conclusion

We sum up:

- The information-geometric optimization (IGO) framework derives from invariance principles the uniquely defined IGO flow (Definition 4), and allows, via discretization in time and space, to build optimization algorithms from any family of distributions on any search space. In some instances (Gaussian distributions on $\mathbb{R}^d$ or Bernoulli distributions on $\{0, 1\}^d$) it recovers versions of known algorithms (PBIL, CMA-ES, cGA, NES); in other instances (restricted Boltzmann machine distributions) it produces new, hopefully efficient optimization algorithms.

- The use of a quantile-based, time-dependent transform of the objective function, Equation (5), provides a rigorous derivation of rank-based update rules, frequently used in current optimization algorithms. Theorem 6 uniquely identifies the infinite-population limit of these update rules.

- The IGO flow is singled out by its equivalent description as an infinitesimal weighted maximum log-likelihood update (Theorem 10). In a particular parametrization and with a step size of 1, IGO recovers the cross-entropy method (Corollary 13). This reveals a connection between CEM and the natural gradient, and allows to define a new, fully parametrization-invariant smoothed maximum likelihood update, the IGO-ML.

- Theoretical arguments suggest that the IGO flow minimizes the change of diversity in the course of optimization. As diversity usually decreases in the course of optimization, IGO algorithms tend to exhibit minimal diversity loss for the observed improvement. In particular, starting with high diversity and using multimodal distributions may allow simultaneous exploration of multiple optima of the objective function. Preliminary experiments with restricted Boltzmann machines confirm this effect in a simple situation.

Thus, the IGO framework provides sound theoretical foundations to optimization algorithms based on probability distributions. In particular, this viewpoint helps to bridge the gap between continuous and discrete optimization.

The invariance properties, which reduce the number of arbitrary choices, together with the relationship between natural gradient and diversity, may contribute to a theoretical explanation of the good practical performance of those currently used algorithms, such as CMA-ES, which can be interpreted as instantiations of IGO.

We hope that invariance properties will acquire in computer science the importance they have in mathematics, where intrinsic thinking is the first step for abstract linear algebra or differential geometry, and in modern physics, where the notions of invariance w.r.t. the coordinate system and so-called gauge invariance play a central role.

Acknowledgments

The authors would like to thank Michèle Sebag for the acronym and for helpful comments. We also thank Youhei Akimoto for helpful feedback and inspiration. Y. O. would like to thank Cédric Villani and Bruno Sévennec for helpful discussions on the Fisher metric. A. A. and N. H. would like to acknowledge the Dagstuhl Seminar No 10361 on the Theory of Evolutionary Computation (`http://www.dagstuhl.de/10361`) for crucially inspiring this paper, their work on natural gradients and beyond. This work was partially supported by the ANR-2010-COSI-002 grant (SIMINOLE) of the French National Research Agency.

Appendix A. Further Discussion and Perspectives

This appendix touches upon further aspects and perspectives of the IGO framework and its implementations.

A.1 A Single Framework for Optimization on Arbitrary Spaces

A strength of the IGO viewpoint is to automatically provide a distinct and arguably in some sense optimal optimization algorithm from any family of probability distributions on any given space, discrete or continuous. IGO algorithms feature desired invariance properties and thereby make fewer arbitrary choices.

In particular, IGO describes several well-known optimization algorithms within a single framework. For Bernoulli distributions, to the best of our knowledge, PBIL or cGA have never been identified as a natural gradient ascent in the literature[9]. For Gaussian distributions, algorithms of the same form (17) had been developed previously (Hansen and Ostermeier, 2001; Wierstra et al., 2008) and their close relationship with a natural gradient ascent had been recognized (Akimoto et al., 2010; Glasmachers et al., 2010). These works, however, also strongly suggest that IGO algorithms may need to be complemented with further heuristics to achieve efficient optimization algorithms. In particular, learning rate settings and diversity control (step-size) deviate from the original framework. These deviations mostly stem from time discretization and the choice of a finite sample size, both necessary to derive an IGO *algorithm* from the IGO *flow*. Further work is needed to fully understand this from a theoretical viewpoint.

The wide applicability of natural gradient approaches seems not to be widely known in the optimization community (though see Malagò et al. 2008).

A.2 About Invariance

The role of invariance is two-fold. First, invariance breaks otherwise arbitrary choices in the algorithm design. Second, and more importantly, invariance implies generalization of behavior from single problem instances to entire problem classes (with the caveat to choose the initial state of the algorithm accordingly), thereby making the outcome of optimization more predictable.

Optimization problems faced in reality—and algorithms used in practice—are often far too complex to be amenable to a rigorous mathematical analysis. Consequently, the judgement of algorithms in practice is to a large extent based in empirical observations, either on artificial benchmarks or on the experience with real-world problems. Invariance, as a guarantee of generalization, has an immediate impact on the relevance of any such empirical observations and, in the same line of reasoning, can be interpreted as a notion of robustness.

A.3 About Quantiles

The IGO flow in (8) has, to the best of our knowledge, never been defined before. The introduction of the quantile-rewriting (5) of the objective function provides the first rigorous

9. Thanks to Jonathan Shapiro for giving an early argument confirming this property (personal communication).

derivation of quantile- or rank- or comparison-based optimization from a gradient ascent in θ-space.

NES and CMA-ES have been claimed to maximize $-\mathbb{E}_{P_\theta} f$ via natural gradient ascent (Wierstra et al., 2008; Akimoto et al., 2010). However, we have proved that the NES and CMA-ES updates actually converge to the IGO flow, not to the similar flow with the gradient of $\mathbb{E}_{P_\theta} f$ (Theorem 6). So we find that in reality these algorithms maximize $\mathbb{E}_{P_\theta} W^f_{\theta^t}$, where $W^f_{\theta^t}$ is a decreasing transformation of the f-quantiles under the current sample distribution.

Moreover, in practice, maximizing $-\mathbb{E}_{P_\theta} f$ tends to be a rather unstable procedure and has been discouraged, see for example Whitley (1989) and Sun et al. (2009).

A.4 About Choice of P_θ: Learning a Model of Good Points

The choice of the family of probability distributions P_θ plays a double role.

First, it is analogous to choosing the variation operators (namely *mutation* or *recombination*) as seen in evolutionary algorithms: indeed, P_θ encodes possible moves according to which new sample points are explored.

Second, optimization algorithms using distributions can be interpreted as learning a probabilistic model of where the points with good values lie in the search space. With this point of view, P_θ describes *richness of this model*: for instance, restricted Boltzmann machines with h hidden units can describe distributions with up to 2^h modes, whereas the Bernoulli distribution is unimodal. This influences, for instance, the ability to explore several valleys and optimize multimodal functions in a single run.

More generally, the IGO framework makes it tempting to use more complex models of where good points lie, inspired, e.g., from machine learning, and adapt them for optimization. The restricted Boltzmann machines of Appendix B are a first step in this direction. The initial idea behind these machines is that each hidden unit controls a block of coordinates of the search space (a block of features), so that the optimization algorithm hopefully builds a good model of which features must be activated or de-activated together to obtain good values of f. This is somewhat reminiscent of a crossover operator: if observation of good points shows that a block of features go together, this information is stored in the RBM structure and this block may be later activated as a whole, thus effectively transferring blocks of features from one good solution to another. Inspired by models of deep learning (Bengio et al., 2012), one might be tempted to stack such models on top of each other, so that optimization would operate on a more and more abstract representation of the problem. IGO and the natural gradient might help in exploiting the added expressivity that comes with richer models: in our simple experiment, the vanilla gradient ignores the additional expressivity of RBMs with respect to Bernoulli distributions (Appendix B). The downside of a richer model is that either the sample size N must be increased or the learning rate δt must be decreased to obtain a stable algorithm (a situation analogous to overfitting in machine learning: richer models will be quicker to concentrate too much around a few observed good samples).

A.5 Natural Gradient and Parametrization Invariance

Central to IGO is the use of the natural gradient, which follows from θ-invariance (parametrization invariance) and makes sense on any search space, discrete or continuous.

While the IGO flow is exactly θ-invariant, for any practical implementation of an IGO algorithm, a parametrization choice has to be made. Different parametrizations lead to different algorithms and larger values of δt are likely to result in more differing algorithms. Still, since all IGO algorithms approximate the IGO flow, two parametrizations in combination with IGO will differ less than the same two parametrizations in combination with another algorithm (such as the vanilla gradient or the smoothed CEM method)—at least if the learning rate δt is not too large. The chosen parametrization becomes more relevant as the step size δt increases.

On the other hand, natural evolution strategies have not emphasized θ-invariance: the chosen parametrization (Cholesky, exponential) has been considered as a defining feature. We believe the term "natural evolution strategy" should rather be used independently of the chosen parameterization, thereby referring to the usage of the natural gradient as the main principle for the update of distribution parameters.

A.6 IGO, Maximum Likelihood and Cross-Entropy

The cross-entropy method (CEM, de Boer et al. 2005) can be used to produce optimization algorithms given a family of probability distributions on an arbitrary space, by performing a jump to a maximum likelihood estimate of the parameters.

We have seen (Corollary 13) that the standard CEM is an IGO algorithm *in a particular parametrization*, with a learning rate δt equal to 1. However, it is well-known, both theoretically and experimentally (Branke et al., 2007; Hansen, 2006b; Wagner et al., 2004), that standard CEM loses diversity too fast in many situations. The usual solution (de Boer et al., 2005) is to reduce the learning rate (smoothed CEM, Equation 32), but this breaks the reparametrization invariance of non-smoothed CEM.

On the other hand, the IGO flow can be seen as a *maximum likelihood update with infinitesimal learning rate* (Theorem 10). This interpretation allows to define a particular IGO algorithm, the IGO-ML (Definition 11): it performs a maximum likelihood update with an arbitrary learning rate, and keeps the reparametrization invariance. It coincides with CEM when the learning rate is set to 1, but it differs from smoothed CEM by the exchange of the order of argmax and averaging (compare Equations 30 and 32), and coincides with the IGO flow for small learning rates. We argue that this new, fully invariant algorithm is the conceptually better way to reduce the learning rate and achieve smoothing in CEM.

Standard CEM with rate 1 can lose diversity, yet is a particular case of an IGO algorithm: this illustrates the fact that reasonable values of the learning rate δt depend on the parametrization. We have studied this phenomenon in detail for various Gaussian IGO algorithms (Section 5.2).

Why would a smaller learning rate perform better than a large one in an optimization setting? It might seem more efficient to jump directly to the maximum likelihood estimate of currently known good points, instead of performing an iterated gradient ascent towards this maximum. Due to having only a limited number of samples, however, optimization faces a "moving target", contrary to a learning setting in which the example distribution is often stationary. Currently known good points heavily depend on the current distribution and are likely not to indicate the *position* at which the optimum lies, but, rather, the broad

direction in which the optimum is to be found. After each update, the next elite sample points are going to be located somewhere new.

A.7 Diversity and Multiple Optima

The IGO framework emphasizes the relation of natural gradient and diversity: we argued that IGO provides minimal diversity change for a given objective function improvement. In particular, provided the initial diversity is large, diversity is kept at a maximum after the upate. This theoretical relationship can be observed experimentally for restricted Boltzmann machines (Appendix B).

On the other hand, using the vanilla gradient does not lead to a balanced distribution between the two optima in our experiments. Using the vanilla gradient introduces hidden arbitrary choices between those points (more exactly between moves in Θ-space). This results in unnecessary and unwelcome loss of diversity, and might also be detrimental at later stages in the optimization. This may reflect the fact that the Euclidean metric on the space of parameters, implicitly used in the vanilla gradient, becomes less and less meaningful for gradient descent on complex distributions.

IGO and the natural gradient are certainly relevant to the well-known problem of exploration-exploitation balance: as we have seen, arguably the natural gradient realizes the largest improvement in the objective with the least possible change of diversity in the distribution.

More generally, like other distribution-based optimization algorithms, IGO tries to learn a model of where the good points are. This is typical of machine learning, one of the contexts for which the natural gradient was studied. The conceptual relationship of IGO and IGO-like optimization algorithms with machine learning is still to be explored and exploited.

We now present some ideas which we believe would be worth exploring.

A.8 Adaptive Learning Rate

Comparing consecutive updates to evaluate a learning rate or step size is an effective measure. For example, in back-propagation, the update sign has been used to adapt the learning rate of each single weight in an artificial neural network (Silva and Almeida, 1990). In CMA-ES, a step size is adapted depending on whether recent steps tended to move in a consistent direction or to backtrack. This is measured by considering the changes of the mean m of the Gaussian distribution.

For a probability distribution P_θ on an arbitrary search space, in general no notion of mean may be defined. However, it is still possible to define "backtracking" in the evolution of θ as follows.

Consider two successive updates $\delta\theta^t = \theta^t - \theta^{t-\delta t}$ and $\delta\theta^{t+\delta t} = \theta^{t+\delta t} - \theta^t$. Their scalar product in the Fisher metric $I(\theta^t)$ is

$$\langle \delta\theta^t, \delta\theta^{t+\delta t} \rangle = \sum_{ij} I_{ij}(\theta^t)\, \delta\theta^t_i\, \delta\theta^{t+\delta t}_j \quad .$$

Dividing by the associated norms will yield the cosine $\cos\alpha$ of the angle between $\delta\theta^t$ and $\delta\theta^{t+\delta t}$.

If this cosine is positive, the learning rate δt may be increased. If the cosine is negative, the learning rate probably needs to be decreased. Various schemes for the change of δt can be devised; for instance, inspired by step size updates commonly used in evolution strategies, one can multiply δt by $\exp(\beta(\cos\alpha))$ or $\exp(\beta\,\mathrm{sign}(\cos\alpha))$, where $\beta \approx \min(N/\dim\Theta, 1/2)$. As this cosine can be quite noisy, cumulation over several time steps might be advisable.

As before, this scheme is constructed to be robust w.r.t. reparametrization of θ, thanks to the use of the Fisher metric. However, for large learning rates δt, in practice the parametrization might well become relevant.

A.9 Geodesic Parametrization

While the IGO flow is fully invariant under θ-reparametrization, an IGO algorithm does depend on the choice of parametrization for θ, even if for small δt the difference between two IGO algorithms is $O(\delta t^2)$, one order of magnitude smaller than between IGO and vanilla gradient in general.

So one can wonder how to discretize time in the IGO flow in a fully intrinsic way, not depending at all on a parametrization for θ. A first possibility is given by the IGO-ML algorithm (Definition 11)—this means, for exponential families, that we can decide to single out the parametrization by expectation parameters.

Another, more geometric solution is to use *geodesics* on the statistical manifold. This means we approximate the trajectories of the IGO flow by successive geodesic segments of length δt in the Fisher metric, where the initial direction of each segment is given by the direction of the IGO flow (16). This defines an approximation to the IGO flow that depends on the step size δt and sample size N, but *not* on any choice of parametrization. This approach is fully developed in Bensadon (2015).

A.10 Finite Sample Size and Noisy IGO Flow

The IGO flow is an ideal model of the IGO algorithms. But the IGO flow is deterministic while IGO algorithms are stochastic, depending on a finite number N of random samples. This might result in important differences in their behavior and one can wonder if there is a way to reflect stochasticity directly in the definition of the IGO flow.

The IGO update (16) is a stochastic update

$$\theta^{t+\delta t} = \theta^t + \delta t \sum_{i=1}^{N} \widehat{w}_i\ \widetilde{\nabla}_\theta \ln P_\theta(x_i)\Big|_{\theta=\theta^t}$$

because the term $\sum_{i=1}^N \widehat{w}_i\ \widetilde{\nabla}_\theta \ln P_\theta(x_i)\Big|_{\theta=\theta^t}$ involves a random sample. As such, this term has an expectation and a variance. So for a fixed N and δt, this random update is a weak approximation with step size δt (Kloeden and Platen, 1992, Chapter 9.7) of a stochastic differential equation on θ, whose drift is the expectation of the IGO update (which tends to the IGO flow when $N \to \infty$), and whose noise term is $\sqrt{\delta t}$ times the square root of the covariance matrix of the update applied to a normal random vector.

Such a stochastic differential equation, defining a *noisy IGO flow*, might be a better theoretical object with which to compare the actual behavior of IGO algorithms, than the ideal noiseless IGO flow.

For instance, this strongly suggests that if we have $\delta t \to 0$ while N is kept fixed in an IGO algorithm, noise will disappear (compare Remark 2 in Akimoto et al. 2012).

Second, for large N, one expects the variance of the IGO update to scale like $1/N$, so that the noise term will scale like $\sqrt{\delta t/N}$. This formally suggests that, within reasonable bounds, multiplying or dividing both N and δt by the same factor should result in similar behavior of the algorithm, so that for instance it should be reasonable to reset δt to $10\delta t/N$ and N to 10. (Note that the cost in terms of f-calls of these two algorithms is similar.)

This dependency is reflected in evolution strategies in several ways, provided N is smaller than the search space dimension. First, theoretical results for large search space dimension on the sphere function, $f(x) = \|x\|^2$, indicate that the optimal step size δt for the mean vector is proportional to N, provided the weighting function w is either truncation selection with a fixed truncation ratio (Beyer, 2001) or optimal weights (Arnold, 2006). Second, the learning rate δt of the covariance matrix in CMA-ES is chosen proportional to $\left(\sum_{i=1}^N \widehat{w}_i\right)^2 / \sum_{i=1}^N \widehat{w}_i^2$ which is again proportional to N (Hansen and Kern, 2004). For small enough N, the progress per f-call is then in both cases rather independent of the choice of N.

These results suggest that in implementations, N can be chosen rather freely, whereas δt will be set to $\text{const} \cdot N$. The constant is chosen small enough to ensure stability, but as large as possible to maximize speed; still, $\delta t \leq 1$ is another constraint for (very) large N.

A.11 Influence of the Fisher Geometry of the Statistical Manifold

The global Riemannian geometry of the statistical manifold P_θ might have a bearing on the behavior of stochastic algorithms exploring this manifold. For instance, the Fisher metric identifies the set of 1-dimensional normal distributions $\mathcal{N}(m, \sigma^2)$ with the two-dimensional hyperbolic plane. The latter has negative curvature. The sign of curvature has a strong influence on the behavior of random walks in a Riemannian manifold: in particular, in negative curvature, successive random errors tend to not compensate as much as in the Euclidean case (because geodesics diverge more quickly); this might be relevant to the settings of a stochastic optimization algorithm, suggesting to use larger sample size or smaller steps when curvature is negative; Bensadon (2015) provides first observations in this direction, associated with negative curvature in the space of Gaussians. This is speculative and remains to be explored.

Appendix B. Implementing an IGO Algorithm with a New Family of Probability Distributions: Restricted Boltzmann Machines

In this appendix we show how to apply IGO to restricted Boltzmann machines (RBMs, Smolensky 1986; Ackley et al. 1985), a family of probability distributions on the discrete hypercube that extends Bernoulli distributions and can represent correlations between variables.

This first illustrates how to set up an IGO algorithm from a family of probability distributions for which no fully explicit expression for the Fisher information matrix is available.

Second, we test the relationship between IGO and diversity of p_θ, in view of the entropy-maximizing property of the IGO flow (Proposition 2). We consider a simple function with

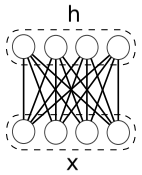


Figure 2: The RBM architecture with the observed ($\mathbf{x}$) and latent ($\mathbf{h}$) variables. In our experiments, a single hidden unit was used.

two optima, and compare the behavior of the IGO flow with that of a vanilla gradient flow. The latter always converges to a single optimum, even though RBMs allow for multimodal distributions. On the other hand, IGO seems to always converge to both optima at once, taking advantage of RBM multimodality.

Finally we interpret this observations by pointing out a non-obvious breach of symmetry between 0 and 1 on $\{0,1\}^d$ for the vanilla gradient of RBMs; the natural gradient automatically compensates for this.

B.1 Restricted Boltzmann Machines.

RBMs first define a joint distribution on $\mathbf{x} \in \{0,1\}^d$ together with a *hidden* or *latent* variable $\mathbf{h} \in \{0,1\}^{d_h}$ (Ghahramani, 2004). Summation over $\mathbf{h}$ provides the distribution over $\mathbf{x}$. The probability associated with an observation $\mathbf{x} = (x_i) \in \{0,1\}^d$ and latent variable $\mathbf{h} = (h_j) \in \{0,1\}^{d_h}$ is

$$P_\theta(\mathbf{x}, \mathbf{h}) = \frac{e^{-E(\mathbf{x},\mathbf{h})}}{\sum_{\mathbf{x}',\mathbf{h}'} e^{-E(\mathbf{x}',\mathbf{h}')}}\,, \qquad P_\theta(\mathbf{x}) = \sum_{\mathbf{h}} P_\theta(\mathbf{x}, \mathbf{h}), \tag{51}$$

where the *energy function* E is

$$E(\mathbf{x}, \mathbf{h}) = -\sum_i a_i x_i - \sum_j b_j h_j - \sum_{i,j} w_{ij} x_i h_j \tag{52}$$

(compare Section 3.3). The distribution is parametrized by $\theta = (\mathbf{a}, \mathbf{b}, \mathbf{w})$. The "biases" $\mathbf{a}$ and $\mathbf{b}$ can be subsumed into the weights $\mathbf{w}$ by introducing variables x_0 and h_0 always equal to one, which we implicitly do from now on.

The hidden variable $\mathbf{h}$ introduces correlations between the components of $\mathbf{x}$ (see Figure 2). For instance, with $d_h = 1$, the distribution on $\mathbf{x}$ is the sum of two Bernoulli distributions with $\mathbf{h}$ acting as a "switch".

B.2 IGO for RBMs.

RBMs constitute an exponential family with latent variables (with the statistics $T(\mathbf{x}, \mathbf{h})$ being all the $x_i h_j$). So the IGO flow (10) is given by (25) where the first contribution is the Fisher matrix (24) and the second contribution is the log-probability derivatives weighted

by the objective function values. However, these expressions are not explicit due to the expectations under P_θ. Still, these expectations can be replaced with Monte Carlo sampling; this is what we will do.

Actually, when applying IGO to such distributions to optimize an objective function $f(\mathbf{x})$, we have two choices. The first is to see the objective function $f(\mathbf{x})$ as a function of $(\mathbf{x}, \mathbf{h})$ where $\mathbf{h}$ is a dummy variable; then we can use the IGO algorithm to optimize over $(\mathbf{x}, \mathbf{h})$ using the distributions $P_\theta(\mathbf{x}, \mathbf{h})$. A second possibility is to marginalize $P_\theta(\mathbf{x}, \mathbf{h})$ over the hidden units $\mathbf{h}$ as in (51), to define the distribution $P_\theta(\mathbf{x})$; then we can use the IGO algorithm to optimize f over $\mathbf{x}$ using $P_\theta(\mathbf{x})$.

These two approaches yield slightly different algorithms. The Fisher matrix for the distributions $P_\theta(\mathbf{x}, \mathbf{h})$ is given by (20) (exponential families) whereas the one for the distributions $P_\theta(\mathbf{x})$ is given by (24) (exponential families with latent variables). For instance, with $I_{w_{ij} w_{i'j'}}$ denoting the entry of the Fisher matrix corresponding to the components w_{ij} and $w_{i'j'}$ of the parameter θ, from (20) we get

$$I_{w_{ij} w_{i'j'}}(\theta) = \mathrm{Cov}_{P_\theta}(x_i h_j, x_{i'} h_{j'}) = \mathbb{E}_{P_\theta}[x_i h_j x_{i'} h_{j'}] - \mathbb{E}_{P_\theta}[x_i h_j] \, \mathbb{E}_{P_\theta}[x_{i'} h_{j'}] \tag{53}$$

whereas from (24) we get the same expression in which each h_j is replaced with its expectation $\bar{h}_j$ knowing $\mathbf{x}$ namely $\bar{h}_j = \mathbb{E}_{P_\theta}[h_j|\mathbf{x}] = \left(1 + e^{-b_j - \sum_i x_i w_{ij}}\right)^{-1}$ and likewise for $h_{j'}$.

Both versions were tested on a small instance and found to be viable. However the version using $(\mathbf{x}, \mathbf{h})$ is numerically more stable and requires fewer samples to get a reliable (in particular, invertible) estimate of the Fisher matrix, both in practice and theory[10]. For this reason we selected the first approach: we optimize f as a function of $(\mathbf{x}, \mathbf{h})$ using IGO for the probability distributions $P_\theta(\mathbf{x}, \mathbf{h})$.

A practical IGO update is thus obtained from (21) by replacing the expectations with Monte Carlo samples $(\mathbf{x}, \mathbf{h})$ from P_θ. A host of strategies are available to sample from RBMs (Ackley et al., 1985; Salakhutdinov and Murray, 2008; Salakhutdinov, 2009; Desjardins et al., 2010); we simply used Gibbs sampling (Hinton., 2002). So the IGO update reads

$$\theta^{t+\delta t} = \theta^t + \delta t \, \widehat{\mathrm{Cov}}(T, T)^{-1} \, \widehat{\mathrm{Cov}}(T, W^f_{\theta^t}) \tag{54}$$

where T denotes the vector of all statistics $T_{ij}(\mathbf{x}, \mathbf{h}) = x_i h_j$ corresponding to the component w_{ij} of the parameter θ, and where $\widehat{\mathrm{Cov}}$ denotes the empirical covariance over a set of Monte Carlo samples $(\mathbf{x}, \mathbf{h})$ taken from $P_{\theta^t}(\mathbf{x}, \mathbf{h})$. Thus $\hat{I} = \widehat{\mathrm{Cov}}(T, T)$ is the estimated Fisher matrix, a matrix of size $\dim(\theta)^2$ as in (53). $\widehat{\mathrm{Cov}}(T, W^f_{\theta^t})$ is a vector of size $\dim(\theta)$ giving the correlation, in the Monte Carlo sample, between the statistics T_{ij} and the ranked values $W^f_{\theta^t}$ of the objective function of the points $(\mathbf{x}, \mathbf{h})$ in the sample: this vector is the sum of weighted log-probability derivatives in the IGO update (17), thanks to (19).

Different sample sizes N_{Fish} and N can be used to evaluate $\widehat{\mathrm{Cov}}(T, T)$ and $\widehat{\mathrm{Cov}}(T, W^f_{\theta^t})$. The sample size for $\widehat{\mathrm{Cov}}(T, W^f_{\theta^t})$ is just the IGO parameter N, the number of points on which the objective function has to be evaluated; it is typically kept small especially if f is costly. On the other hand it is important to obtain a reliable (in particular, invertible) estimate of the Fisher matrix: invertibility requires a number of samples at least, and ideally much larger than, $\dim(\theta)$, because each point in the sample contributes a rank-1 term to the empirical covariance matrix $\widehat{\mathrm{Cov}}(T, T)$. Increasing N_{Fish} does not require additional f-calls.

10. Indeed, if $I_1(\theta)$ is the Fisher matrix at θ in the first approach and $I_2(\theta)$ in the second approach, we always have $I_1(\theta) \geqslant I_2(\theta)$ in the sense of positive-definite matrices. This is because probability distributions on the pair $(\mathbf{x}, \mathbf{h})$ carry more information than their projections on $\mathbf{x}$ only, and so the Kullback–Leibler distances will always be larger. In particular, there exist values of θ for which the Fisher matrix I_2 is not invertible whereas I_1 is.

B.3 An Experiment with Two Optima: IGO, Diversity, and Multimodal Optimization.

We tested the resulting IGO trajectories on a simple objective function with two optima on $\{0, 1\}^d$, namely, the *two-min function based at* $\mathbf{y}$ defined as

$$f_{\mathbf{y}}(\mathbf{x}) = \min\left(\sum_i |x_i - y_i| \, , \sum_i |(1 - x_i) - y_i|\right) \tag{55}$$

which, as a function of $\mathbf{x}$, has two optima, one at $\mathbf{x} = \mathbf{y}$ and the other at its binary complement $\mathbf{x} = \bar{\mathbf{y}}$. The value of the base point $\mathbf{y}$ was randomized for each independent run.

We ran both the IGO algorithm as described above, and a version using the vanilla gradient instead of the natural gradient (that is, omitting the Fisher matrix in the IGO update).

The dimension was $d = 40$ and we used an RBM with only one latent variable ($d_h = 1$). Therefore $\dim(\theta) = 81$. We used a large sample size of $N = 10,000$ for Monte Carlo sampling, so as to be close to the theoretical IGO flow behavior. We also tested a smaller, more realistic sample size of $N = 10$ (still keeping $N_{\text{Fish}} = 10,000$), with similar but noisier results. The selection scheme (Section 2.2) was $w(q) = \mathbb{1}_{q \leqslant 1/5}$ (cf. Rechenberg 1994) so that the best 20% points in the sample are given weight 1 for the update.

The RBM was initialized so that at startup, the distribution P_{θ^0} is close to uniform on $(\mathbf{x}, \mathbf{h})$, in line with Proposition 2. Explicitly we set $w_{ij} \leftarrow \mathcal{N}(0, \frac{1}{d.d_h})$ and then $b_j \leftarrow -\sum_i \frac{w_{ij}}{2}$ and $a_i \leftarrow -\sum_j \frac{w_{ij}}{2} + \mathcal{N}(0, \frac{0.01}{d^2})$ which ensure a close-to-uniform initial distribution.

Full experimental details, including detailed setup and additional results, can be found in a previous version of this article (Ollivier et al., 2011, Section 5). (In particular, IGO runs are frozen when the estimated Fisher matrix becomes singular or unreliable.) The code used for these experiments can be found at `http://www.ludovicarnold.com/projects:igocode` .

The results show that, most of the time, with IGO the distribution P_{θ^t} converges to a distribution giving positive mass to both optima; on the other hand, over 300 independent runs, the same algorithm using the vanilla gradient only converges to one optimum at the expense at the other, so that the vanilla gradient *never* exploited the possibility offered by the RBM to create a bimodal probability distribution on $\{0, 1\}^d$. Figure 3 shows ten random runs (out of 300 in our experiments) of the two algorithms: for each of the two optima we plot its distance to the nearest of the points drawn from P_{θ^t}, as a function of time t.[11]

11. Note that the value of δt is not directly comparable between the natural and vanilla gradients. Theory suggests that at startup the IGO trajectory with δt is most comparable to the vanilla gradient trajectory with $4\delta t$, because from (53) most of the diagonal terms of the Fisher matrix are equal to $1/4$ and most off-diagonal terms are 0 at startup. The experiments confirm that this yields roughly comparable convergence speeds.

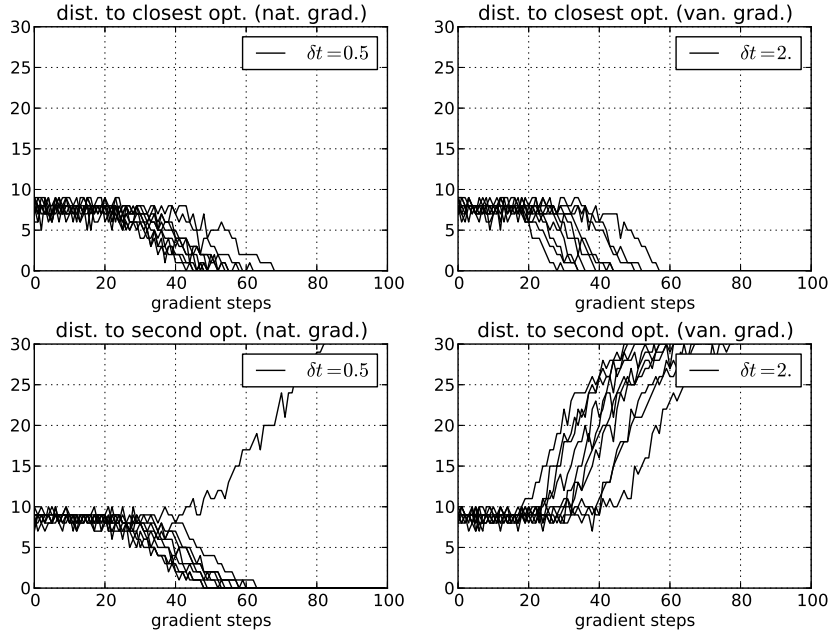


Figure 3: Distance to the two optima during 10 IGO optimization runs and 10 vanilla gradient runs. For each optimum, we plot its distance to the nearest point among the samples from P_{θ^t} found at each step.

This is even clearer when looking at the hidden variable $\mathbf{h}$. With $d_h = 1$ the two possible values $h = 0$ and $h = 1$ can create a sum of two Bernoulli distributions on $\mathbf{x}$. Figure 4 plots the average value of h in the sample for IGO and for the vanilla gradient. As can be seen, over IGO optimization the balance between the two modes is preserved, whereas using vanilla gradient optimization the zero mode is lost.

We interpret this as an illustration of Proposition 2: for a given improvement on the objective function, IGO will favor preserving the diversity (entropy) of P_θ. Using more richly multimodal distributions (e.g., RBMs with $d_h > 1$), this might be useful for multimodal optimization or for optimization situations in which there are several almost equally deep valleys only one of which contains the true optimum.

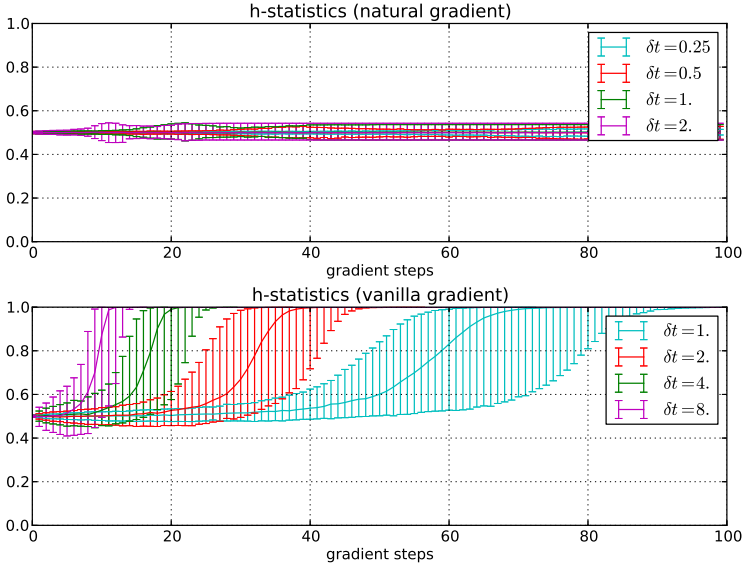


Figure 4: Median over 300 runs of the average of h in the sample, for IGO and vanilla gradient optimization. Error bars indicate the 16th and 84th quantile over the runs.

B.4 Breach of Symmetry by the Vanilla Gradient.

This experiment reveals that, curiously, the vanilla gradient loses multimodality by always setting the hidden variable h to 1, not to 0 (Figure 4). So the vanilla gradient for RBMs seems to favor $h = 1$.

Of course, exchanging the values 0 and 1 for the hidden variables in a restricted Boltzmann machine still gives a distribution of another Boltzmann machine. Changing h_j into $1 - h_j$ is equivalent to resetting $a_i \leftarrow a_i + w_{ij}$, $b_j \leftarrow -b_j$, and $w_{ij} \leftarrow -w_{ij}$.

IGO and the natural gradient are impervious to such a change by Proposition 19. But the vanilla gradient implicitly relies on the Euclidean norm on parameter space, as explained in Section 2.1. For this norm, the distance between the RBM distributions (a_i, b_j, w_{ij}) and (a'_i, b'_j, w'_{ij}) is $\sum_i |a_i - a'_i|^2 + \sum_j \left|b_j - b'_j\right|^2 + \sum_{ij} \left|w_{ij} - w'_{ij}\right|^2$. However, the change of variables $a_i \leftarrow a_i + w_{ij}$, $b_j \leftarrow -b_j$, $w_{ij} \leftarrow -w_{ij}$ does *not* preserve this Euclidean metric. Thus, exchanging 0 and 1 for the hidden variables will result in two different vanilla gradient ascents. The observed asymmetry on h is a consequence of this implicit asymmetry.

Of course it is possible to use parametrizations for which the vanilla gradient will be more symmetric: for instance, using $-1/1$ instead of $0/1$ for the variables, or rewriting the

energy as

$$E(\mathbf{x},\mathbf{h}) = -\sum_i A_i(x_i - \tfrac{1}{2}) - \sum_j B_j(h_j - \tfrac{1}{2}) - \sum_{i,j} W_{ij}(x_i - \tfrac{1}{2})(h_j - \tfrac{1}{2}) \qquad (56)$$

with "bias-free" parameters A_i, B_j, W_{ij} related to the usual parametrization by $w_{ij} = W_{ij}$, $a_i = A_i - \frac{1}{2}\sum_j w_{ij}$, and $b_j = B_j - \frac{1}{2}\sum_i w_{ij}$. The vanilla gradient might perform better in this parametrization.

However, we adopted the approach of using a family of probability distributions found in the literature, with the parametrization commonly found in the literature. We then used the vanilla gradient and the natural gradient on these distributions—and indeed the vanilla gradient, or an approximation thereof, is routinely applied to RBMs in the literature to optimize the log-likelihood of data (Hinton., 2002; Hinton et al., 2006; Bengio et al., 2007). It was not obvious a priori (at least for us) that the vanilla gradient ascent favors $h = 1$. Since the first version of this article was written, this phenomenon has been recognized for Boltzmann machines (Montavon and Müller, 2012).

This directly illustrates the specific influence of the chosen gradient (the two implementations only differ by the inclusion of the Fisher matrix): the natural gradient offers a systematic way to recover symmetry from a non-symmetric gradient update.

Symmetry alone does not explain why IGO reaches the two optima simultaneously: a symmetry-preserving stochastic algorithm could as well end up on either single optimum with 50% probability in each run. The diversity-preserving property of IGO (Proposition 2) offers a reasonable interpretation of why this does not happen.

Appendix C. Further Mathematical Properties of the IGO Flow

This appendix provides further mathematical properties of the IGO flow in general and in specific scenarios.

C.1 Invariance Properties

Here we formally state the invariance properties of the IGO flow under various reparametrizations. Since these results follow from the very construction of the algorithm, the proofs are omitted.

Proposition 17 (f-invariance) *Let $\varphi : \mathbb{R} \to \mathbb{R}$ be a strictly increasing function. Then the trajectories of the IGO flow when optimizing the functions f and $\varphi(f)$ are the same.*

The same is true for the discretized algorithm with population size N and step size $\delta t > 0$.

Proposition 18 (θ-invariance) *Let $\theta' = \varphi(\theta)$ be a smooth bijective function of θ and let $P'_{\theta'} = P_{\varphi^{-1}(\theta')}$. Let θ^t be the trajectory of the IGO flow when optimizing a function f using the distributions P_θ, initialized at θ^0. Then the IGO flow trajectory $(\theta')^t$ obtained from the optimization of the function f using the distributions $P'_{\theta'}$, initialized at $(\theta')^0 = \varphi(\theta^0)$, is the same, namely $(\theta')^t = \varphi(\theta^t)$.*

For the algorithm with finite N and $\delta t > 0$, invariance under reparametrization of θ is only true approximately, in the limit when $\delta t \to 0$. As mentioned above, the IGO update

(17), with $N = \infty$, is simply the Euler approximation scheme for the ordinary differential equation (8) defining the IGO flow. At each step, the Euler scheme is known to make an error $O(\delta t^2)$ with respect to the true flow. This error actually depends on the parametrization of θ.

So the IGO updates for different parametrizations coincide at first order in δt, and may, in general, differ by $O(\delta t^2)$. For instance the difference between the CMA-ES and xNES updates is indeed $O(\delta t^2)$, see Section 5.2.

For comparison, using the vanilla gradient results in a divergence of $O(\delta t)$ at each step between different parametrizations, so this divergence could be of the same magnitude as the steps themselves.

In that sense, one can say that IGO algorithms are "more parametrization-invariant" than other algorithms. This stems from their origin as a discretization of the IGO flow.

However, if the map φ is affine then this phenomenon disappears: parametrizations that differ by an affine map on θ yield the same IGO algorithm.

The next proposition states that, for example, if one uses a family of distributions on $\mathbb{R}^d$ which is invariant under affine transformations, then IGO algorithms optimize equally well a function and its image under any affine transformation (up to an obvious change in the initialization). This proposition generalizes the well-known corresponding property of CMA-ES (Hansen and Auger, 2014, Proposition 9).

This invariance under X-transformations only holds provided the X-transformation preserves the "shape" of the family of probability distributions P_θ, as follows.

Let us define, as usual, the image (*pushforward*) of a probability distribution P by a transformation $\varphi : X \to X$ as the probability distribution P' such that $P'(Y) = P(\varphi^{-1}(Y))$ for any subset $Y \subset X$ (Schilling, 2005, Chapter 7). In the continuous domain, the density of the new distribution P' is obtained by the usual change of variable formula involving the Jacobian of φ (Schilling, 2005, Chapter 15).

We say that a transformation $\varphi : X \to X$ *globally preserves* a family of probability distributions (P_θ), if the image of any P_θ by φ is equal to some distribution $P_{\theta'}$ in the same family, and if moreover the correspondence $\theta \mapsto \theta'$ is locally a diffeomorphism.

Proposition 19 (X-invariance) *Let $\varphi : X \to X$ be a one-to-one transformation of the search space which globally preserves the family of measures P_θ. Let θ^t be the IGO flow trajectory for the optimization of function f, initialized at P_{θ^0}. Let $(\theta')^t$ be the IGO flow trajectory for optimization of $f \circ \varphi^{-1}$, initialized at the image of P_{θ^0} by φ. Then $P_{(\theta')^t}$ is the image of P_{θ^t} by φ.*

For the discretized algorithm with population size N and step size $\delta t > 0$, the same is true up to an error of $O(\delta t^2)$ per iteration. This error disappears if the action of the map φ on Θ by pushforward is affine.

The latter case of affine transforms is well exemplified by CMA-ES: here, using the variance and mean as the parametrization of Gaussians, the new mean and variance after an affine transform of the search space are an affine function of the old mean and variance; specifically, for the affine transformation $A : x \mapsto Ax + b$ we have $(m, C) \mapsto (Am + b, ACA^\mathsf{T})$. Another example, on the discrete search space $X = \{0,1\}^d$, is the exchange of 0 and 1: for reasonable choices of the family P_θ, the IGO flow and IGO algorithms will be invariant under such a change in the way the data is presented.

C.2 Speed of the IGO Flow

Proposition 20 *The speed of the IGO flow, i.e. the norm of $\frac{\mathrm{d}\theta^t}{\mathrm{d}t}$ in the Fisher metric, is at most $\sqrt{\int_0^1 w^2 - (\int_0^1 w)^2}$ where w is the selection scheme.*

The proof is given in Appendix D.

A bounded speed means that the IGO flow will not explode in finite time, or go out-of-domain if the Fisher metric on the statistical manifold Θ is complete (for instance, the IGO flow on Gaussian distributions will not yield non-positive or degenerate covariance matrices). Due to the approximation terms $O(\delta t^2)$, this may not be true of IGO algorithms.

This speed can be monitored in practice in at least two ways. The first is just to compute the Fisher norm of the increment $\theta^{t+\delta t} - \theta^t$ using the Fisher matrix; for small δt this is close to $\delta t \| \frac{\mathrm{d}\theta}{\mathrm{d}t} \|$ with $\| \cdot \|$ the Fisher metric. The second is as follows: since the Fisher metric coincides with the Kullback–Leibler divergence up to a factor $1/2$, we have $\mathrm{KL}(P_{\theta^{t+\delta t}} \,\|\, P_{\theta^t}) \approx \frac{1}{2} \delta t^2 \| \frac{\mathrm{d}\theta}{\mathrm{d}t} \|^2$ at least for small δt. Since it is relatively easy to estimate $\mathrm{KL}(P_{\theta^{t+\delta t}} \,\|\, P_{\theta^t})$ by comparing the new and old log-likelihoods of points in a Monte Carlo sample, one can obtain an estimate of $\| \frac{\mathrm{d}\theta}{\mathrm{d}t} \|$.

Corollary 21 *Consider an IGO algorithm with selection scheme w, step size δt and sample size N. Then, with probability* 1*,*

$$\frac{\mathrm{KL}(P_{\theta^{t+\delta t}} \,\|\, P_{\theta^t})}{\delta t^2} \leqslant \frac{1}{2} \mathrm{Var}_{[0,1]}\, w + o(1)_{(\delta t, N) \to (0,\infty)} \ .$$

For instance, with $w(q) = \mathbb{1}_{q \leqslant q_0}$ and neglecting the error terms, an IGO algorithm introduces at most $\frac{1}{2} \delta t^2\, q_0 (1 - q_0)$ bits of information (in base e) per iteration into the probability distribution P_θ. The proof is given in Appendix D.

Thus, the time discretization parameter δt is not just an arbitrary variable: it has an intrinsic interpretation related to a number of bits introduced at each step of the algorithm. This kind of relationship suggests, more generally, to use the Kullback–Leibler divergence as an external and objective way to measure learning rates in those optimization algorithms which use probability distributions.

The result above is only an upper bound. Maximal speed can be achieved only if all "good" points point in the same direction. If the various good points in the sample suggest moves in inconsistent directions, then the IGO update will be much smaller. While non-consistent moves are generally to be expected if $N < \dim \Theta$, it may also be a sign that the signal is noisy, or that the family of distributions P_θ is not well suited to the problem at hand and should be enriched.

As an example, using a family of Gaussian distributions with unkown mean and fixed identity variance on $\mathbb{R}^d$, one checks that for the optimization of a linear function on $\mathbb{R}^d$, with the weight $w(u) = \mathbb{1}_{u<1/2}$, the IGO flow moves at constant speed $1/\sqrt{2\pi} \approx 0.4$, whatever the dimension d. On a rapidly varying sinusoidal function, the moving speed will be much slower because there are "good" and "bad" points in all directions.

This may suggest ways to design the selection scheme w to achieve maximal speed in some instances. Indeed, looking at the proof of the proposition, which involves a Cauchy–Schwarz inequality, one can see that the maximal speed is achieved only if there is a linear relationship

between the weights $W_\theta^f(x)$ and the gradient $\widetilde{\nabla}_\theta \ln P_\theta(x)$. For instance, for the optimization of a linear function on $\mathbb{R}^d$ using Gaussian measures of known variance, the maximal speed will be achieved when the selection scheme $w(u)$ is the inverse of the Gaussian cumulative distribution function. (In particular, $w(u)$ tends to $+\infty$ when $u \to 0$ and to $-\infty$ when $u \to 1$.) This is in accordance with known results: the expected value of the i-th order statistic of N standard Gaussian variates is the optimal $\widehat{w}_i$ value for Gaussians on the sphere function, $f(x) = \sum_i x_i^2$, where $d \to \infty$ (Beyer, 2001; Arnold, 2006). For $N \to \infty$, this order statistic converges to the inverse Gaussian cumulative distribution function.

C.3 Noisy Objective Functions

Suppose that the objective function f is non-deterministic: each time we ask for the value of f at a point $x \in X$, we get a random result. In this setting we may write the random value $f(x)$ as $f(x) = \tilde{f}(x, \omega)$ where ω is an unseen random seed, and $\tilde{f}$ is a deterministic function of x and ω. Without loss of generality, up to a change of variables we can assume that ω is uniformly distributed in $[0, 1]$.

We can still use the IGO algorithm without modification in this context. One might wonder which properties (consistency of sampling, etc.) still apply when f is not deterministic. Actually, IGO algorithms for noisy functions fit very nicely into the IGO framework: the following proposition allows to transfer any property of IGO to the case of noisy functions.

Proposition 22 (Noisy IGO) *Let f be a random function of $x \in X$, namely, $f(x) = \tilde{f}(x, \omega)$ where ω is a random variable uniformly distributed in $[0, 1]$, and $\tilde{f}$ is a deterministic function of x and ω. Then the two following algorithms coincide:*

- *The IGO algorithm* (16)*, using a family of distributions P_θ on space X, applied to the noisy function f, and where the samples are ranked according to the random observed value of f (here we assume that, for each sample, the noise ω is independent from everything else);*
- *The IGO algorithm on space $X \times [0, 1]$, using the family of distributions $\tilde{P}_\theta = P_\theta \otimes U_{[0,1]}$, applied to the deterministic function $\tilde{f}$. Here $U_{[0,1]}$ denotes the uniform law on $[0, 1]$.*

The (easy) proof is given in the Appendix D.

This proposition states that noisy optimization can be modeled as ordinary distribution-based optimization with a component of the distribution being independent of the control parameter θ. Conversely, any component of the search space in which a distribution-based optimization algorithm cannot perform selection or specialization will effectively act as a random noise on the objective function.

As a consequence of this result, all properties of IGO can be transferred to the noisy case. Consider, for instance, consistency of sampling (Theorem 6). The N-sample IGO update rule for the noisy case is identical to the non-noisy case (17):

$$\theta^{t+\delta t} = \theta^t + \delta t\, I^{-1}(\theta^t) \sum_{i=1}^{N} \widehat{w}_i \ \frac{\partial \ln P_\theta(x_i)}{\partial \theta}\bigg|_{\theta=\theta^t}$$

where each weight $\widehat{w}_i$ computed from (14) now incorporates noise from the objective function because the rank of x_i is computed on the random function, or equivalently on the deterministic function $\tilde{f}$: $\mathrm{rk}(x_i) = \#\{j, \tilde{f}(x_j, \omega_j) < \tilde{f}(x_i, \omega_i)\}$.

Consistency of sampling (Theorem 6) thus takes the following form through Proposition 22: When $N \to \infty$, the N-sample IGO update rule on the noisy function f converges with probability 1 to the update rule

$$\begin{aligned}\theta^{t+\delta t} &= \theta^t + \delta t\, \widetilde{\nabla}_\theta \int_0^1 \int W^{\tilde{f}}_{\theta^t}(x,\omega)\, P_\theta(\mathrm{d}x)\, \mathrm{d}\omega \\ &= \theta^t + \delta t\, \widetilde{\nabla}_\theta \int \bar{W}^f_{\theta^t}(x)\, P_\theta(\mathrm{d}x) \end{aligned} \tag{57}$$

where $\bar{W}^f_\theta(x) = \mathbb{E}_\omega W^{\tilde{f}}_\theta(x,\omega)$. Thus when $N \to \infty$ the update becomes deterministic as the effects of noise get averaged, as could be expected.

Thus, applying the IGO algorithm to noisy objective functions as in Proposition 22 requires defining the IGO flow for noisy objective functions as

$$\frac{\mathrm{d}\theta^t}{\mathrm{d}t} = \widetilde{\nabla}_\theta \int \bar{W}^f_{\theta^t}(x)\, P_\theta(\mathrm{d}x) \tag{58}$$

where $\bar{W}^f_\theta(x) = \mathbb{E}_\omega W^{\tilde{f}}_\theta(x,\omega)$ is the *average* weight of x over the values $f(x)$. Thus, the effects of noise remain visible even for $N \to \infty$, as $\bar{W}$ is in general flatter than W.

Note that there is another possible way to define the IGO flow for noisy objective functions. The flow (58) amounts to taking a point x, computing a (random) value $f(x)$, computing the level $q^<_\theta(x) = \Pr_{x'\sim P_\theta}(f(x') < f(x))$ of this random value $f(x)$, applying the selection scheme w, and then averaging. In this approach the value $q^<_\theta(x)$ is a random variable depending on the value of $f(x)$. Alternatively, we could define the value $q_\theta(x)$ by applying Definition 3 unchanged, namely, $q^<_\theta(x) = \Pr_{x'\sim P_\theta}(f(x') < f(x))$ in which $f(x)$ itself is treated as a random variable under the probability Pr, so that $q^<_\theta(x)$ is deterministic and takes into account all possible values for $f(x)$. Then we could apply the selection scheme w to this value $q^<_\theta(x)$ as in Definition 3 and define the IGO flow accordingly. The value of $q^<_\theta(x)$ in this second version is the expected value of $q^<_\theta(x)$ in the first version.

When the selection scheme w is affine, these two approaches coincide; however this is not the case in general, as the second version averages the q-values while the first version averages the weights $w(q)$. The second version would be the $N \to \infty$ limit of a slightly more complex algorithm using several evaluations of f for each sample x_i in order to compute noise-free average ranks and quantiles. The first version has the advantage of leaving the algorithm unchanged and of inheriting properties from the deterministic case via Proposition 22.

C.4 The IGO Flow for Linear Functions on $\{0,1\}^d$ and $\mathbb{R}^d$

In this section we take a closer look at the IGO differential equation solutions of (8) for some simple examples of objective functions, for which it is possible to obtain exact information about these IGO trajectories.

We start with the discrete search space $X = \{0,1\}^d$ and linear functions (to be minimized) defined as $f(x) = c - \sum_{i=1}^d \alpha_i x_i$ with $\alpha_i > 0$. (So maximization of the classical onemax function $f_{\mathrm{onemax}}(x) = \sum_{i=1}^d x_i$ is covered by setting $\alpha_i = 1$.) The differential equation of

the IGO flow (8) for the Bernoulli measures $P_\theta(x) = p_{\theta_1}(x_1)\dots p_{\theta_d}(x_d)$ defined on X is the $\delta t \to 0$ limit of the IGO-PBIL update (36):

$$\frac{\mathrm{d}\theta^t_i}{\mathrm{d}t} = \int W^f_{\theta^t}(x)(x_i - \theta^t_i) P_{\theta^t}(\mathrm{d}x) =: g_i(\theta^t)\ . \tag{59}$$

Although finding the analytical solution of the differential equation (59) for any initial condition seems a bit intricate, we show that the solution of (59) converges to $(1,\dots,1)$ starting from any initial $\theta \in (0,1]^d$. Note that starting with $\theta_i = 0$ for some i prevents IGO (and PBIL) from sampling any 1 for component i, so that the components of θ that are equal to 0 at startup will always stay so; in that case the IGO flow effectively works as in a smaller-dimensional space.

To prove convergence to $(1,\dots,1)$, we establish the following result:

Lemma 23 *Assume that the selection scheme $w : [0,1] \to \mathbb{R}$ is bounded, that w is a nonincreasing function and that there exists $q_0, q_1 \in (0,1)$ such that $w(q_0) \neq w(q_1)$. On $f(x) = c - \sum_{i=1}^d \alpha_i x_i$, the solution of (59) satisfies $\sum_{i=1}^d \alpha_i \frac{\mathrm{d}\theta^t_i}{\mathrm{d}t} \geqslant 0$; moreover $\sum \alpha_i \frac{\mathrm{d}\theta^t_i}{\mathrm{d}t} = 0$ if and only $\theta \in \{0,1\}^d$.*

Proof We compute $\sum_{i=1}^d \alpha_i g_i(\theta^t)$ and find that

$$\begin{aligned}\sum_{i=1}^d \alpha_i \frac{\mathrm{d}\theta^t_i}{\mathrm{d}t} &= \int W^f_{\theta^t}(x) \left(\sum_{i=1}^d \alpha_i x_i - \sum_{i=1}^d \alpha_i \theta^t_i\right) P_{\theta^t}(\mathrm{d}x) \\ &= \int W^f_{\theta^t}(x)(f(\theta^t) - f(x)) P_{\theta^t}(\mathrm{d}x) \\ &= \mathbb{E}[W^f_{\theta^t}(x)]\,\mathbb{E}[f(x)] - \mathbb{E}[W^f_{\theta^t}(x) f(x)] \end{aligned}$$

where the expectations are taken under P_{θ^t}. Using Lemma 24 below, we have that $\mathbb{E}[W^f_{\theta^t}(x)]\,\mathbb{E}[f(x)] - \mathbb{E}[W^f_{\theta^t}(x) f(x)] \geqslant 0$ and in addition $\mathbb{E}[W^f_{\theta^t}(x)]\,\mathbb{E}[f(x)] - \mathbb{E}[W^f_{\theta^t}(x) f(x)] = 0$ if and only if $\theta^t \in \{0,1\}^d$. ∎

Lemma 24 *Under the assumptions of Lemma 23*

$$\mathbb{E}[-W^f_{\theta^t}(x) f(x)] \geqslant \mathbb{E}[-W^f_{\theta^t}(x)]\mathbb{E}[f(x)] \tag{60}$$

with equality if and only if $\theta^t \in \{0,1\}^d$. Here the expectations are taken under $x \sim P_{\theta^t}$.

Proof We want to prove that $\mathrm{Cov}[-W^f_{\theta^t}(x), f(x)] \geqslant 0$ with equality if and only if $\theta^t \in \{0,1\}^d$. Let us define the random variable $Z = f(x)$ when $x \sim P_{\theta^t}$. Remark that $W^f_{\theta^t}(x) = \mathscr{G}(Z)$ where

$$\mathscr{G}(z) = \frac{1}{F^\leqslant_{\theta^t}(z) - F^<_{\theta^t}(z)} \int_{F^<_{\theta^t}(z)}^{F^\leqslant_{\theta^t}(z)} w(q)dq\ , \tag{61}$$

where $F^\leqslant_{\theta^t}(z) = \Pr_{x'\sim P_{\theta^t}}(f(x') \leqslant z)$ and $F^<_{\theta^t}(z) = \Pr_{x'\sim P_{\theta^t}}(f(x') < z)$. That is $\mathscr{G}$ is the average of w on the interval $(F^<_{\theta^t}(z), F^\leqslant_{\theta^t}(z))$. Since w is nonincreasing, the function $\mathscr{G}$ is also nonincreasing. We have the following equality

$$\mathrm{Cov}[-W^f_{\theta^t}(x), f(x)] = \mathrm{Cov}[-\mathscr{G}(Z), Z]$$

where $-\mathscr{G}$ is nondecreasing. Following (Thorisson, 2000, Chapter 1), we write

$$\mathrm{Cov}[-\mathscr{G}(Z), Z] = \frac{1}{2}\mathrm{Cov}[-\mathscr{G}(Z) + \mathscr{G}(Z'), Z - Z']$$

where Z and Z' are independent following the distribution P_{θ^t}. Given that both the mean of $-\mathscr{G}(Z) + \mathscr{G}(Z')$ and $Z - Z'$ are zero

$$\mathrm{Cov}[-\mathscr{G}(Z), Z] = \frac{1}{2}E[(-\mathscr{G}(Z) + \mathscr{G}(Z'))(Z - Z')] = \frac{1}{2}E[|-\mathscr{G}(Z) + \mathscr{G}(Z')||Z - Z'|] \geqslant 0 \ ,$$

where the last equality holds because $-\mathscr{G}$ is nondecreasing. This proves the main statement of the lemma.

We will now show that if $\theta^t \notin \{0,1\}^d$, then $E[|-\mathscr{G}(Z) + \mathscr{G}(Z')||Z - Z'|] > 0$ and thus consequently $\mathrm{Cov}[-W^f_{\theta^t}(x), f(x)] > 0$. We simply need to show that Z can take with strictly positive probabilities p_1 and p_2 two distinct values z_1 and z_2 such that $\mathscr{G}(z_1)$ and $\mathscr{G}(z_2)$ are distinct. We will then have

$$E[|-\mathscr{G}(Z) + \mathscr{G}(Z')||Z - Z'|] \geqslant (|-\mathscr{G}(z_1) + \mathscr{G}(z_2)||z_1 - z_2|)\, p_1 p_2 > 0 \ .$$

Let us assume that one single θ_i belongs to $(0,1)$ and all the others are either 0 or 1. We can assume without loss of generality that $\theta_1 \in (0,1)$. Then $f(x)$ takes (only) two distinct values with positive probability; let z_1 be the one associated with $x_1 = 1$ and z_2 with $x_1 = 0$. Then $z_1 < z_2$ thanks to the definition of f and because $\alpha_i > 0$. Moreover, unravelling the definitions we find $\mathscr{G}(z_1) = \frac{1}{\theta_1}\int_0^{\theta_1} w(q)dq$ and $\mathscr{G}(z_2) = \frac{1}{1-\theta_1}\int_{\theta_1}^1 w(q)dq$.

The assumption states that there exists $q_0, q_1 \in (0,1)$ such that $w(q_0) \neq w(q_1)$. Assume without loss of generality that $q_0 < q_1$ and $w(q_0) > w(q_1)$. Either $w(q_1) \geqslant w(\theta_1)$ or $w(q_1) < w(\theta_1)$. In the first case, then $\theta_1 > q_0$ and

$$\frac{1}{\theta_1}\int_0^{\theta_1} w(q)dq = \frac{1}{\theta_1}\left(\int_0^{q_0} w(q)dq + \int_{q_0}^{\theta_1} w(q)dq\right) > \frac{1}{\theta_1}\left(w(q_0)q_0 + w(\theta_1)(\theta_1 - q_0)\right) > w(\theta_1) \ ,$$

Meanwhile, $\mathscr{G}(z_2) = \frac{1}{1-\theta_1}\int_{\theta_1}^1 w(q)dq \leqslant w(\theta_1)$ and thus

$$\mathscr{G}(z_1) > \mathscr{G}(z_2) \ .$$

If $w(q_1) < w(\theta_1)$, then $q_1 > \theta_1$ and

$$\begin{aligned}\mathscr{G}(z_2) = \frac{1}{1-\theta_1}\int_{\theta_1}^1 w(q)dq &= \frac{1}{1-\theta_1}\left(\int_{\theta_1}^{q_1} w(q)dq + \int_{q_1}^1 w(q)dq\right) \\ &\leqslant \frac{1}{1-\theta_1}(w(\theta_1)(q_1 - \theta_1) + w(q_1)(1 - q_1)) < w(\theta_1) \quad (62)\end{aligned}$$

and thus $\mathscr{G}(z_2) < w(\theta_1) \leqslant \mathscr{G}(z_1)$ as well.

In the case where we have not only two but l distinct fitnesses that can be sampled with strictly positive probabilities, say $z_1 < \ldots < z_l$, we can show similarly that $\mathscr{G}(z_1) > \mathscr{G}(z_l)$. ■

We are now ready to state the convergence of the trajectories of IGO-PBIL (and cGA) update in the limit of $\delta t \to 0$.

Proposition 25 *Assume that the selection scheme $w : [0,1] \to \mathbb{R}$ is bounded, that w is a nonincreasing function and that there exists $q_0, q_1 \in (0,1)$ such that $w(q_0) \neq w(q_1)$. On the linear functions $f(x) = c - \sum_{i=1}^d \alpha_i x_i$, the critical point $\theta = (1, \ldots, 1)$ of the IGO-PBIL differential equation* (59) *is stable and for any initial condition $\theta \in (0;1]^d$, the continuous-time trajectory solving* (59) *converges to* $(1, \ldots, 1)$.

Proof Remark first that if we start at $\theta^0 \in [\varepsilon, 1]^d$, then the solution of (59) stays in $[\varepsilon, 1]^d$. Indeed, the trajectory cannot go out of $[0,1]^d$ and, in addition, for $\theta \in [\varepsilon, 1)^d$ we have $g_i(\theta) \geqslant 0$ so that the trajectory of (59) cannot go out of $[\varepsilon, 1]^d$. The proof that $g_i(\theta) \geqslant 0$ for $\theta \in [\varepsilon, 1)^d$ is similar to the proof that $\mathrm{Cov}[-W^f_{\theta^t}(x), f(x)] \geqslant 0$ in Lemma 24: namely, we can write $g_i(\theta) = \int \mathbb{E}[W^f_{\theta^t}(x)|x_i](x_i - \theta_i)p_{\theta_i}(x_i)dx_i = \theta_i(1-\theta_i)[h(1) - h(0)]$ where $h(x_i) = \mathbb{E}[W^f_{\theta^t}(x)|x_i] = \mathbb{E}[\mathscr{G}(f(x))|x_i]$ where $\mathscr{G}$ is defined in (61). Then $h(1) \geqslant h(0)$ comes from the increase of $\mathscr{G}$ when going to better function values.

Consider now on $[\varepsilon, 1]^d$, the non-negative function $V(\theta) = \sum_{i=1}^d \alpha_i - \sum_{i=1}^d \alpha_i \theta_i$. Then $V^*(\theta) := \nabla V(\theta) \cdot g(\theta) \leqslant 0$, and $V^*(\theta)$ equals zero only for $\theta \in \{0,1\}^d$ according to Lemma 23. Hence for $\theta \in [\varepsilon, 1]^d$, the function V is a Lyapunov function (Khalil, 1996; Agarwal and O'Regan, 2008), which is minimal at $(1, \ldots, 1)$ and such that $V^*(\theta) < 0$ except at $(1, \ldots, 1)$. Therefore the trajectory of (59) will converge to $(1, \ldots, 1)$ as t goes to infinity (the proof is similar to that of (Khalil, 1996, Theorem 4.1)). Given that this holds for any $\varepsilon > 0$, we can conclude that the trajectory of (59) converges to $(1, \ldots, 1)$ starting from any $\theta \in (0,1]^d$. ■

We now consider on $\mathbb{R}^d$ the family of multivariate normal distributions $P_\theta = \mathcal{N}(m, \sigma^2 I_d)$ with covariance matrix equal to $\sigma^2 I_d$. The parameter θ thus has $d+1$ components $\theta = (m, \sigma) \in \mathbb{R}^d \times \mathbb{R}$. The natural gradient update using this family was derived in Glasmachers et al. (2010); from this we can derive the IGO differential equation which reads:

$$\frac{\mathrm{d}m^t}{\mathrm{d}t} = \int_{\mathbb{R}^d} W^f_{\theta^t}(x)(x - m^t)P_{\mathcal{N}(m^t, (\sigma^t)^2 I_d)}(x)\mathrm{d}x \quad (63)$$

$$\frac{\mathrm{d}\tilde{\sigma}^t}{\mathrm{d}t} = \int_{\mathbb{R}^d} \frac{1}{2d}\left\{\sum_{i=1}^d \left(\frac{x_i - m_i^t}{\sigma^t}\right)^2 - 1\right\} W^f_{\theta^t}(x)P_{\mathcal{N}(m^t, (\sigma^t)^2 I_d)}(x)\mathrm{d}x \quad (64)$$

where σ^t and $\tilde{\sigma}^t$ are linked via $\sigma^t = \exp(\tilde{\sigma}^t)$ or $\tilde{\sigma}^t = \ln(\sigma^t)$. Denoting $\mathcal{N}$ a random vector following a centered multivariate normal distribution with identity covariance matrix we can rewrite the gradient flow as

$$\frac{\mathrm{d}m^t}{\mathrm{d}t} = \sigma^t \mathbb{E}\left[W^f_{\theta^t}(m^t + \sigma^t\mathcal{N})\mathcal{N}\right] \quad (65)$$

$$\frac{\mathrm{d}\tilde{\sigma}^t}{\mathrm{d}t} = \mathbb{E}\left[\frac{1}{2}\left(\frac{\|\mathcal{N}\|^2}{d} - 1\right) W^f_{\theta^t}(m^t + \sigma^t\mathcal{N})\right] \ . \quad (66)$$

Let us analyze the solution of the previous system on linear functions. Without loss of generality (thanks to invariance) we can consider the linear function $f(x) = x_1$. We have

$$W^f_{\theta^t}(x) = w(\Pr(m_1^t + \sigma^t Z_1 < x_1))$$

where Z_1 follows a standard one-dimensional normal distribution and thus

$$W^f_{\theta^t}(m^t + \sigma^t \mathcal{N}) = w(\mathrm{Pr}_{Z_1 \sim \mathcal{N}(0,1)}(Z_1 < \mathcal{N}_1)) \tag{67}$$
$$= w(\mathcal{F}(\mathcal{N}_1)) \tag{68}$$

with $\mathcal{F}$ the cumulative distribution of a standard normal distribution, and $\mathcal{N}_1$ the first component of $\mathcal{N}$. The differential equation thus simplifies into

$$\frac{\mathrm{d}m^t}{\mathrm{d}t} = \sigma^t \begin{pmatrix} \mathbb{E}\left[w(\mathcal{F}(\mathcal{N}_1))\mathcal{N}_1\right] \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{69}$$

$$\frac{\mathrm{d}\tilde{\sigma}^t}{\mathrm{d}t} = \frac{1}{2d}\mathbb{E}\left[(|\mathcal{N}_1|^2 - 1)w(\mathcal{F}(\mathcal{N}_1))\right] \ . \tag{70}$$

Consider, for example, the truncation selection function, i.e. $w(q) = 1_{q \leqslant q_0}$ where $q_0 \in (0, 1)$. (Within the IGO algorithm, this results in so-called *intermediate recombination* of the $q_0 \times N$ best samples.) We find that

$$\frac{\mathrm{d}m^t_1}{\mathrm{d}t} = \sigma^t \mathbb{E}[\mathcal{N}_1 1_{\{\mathcal{N}_1 \leqslant \mathcal{F}^{-1}(q_0)\}}] =: \sigma^t \beta \tag{71}$$

$$\frac{\mathrm{d}\tilde{\sigma}^t}{\mathrm{d}t} = \frac{1}{2d}\left(\int_0^{q_0} \mathcal{F}^{-1}(u)^2 \mathrm{d}u - q_0\right) =: \alpha \ . \tag{72}$$

The solution of the IGO flow for the linear function $f(x) = x_1$ is thus given by

$$m^t_1 = m^0_1 + \frac{\sigma^0 \beta}{\alpha}\exp(\alpha t) \tag{73}$$

$$\sigma^t = \sigma^0 \exp(\alpha t) \ . \tag{74}$$

The coefficient β is negative for any $q_0 < 1$. The coefficient α is positive if and only if $q_0 < 1/2$ by a simple calculus argument[12]; this corresponds to selecting less than half of the sampled points in an ES. In this case the step size σ^t grows exponentially fast to infinity and the mean vector moves along the gradient direction towards minus ∞ at the same rate. If more than half of the points are selected, $q_0 \geqslant 1/2$, the step size will decrease to zero exponentially fast and the mean vector will get stuck (compare also Grahl et al. 2005; Hansen 2006a; Pošík 2008).

For an analysis of the solutions of the system of differential equations (65) and (66) on more complex functions, namely convex-quadratic functions and twice continuously differentiable functions, we refer to Akimoto et al. (2012).

12. Indeed $\alpha = \frac{1}{2d\sqrt{2\pi}} \int_{-\infty}^{\mathcal{F}^{-1}(q_0)} (x^2 - 1)\exp(-x^2/2)\mathrm{d}x = \frac{1}{2d\sqrt{2\pi}} g(\mathcal{F}^{-1}(q_0))$ where $g(y) = \int_{-\infty}^{y} (x^2 - 1)\exp(-x^2/2)\mathrm{d}x$. Using $g(0) = 0$ and $\lim_{y \to \pm\infty} g(y) = 0$, and studying the sign of $g'(y)$, we find that g is positive for $y < 0$ and negative for $y > 0$. Since $\mathcal{F}^{-1}(q_0) < 0$ if and only if $q_0 < 1/2$, we find that $\alpha = \frac{1}{2d\sqrt{2\pi}} g(\mathcal{F}^{-1}(q_0))$ is positive if and only if $q_0 < 1/2$.

Appendix D. Proofs

This final appendix provides longer proofs of propositions and theorems of the paper.

D.1 Proof of Proposition 2

We begin with a calculus lemma which will also be used for the proof of Theorem 10. The proof is omitted and amounts to writing the maximum of a quadratic function obtained by the second-order Taylor expansion of f.

Lemma 26 *Let f be real-valued function on a finite-dimensional vector space E equipped with a positive definite quadratic form $\|\cdot\|^2$. Assume f is smooth and has at most quadratic growth at infinity. Then, for any $x \in E$, we have*

$$\nabla f(x) = \lim_{\varepsilon \to 0} \frac{1}{\varepsilon} \arg\max_h \left\{ f(x + h) - \frac{1}{2\varepsilon}\|h\|^2 \right\}$$

where ∇ is the gradient associated with the norm $\|\cdot\|$. Equivalently,

$$\arg\max_y \left\{ f(y) - \frac{1}{2\varepsilon}\|y - x\|^2 \right\} = x + \varepsilon \nabla f(x) + O(\varepsilon^2)$$

when $\varepsilon \to 0$.

Proposition 2 follows by taking the Fisher information metric at θ^0 for the metric $\|\cdot\|^2$, and using the relation $\mathrm{KL}(P \,\|\, Q) = -\mathrm{Ent}(P) + \log \# X$ if Q is the uniform distribution on a finite space X.

D.2 Proof of Theorem 6 (Convergence of Empirical Means and Quantiles)

Let us give a more precise statement including the necessary regularity conditions.

Proposition 27 *Let $\theta \in \Theta$. Assume that the derivative $\frac{\partial \ln P_\theta(x)}{\partial \theta}$ exists for P_θ-almost all $x \in X$ and that $\mathbb{E}_{P_\theta}\left|\frac{\partial \ln P_\theta(x)}{\partial \theta}\right|^2 < +\infty$. Assume that the function w is non-decreasing and bounded.*

Let $(x_i)_{i \in \mathbb{N}}$ be a sequence of independent samples of P_θ. Then with probability 1, as $N \to \infty$ we have

$$\frac{1}{N}\sum_{i=1}^{N} \widehat{W}^f(x_i) \frac{\partial \ln P_\theta(x_i)}{\partial \theta} \to \int W^f_\theta(x) \frac{\partial \ln P_\theta(x)}{\partial \theta} P_\theta(\mathrm{d}x)$$

where

$$\widehat{W}^f(x_i) = w\left(\frac{\mathrm{rk}_N(x_i) + 1/2}{N}\right)$$

with $\mathrm{rk}_N(x_i) = \#\{1 \leqslant j \leqslant N,\ f(x_j) < f(x_i)\}$. (When there are f-ties in the sample, $\widehat{W}^f(x_i)$ is defined as the average of $w((r + 1/2)/N)$ over the possible rankings r of x_i.)

Proof Let $g : X \to \mathbb{R}$ be any function with $\mathbb{E}_{P_\theta} g^2 < \infty$. We will show that $\frac{1}{N}\sum \widehat{W}^f(x_i)g(x_i) \to \int W_\theta^f(x)g(x)\,P_\theta(\mathrm{d}x)$. Applying this with g equal to the components of $\frac{\partial \ln P_\theta(x)}{\partial\theta}$ will yield the result.

Let us decompose

$$\frac{1}{N}\sum \widehat{W}^f(x_i)g(x_i) = \frac{1}{N}\sum W_\theta^f(x_i)g(x_i) + \frac{1}{N}\sum(\widehat{W}^f(x_i) - W_\theta^f(x_i))g(x_i).$$

Each summand in the first term involves only one sample x_i (contrary to $\widehat{W}^f(x_i)$ which depends on the whole sample). So by the strong law of large numbers, almost surely $\frac{1}{N}\sum W_\theta^f(x_i)g(x_i)$ converges to $\int W_\theta^f(x)g(x)\,P_\theta(\mathrm{d}x)$. So we have to show that the second term converges to 0 almost surely.

By the Cauchy–Schwarz inequality, we have

$$\left|\frac{1}{N}\sum(\widehat{W}^f(x_i) - W_\theta^f(x_i))g(x_i)\right|^2 \leqslant \left(\frac{1}{N}\sum(\widehat{W}^f(x_i) - W_\theta^f(x_i))^2\right)\left(\frac{1}{N}\sum g(x_i)^2\right)$$

By the strong law of large numbers, the second term $\frac{1}{N}\sum g(x_i)^2$ converges to $\mathbb{E}_{P_\theta} g^2$ almost surely. So we have to prove that the first term $\frac{1}{N}\sum(\widehat{W}^f(x_i) - W_\theta^f(x_i))^2$ converges to 0 almost surely.

Since w is bounded by assumption, we can write

$$\begin{aligned}(\widehat{W}^f(x_i) - W_\theta^f(x_i))^2 &\leqslant 2B\left|\widehat{W}^f(x_i) - W_\theta^f(x_i)\right| \\ &= 2B\left|\widehat{W}^f(x_i) - W_\theta^f(x_i)\right|_+ + 2B\left|\widehat{W}^f(x_i) - W_\theta^f(x_i)\right|_-\end{aligned}$$

where B is the bound on $|w|$. We will bound each of these terms.

Let us abbreviate $q_i^< = \Pr_{x'\sim P_\theta}(f(x') < f(x_i))$, $q_i^\leqslant = \Pr_{x'\sim P_\theta}(f(x') \leqslant f(x_i))$, $r_i^< = \#\{j \leqslant N,\, f(x_j) < f(x_i)\}$, $r_i^\leqslant = \#\{j \leqslant N,\, f(x_j) \leqslant f(x_i)\}$.

By definition of $\widehat{W}^f$ we have

$$\widehat{W}^f(x_i) = \frac{1}{r_i^\leqslant - r_i^<}\sum_{k=r_i^<}^{r_i^\leqslant - 1} w((k+1/2)/N)$$

and moreover $W_\theta^f(x_i) = w(q_i^<)$ if $q_i^< = q_i^\leqslant$ or $W_\theta^f(x_i) = \frac{1}{q_i^\leqslant - q_i^<}\int_{q_i^<}^{q_i^\leqslant} w$ otherwise.

The Glivenko–Cantelli theorem (Billingsley, 1995, Theorem 20.6) implies that $\sup_i \left|q_i^\leqslant - r_i^\leqslant/N\right|$ tends to 0 almost surely, and likewise for $\sup_i |q_i^< - r_i^</N|$. So let N be large enough so that these errors are bounded by ε.

Since w is non-increasing, we have $w(q_i^<) \leqslant w(r_i^</N - \varepsilon)$. In the case $q_i^< \neq q_i^\leqslant$, we decompose the interval $[q_i^<; q_i^\leqslant]$ into $(r_i^\leqslant - r_i^<)$ subintervals. The average of w over each such subinterval is compared to a term in the sum defining $w^N(x_i)$: since w is non-increasing, the average of w over the k^{th} subinterval is at most $w((r_i^< + k)/N - \varepsilon)$. So we get

$$W_\theta^f(x_i) \leqslant \frac{1}{r_i^\leqslant - r_i^<}\sum_{k=r_i^<}^{r_i^\leqslant - 1} w(k/N - \varepsilon)$$

so that

$$W_\theta^f(x_i) - \widehat{W}^f(x_i) \leqslant \frac{1}{r_i^\leqslant - r_i^<}\sum_{k=r_i^<}^{r_i^\leqslant - 1}(w(k/N - \varepsilon) - w((k+1/2)/N)).$$

Let us sum over i, remembering that there are $(r_i^\leqslant - r_i^<)$ values of j for which $f(x_j) = f(x_i)$. Taking the positive part, we get

$$\frac{1}{N}\sum_{i=1}^N \left|W_\theta^f(x_i) - \widehat{W}^f(x_i)\right|_+ \leqslant \frac{1}{N}\sum_{k=0}^{N-1}(w(k/N - \varepsilon) - w((k+1/2)/N)).$$

Since w is non-increasing we have

$$\frac{1}{N}\sum_{k=0}^{N-1} w(k/N - \varepsilon) \leqslant \int_{-\varepsilon - 1/N}^{1-\varepsilon-1/N} w$$

and

$$\frac{1}{N}\sum_{k=0}^{N-1} w((k+1/2)/N) \geqslant \int_{1/2N}^{1+1/2N} w$$

(we implicitly extend the range of w so that $w(q) = w(0)$ for $q < 0$ and likewise for $q > 1$). So we have

$$\frac{1}{N}\sum_{i=1}^N \left|W_\theta^f(x_i) - \widehat{W}^f(x_i)\right|_+ \leqslant \int_{-\varepsilon-1/N}^{1/2N} w - \int_{1-\varepsilon-1/N}^{1+1/2N} w \leqslant (2\varepsilon + 3/N)B$$

where B is the bound on $|w|$.

Reasoning symmetrically with $w(k/N + \varepsilon)$ and the inequalities reversed, we get a similar bound for $\frac{1}{N}\sum\left|W_\theta^f(x_i) - \widehat{W}^f(x_i)\right|_-$. This ends the proof. ∎

D.3 Proof of Proposition 7 (Quantile Improvement)

Let us use the weight $w(u) = \mathbb{1}_{u\leqslant q}$. Let m be the value of the q-quantile of f under P_{θ^t}. We want to show that the value of the q-quantile of f under $P_{\theta^{t+\delta t}}$ is less than m, unless the gradient vanishes and the IGO flow is stationary.

Let $p_- = \Pr_{x\sim P_{\theta^t}}(f(x) < m)$, $p_m = \Pr_{x\sim P_{\theta^t}}(f(x) = m)$ and $p_+ = \Pr_{x\sim P_{\theta^t}}(f(x) > m)$. By definition of the quantile value we have $p_- + p_m \geqslant q$ and $p_+ + p_m \geqslant 1 - q$. Let us assume that we are in the more complicated case $p_m \neq 0$ (for the case $p_m = 0$, simply remove the corresponding terms).

We have $W_{\theta_t}^f(x) = 1$ if $f(x) < m$, $W_{\theta_t}^f(x) = 0$ if $f(x) > m$ and $W_{\theta_t}^f(x) = \frac{1}{p_m}\int_{p_-}^{p_- + p_m} w(u)\mathrm{d}u = \frac{q - p_-}{p_m}$ if $f(x) = m$.

Using the same notation as above, let $g_t(\theta) = \int W_{\theta^t}^f(x)\,P_\theta(\mathrm{d}x)$. Decomposing this integral on the three sets $f(x) < m$, $f(x) = m$ and $f(x) > m$, we get that $g_t(\theta) = \Pr_{x\sim P_\theta}(f(x) < m) + \Pr_{x\sim P_\theta}(f(x) = m)\frac{q - p_-}{p_m}$. In particular, $g_t(\theta^t) = q$.

Since we follow a gradient ascent of g_t, for δt small enough we have $g_t(\theta^{t+\delta t}) > g_t(\theta^t)$ unless the gradient vanishes. If the gradient vanishes we have $\theta^{t+\delta t} = \theta^t$ and the quantiles are the same. Otherwise we get $g_t(\theta^{t+\delta t}) > g_t(\theta^t) = q$.

Since $\frac{q-p_-}{p_m} \leqslant \frac{(p_-+p_m)-p_-}{p_m} = 1$, we have $g_t(\theta) \leqslant \Pr_{x\sim P_\theta}(f(x) < m) + \Pr_{x\sim P_\theta}(f(x) = m) = \Pr_{x\sim P_\theta}(f(x) \leqslant m)$.

So $\Pr_{x\sim P_{\theta^{t+\delta t}}}(f(x) \leqslant m) \geqslant g_t(\theta^{t+\delta t}) > q$. This implies, by definition, that the q-quantile value of $P_{\theta^{t+\delta t}}$ is at most m. Moreover, if the objective function has no plateau then $\Pr_{x\sim P_{\theta^{t+\delta t}}}(f(x) = m) = 0$ and so $\Pr_{x\sim P_{\theta^{t+\delta t}}}(f(x) < m) > q$ which implies that the q-quantile of $P_{\theta^{t+\delta t}}$ is stricly less than m.

D.4 Proof of Theorem 10 (Natural Gradient as ML with Infinitesimal Weights)

The proof of Theorem 10 will use Lemma 26. Let W be a function of x, and fix some θ_0 in Θ.

We need some regularity assumptions: we assume that no two points $\theta \in \Theta$ define the same probability distribution and that the map $P_\theta \mapsto \theta$ is continuous. We also assume that the map $\theta \mapsto P_\theta$ is smooth enough, so that $\int \ln P_\theta(x)\, W(x)\, P_{\theta_0}(\mathrm{d}x)$ is a smooth function of θ. (These are restrictions on θ-regularity: this does not mean that W has to be continuous as a function of x.)

The two statements of Theorem 10 using a sum and an integral have similar proofs, so we only include the first. For $\varepsilon > 0$, let θ be the solution of

$$\theta = \arg\max \left\{ (1-\varepsilon) \int \ln P_\theta(x)\, P_{\theta_0}(\mathrm{d}x) + \varepsilon \int \ln P_\theta(x)\, W(x)\, P_{\theta_0}(\mathrm{d}x) \right\}.$$

Then we have

$$\begin{aligned}
\theta &= \arg\max \left\{ \int \ln P_\theta(x)\, P_{\theta_0}(\mathrm{d}x) + \varepsilon \int \ln P_\theta(x)\, (W(x)-1)\, P_{\theta_0}(\mathrm{d}x) \right\} \\
&= \arg\max \left\{ \int \ln P_\theta(x)\, P_{\theta_0}(\mathrm{d}x) - \int \ln P_{\theta_0}(x)\, P_{\theta_0}(\mathrm{d}x) + \varepsilon \int \ln P_\theta(x)\, (W(x)-1)\, P_{\theta_0}(\mathrm{d}x) \right\}
\end{aligned}$$

(because the added term does not depend on θ)

$$\begin{aligned}
&= \arg\max \left\{ -\mathrm{KL}(P_{\theta_0} \,\|\, P_\theta) + \varepsilon \int \ln P_\theta(x)\, (W(x)-1)\, P_{\theta_0}(\mathrm{d}x) \right\} \\
&= \arg\max \left\{ -\frac{1}{\varepsilon}\mathrm{KL}(P_{\theta_0} \,\|\, P_\theta) + \int \ln P_\theta(x)\, (W(x)-1)\, P_{\theta_0}(\mathrm{d}x) \right\}
\end{aligned}$$

When $\varepsilon \to 0$, the first term exceeds the second one if $\mathrm{KL}(P_{\theta_0} \,\|\, P_\theta)$ is too large (because W is bounded), and so $\mathrm{KL}(P_{\theta_0} \,\|\, P_\theta)$ tends to 0. So we can assume that θ is close to θ_0.

When $\theta = \theta_0 + \delta\theta$ is close to θ_0, we have

$$\mathrm{KL}(P_{\theta_0} \,\|\, P_\theta) = \frac{1}{2} \sum I_{ij}(\theta_0)\, \delta\theta_i\, \delta\theta_j + O(\delta\theta^3)$$

with $I_{ij}(\theta_0)$ the Fisher matrix at θ_0. (This actually holds both for $\mathrm{KL}(P_{\theta_0} \,\|\, P_\theta)$ and $\mathrm{KL}(P_\theta \,\|\, P_{\theta_0})$.)

Thus, we can apply Lemma 26 using the Fisher metric $\sum I_{ij}(\theta_0)\, \delta\theta_i\, \delta\theta_j$, and working on a small neighborhood of θ_0 in θ-space (which can be identified with $\mathbb{R}^{\dim\Theta}$). The lemma states that the argmax above is attained at

$$\theta = \theta_0 + \varepsilon \widetilde{\nabla}_\theta \int \ln P_\theta(x)\, (W(x)-1)\, P_{\theta_0}(\mathrm{d}x)$$

up to $O(\varepsilon^2)$, with $\widetilde{\nabla}$ the natural gradient.

Finally, the gradient cancels the constant -1 because $\int (\widetilde{\nabla} \ln P_\theta)\, P_{\theta_0} = 0$ at $\theta = \theta_0$. This proves Theorem 10.

D.5 Proof of Theorem 12 (IGO, CEM and IGO-ML)

Let P_θ be a family of probability distributions of the form

$$P_\theta(\mathrm{d}x) = \frac{1}{Z(\theta)} \exp\left(\sum \theta_i T_i(x)\right)\, H(\mathrm{d}x)$$

where $T_1, \dots, T_k$ is a finite family of functions on X and $H(\mathrm{d}x)$ is some reference measure on X. We assume that the family of functions $(T_i)_i$ together with the constant function $T_0(x) = 1$, are linearly independent. This prevents redundant parametrizations where two values of θ describe the same distribution; this also ensures, by elementary linear algebra, that the Fisher matrix $\mathrm{Cov}(T_i, T_j)$ is invertible.

The IGO update (17) in the parametrization $\bar{T}_i$ is a sum of terms of the form

$$\widetilde{\nabla}_{\bar{T}_i} \ln P(x).$$

So we will compute the natural gradient $\widetilde{\nabla}_{\bar{T}_i}$ in those coordinates.

Let us start with a proposition giving an expression for the Fisher scalar product between two tangent vectors δP and $\delta' P$ of a statistical manifold of exponential distributions. It is one way to express the duality between the coordinates θ_i and $\bar{T}_i$ (compare (Amari and Nagaoka, 2000, (3.30) and Section 3.5)).

Proposition 28 *Let $\delta\theta_i$ and $\delta'\theta_i$ be two small variations of the parameters θ_i. Let $\delta P(x)$ and $\delta' P(x)$ be the resulting variations of the probability distribution P, and $\delta\bar{T}_i$ and $\delta'\bar{T}_i$ the resulting variations of $\bar{T}_i$. Then the scalar product, in Fisher information metric, between the tangent vectors δP and $\delta' P$, is*

$$\langle \delta P, \delta' P \rangle = \sum_i \delta\theta_i\, \delta'\bar{T}_i = \sum_i \delta'\theta_i\, \delta\bar{T}_i.$$

Proof

First, let us prove that $\frac{\partial \bar{T}_i}{\partial \theta_j} = I_{ij}$, the Fisher matrix for θ. Indeed, $\frac{\partial \bar{T}_i}{\partial \theta_j} = \int_x T_i(x) \frac{\partial P(x)}{\partial \theta_j} = \int_x T_i(x) P(x) \frac{\partial \ln P(x)}{\partial \theta_j} = \int_x T_i(x) P(x) (T_j(x) - \bar{T}_j)$ by (19); this is equal to $\mathrm{Cov}(T_i, T_j)$ which is I_{ij} by (20).

Then, by definition of the Fisher metric we have $\langle \delta P, \delta' P\rangle = \sum_{i,j} I_{ij}\, \delta\theta_i\, \delta'\theta_j$ but $\sum_j I_{ij}\delta'\theta_j$ is equal to $\delta'\bar{T}_i$ by the above, because $\delta'\bar{T}_i = \sum_j \frac{\partial \bar{T}_i}{\partial \theta_j}\delta'\theta_j$. Thus we find $\langle \delta P, \delta' P\rangle = \sum_i \delta\theta_i \delta'\bar{T}_i$ as needed. ∎

Proposition 29 *Let f be a function on the statistical manifold of an exponential family as above. Then the components of the natural gradient w.r.t. the expectation parameters are given by the vanilla gradient w.r.t. the natural parameters:*

$$\widetilde{\nabla}_{\bar{T}_i} f = \frac{\partial f}{\partial \theta_i} \tag{75}$$

and conversely

$$\widetilde{\nabla}_{\theta_i} f = \frac{\partial f}{\partial \bar{T}_i} \ . \tag{76}$$

(Beware this does *not* mean that the gradient ascent in any of those parametrizations is the vanilla gradient ascent.)

We could not find a reference for this result, though we think it is known (as a consequence of Amari and Nagaoka 2000, Eq. 3.32).

Proof We saw above that $\frac{\partial \bar{T}}{\partial \theta} = I$. Since $\widetilde{\nabla}_\theta f = I^{-1}\frac{\partial f}{\partial \theta}$, this proves (76) by substituting $\frac{\partial f}{\partial \theta} = \left(\frac{\partial \bar{T}}{\partial \theta}\right)^{\mathsf{T}} \frac{\partial f}{\partial \bar{T}}$.

For the first statement (75) (the one needed for Theorem 12) we have to derive the Fisher matrix for the variables $\bar{T}$. It follows from Proposition 28 that the Fisher matrix in these variables is I^{-1}, by considering the Fisher metric $\sum \delta\theta_i.\delta'\bar{T}_i$ and substituting $I^{-1}\delta\bar{T}$ for $\delta\theta$. Then (75) is proved along the same lines as (76). ∎

Back to the proof of Theorem 12. We can now compute the desired terms:

$$\widetilde{\nabla}_{\bar{T}_i} \ln P(x) = \frac{\partial \ln P(x)}{\partial \theta_i} = T_i(x) - \bar{T}_i$$

by (19). This proves the first statement (34) in Theorem 12 about the form of the IGO update in these parameters.

The other statements follow easily from this together with the additional fact (33) that, for any set of (positive or negative) weights a_i with $\sum a_i = 1$, the value $T^* = \sum_i a_i T(x_i)$ maximizes $\sum_i a_i \ln P(x_i)$.

D.6 Proof of Proposition 20 and Corollary 21 (Speed of IGO)

Lemma 30 *Let X be a centered L^2 random variable with values in $\mathbb{R}^d$ and let A be a real-valued L^2 random variable. Then*

$$\|\mathbb{E}(AX)\| \leqslant \sqrt{\lambda \operatorname{Var} A}$$

where λ is the largest eigenvalue of the covariance matrix of X expressed in an orthonormal basis.

Proof Let v be any vector in $\mathbb{R}^d$; its norm satisfies

$$\|v\| = \sup_{w,\, \|w\|\leqslant 1} (v \cdot w)$$

and in particular

$$\begin{aligned}
\|\mathbb{E}(AX)\| &= \sup_{w,\, \|w\|\leqslant 1} (w \cdot \mathbb{E}(AX)) \\
&= \sup_{w,\, \|w\|\leqslant 1} \mathbb{E}(A\,(w \cdot X)) \\
&= \sup_{w,\, \|w\|\leqslant 1} \mathbb{E}((A - \mathbb{E}A)\,(w \cdot X)) \quad \text{since } (w \cdot X) \text{ is centered} \\
&\leqslant \sup_{w,\, \|w\|\leqslant 1} \sqrt{\operatorname{Var} A}\, \sqrt{\mathbb{E}((w \cdot X)^2)}
\end{aligned}$$

by the Cauchy–Schwarz inequality.

Now, in an orthonormal basis we have

$$(w \cdot X) = \sum_i w_i X_i$$

so that

$$\begin{aligned}
\mathbb{E}((w \cdot X)^2) &= \mathbb{E}\left((\textstyle\sum_i w_i X_i)(\sum_j w_j X_j)\right) \\
&= \textstyle\sum_i \sum_j \mathbb{E}(w_i X_i w_j X_j) \\
&= \textstyle\sum_i \sum_j w_i w_j \mathbb{E}(X_i X_j) \\
&= \textstyle\sum_i \sum_j w_i w_j C_{ij}
\end{aligned}$$

with C_{ij} the covariance matrix of X. The latter expression is the scalar product $(w \cdot Cw)$. Since C is a symmetric positive-semidefinite matrix, $(w \cdot Cw)$ is at most $\lambda\|w\|^2$ with λ the largest eigenvalue of C. ∎

For the IGO flow we have $\frac{\mathrm{d}\theta^t}{\mathrm{d}t} = \mathbb{E}_{x\sim P_\theta} W_\theta^f(x) \tilde{\nabla}_\theta \ln P_\theta(x)$.

So applying the lemma, we get that the norm of $\frac{\mathrm{d}\theta}{\mathrm{d}t}$ is at most $\sqrt{\lambda \operatorname{Var}_{x\sim P_\theta} W_\theta^f(x)}$ where λ is the largest eivengalue of the covariance matrix of $\tilde{\nabla}_\theta \ln P_\theta(x)$ (expressed in a coordinate system where the Fisher matrix at the current point θ is the identity).

By construction of the quantiles, we have $\operatorname{Var}_{x\sim P_\theta} W_\theta^f(x) \leqslant \operatorname{Var}_{[0,1]} w$ (with equality unless there are ties). Indeed, for a given x, let $\mathcal{U}$ be a uniform random variable in $[0,1]$ independent from x and define the random variable $Q = q^<(x) + (q^\leqslant(x) - q^<(x))\mathcal{U}$. Then Q is uniformly distributed between the upper and lower quantiles $q^\leqslant(x)$ and $q^<(x)$ and thus we can rewrite $W_\theta^f(x)$ as $\mathbb{E}(w(Q)|x)$. By the Jensen inequality we have $\operatorname{Var} W_\theta^f(x) = \operatorname{Var} \mathbb{E}(w(Q)|x) \leqslant \operatorname{Var} w(Q)$. In addition when x is taken under P_θ, Q is uniformly distributed in $[0,1]$ and thus $\operatorname{Var} w(Q) = \operatorname{Var}_{[0,1]} w$, i.e., $\operatorname{Var}_{x\sim P_\theta} W_\theta^f(x) \leqslant \operatorname{Var}_{[0,1]} w$.

Besides, consider the tangent space in Θ-space at point θ^t, and let us choose an orthonormal basis in this tangent space for the Fisher metric. Then, in this basis we have $\tilde{\nabla}_i \ln P_\theta(x) =$

$\partial_i \ln P_\theta(x)$. So the covariance matrix of $\tilde{\nabla} \ln P_\theta(x)$ is $\mathbb{E}_{x\sim P_\theta}(\partial_i \ln P_\theta(x) \partial_j \ln P_\theta(x))$, which is equal to the Fisher matrix by definition. So this covariance matrix is the identity, whose largest eigenvalue is 1. This proves Proposition 20.

For Corollary 21, by the relationship (2) between Fisher matrix and Kullback–Leibler divergence, if v is the speed of the IGO flow then the Kullback–Leibler divergence between P_{θ^t} and $P_{\theta^{t+\delta t}}$ (where $P_{\theta^{t+\delta t}}$ is the trajectory of the IGO flow after a time δt) is equal to the square norm of $\delta t.v$ in Fisher metric up to an $O(\|\delta t.v\|^3)$ term. Now if $P_{\theta^{t+\delta t}}$ is obtained by a finite-population IGO *algorithm*, by Theorem 6 the actual v from the IGO algorithm differs from the speed of the IGO flow by an $o(1)_{N\to\infty}$ term. Collecting terms, we find the expression in Corollary 21.

D.7 Proof of Proposition 22 (Noisy IGO)

On the one hand, let P_θ be a family of distributions on X. The IGO algorithm (16) applied to a random function $f(x) = \tilde{f}(x, \omega)$ where ω is a random variable uniformly distributed in $[0, 1]$ reads

$$\theta^{t+\delta t} = \theta^t + \delta t \sum_{i=1}^{N} \hat{w}_i \widetilde{\nabla}_\theta \ln P_\theta(x_i) \tag{77}$$

where $x_i \sim P_\theta$ and $\hat{w}_i$ is according to (14) where ranking is applied to the values $\tilde{f}(x_i, \omega_i)$, with ω_i uniform variables in $[0, 1]$ independent from x_i and from each other.

On the other hand, for the IGO algorithm using $P_\theta \otimes U_{[0,1]}$ and applied to the deterministic function $\tilde{f}$, $\hat{w}_i$ is computed using the ranking according to the $\tilde{f}$ values of the sampled points $\tilde{x}_i = (x_i, \omega_i)$, and thus coincides with the one in (77).

Besides,

$$\partial_\theta \ln P_{\theta\otimes U_{[0,1]}}(\tilde{x}_i) = \partial_\theta \ln P_\theta(x_i) + \underbrace{\partial_\theta \ln U_{[0,1]}(\omega_i)}_{=0}$$

and thus, both the vanilla gradients and the Fisher matrix I (given by the tensor square of the vanilla gradients) coincide. This proves that the IGO algorithm update on space $X \times [0, 1]$, using the family of distributions $\tilde{P}_\theta = P_\theta \otimes U_{[0,1]}$, applied to the deterministic function $\tilde{f}$, coincides with (77).

References

D.H. Ackley, G.E. Hinton, and T.J. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985.

R.P. Agarwal and D. O'Regan. *An Introduction to Ordinary Differential Equations*. Springer, 2008.

Y. Akimoto and Y. Ollivier. Objective improvement in information-geometric optimization. In F. Neumann and K. DeJong, editors, *Foundations of Genetic Algorithms XII (FOGA 2013)*, Adelaide, Australia, 2013.

Y. Akimoto, Y. Nagata, I. Ono, and S. Kobayashi. Bidirectional relation between CMA evolution strategies and natural evolution strategies. In *Proceedings of Parallel Problem Solving from Nature - PPSN XI*, volume 6238 of *Lecture Notes in Computer Science*, pages 154–163. Springer, 2010.

Y. Akimoto, A. Auger, and N. Hansen. Convergence of the continuous time trajectories of isotropic evolution strategies on monotonic $\mathcal{C}^2$-composite functions. In C.A. Coello Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, and M. Pavone, editors, *PPSN (1)*, volume 7491 of *Lecture Notes in Computer Science*, pages 42–51. Springer, 2012. ISBN 978-3-642-32936-4.

S.-I. Amari. Natural gradient works efficiently in learning. *Neural Comput.*, 10:251–276, February 1998. ISSN 0899-7667. doi: 10.1162/089976698300017746. URL `http://portal.acm.org/citation.cfm?id=287476.287477`.

S.-I. Amari and H. Nagaoka. *Methods of information geometry*, volume 191 of *Translations of Mathematical Monographs*. American Mathematical Society, Providence, RI, 2000. ISBN 0-8218-0531-2. Translated from the 1993 Japanese original by Daishi Harada.

D.V. Arnold. Weighted multirecombination evolution strategies. *Theoretical computer science*, 361(1):18–37, 2006.

S. Baluja. Population based incremental learning: A method for integrating genetic search based function optimization and competitve learning. Technical Report CMU-CS-94-163, Carnegie Mellon Report, 1994.

S. Baluja and R. Caruana. Removing the genetics from the standard genetic algorithm. In *Proceedings of ICML'95*, pages 38–46, 1995.

Y. Bengio, P. Lamblin, V. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 153–160. MIT Press, Cambridge, MA, 2007.

Y. Bengio, A.C. Courville, and P. Vincent. Unsupervised feature learning and deep learning: A review and new perspectives. *ArXiv preprints*, arXiv:1206.5538, 2012.

J. Bensadon. Black-box optimization using geodesics in statistical manifolds. *Entropy*, 17 (1):304–345, 2015.

A. Berny. Selection and reinforcement learning for combinatorial optimization. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. Merelo, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature PPSN VI*, volume 1917 of *Lecture Notes in Computer Science*, pages 601–610. Springer Berlin Heidelberg, 2000a.

A. Berny. An adaptive scheme for real function optimization acting as a selection operator. In *Combinations of Evolutionary Computation and Neural Networks, 2000 IEEE Symposium on*, pages 140 –149, 2000b. doi: 10.1109/ECNN.2000.886229.

A. Berny. Boltzmann machine for population-based incremental learning. In *ECAI*, pages 198–202, 2002.

H.-G. Beyer. *The Theory of Evolution Strategies*. Natural Computing Series. Springer-Verlag, 2001.

H.-G. Beyer and H.-P. Schwefel. Evolution strategies—a comprehensive introduction. *Natural computing*, 1(1):3–52, 2002. ISSN 1567-7818.

P. Billingsley. *Probability and measure*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons Inc., New York, third edition, 1995. ISBN 0-471-00710-2. A Wiley-Interscience Publication.

J. Branke, C. Lode, and J.L. Shapiro. Addressing sampling errors and diversity loss in UMDA. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 508–515. ACM, 2007.

T.M. Cover and J.A. Thomas. *Elements of information theory*. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, second edition, 2006. ISBN 978-0-471-24195-9; 0-471-24195-4.

S. Das, S. Maity, B.-Y. Qu, and P.N. Suganthan. Real-parameter evolutionary multimodal optimization - a survey of the state-of-the-art. *Swarm and Evolutionary Computation*, 1 (2):71–88, 2011.

P.-T. de Boer, D.P. Kroese, S. Mannor, and R.Y. Rubinstein. A tutorial on the cross-entropy method. *Annals OR*, 134(1):19–67, 2005.

G. Desjardins, A. Courville, Y. Bengio, P. Vincent, and O. Dellaleau. Parallel tempering for training of restricted Boltzmann machines. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

P. Deuflhard. *Newton methods for nonlinear problems: affine invariance and adaptive algorithms*, volume 35. Springer, 2011.

E.D. Dolan and J.J. Moré. Benchmarking optimization software with performance profiles. *Mathematical programming*, 91(2):201–213, 2002.

S. Bhatnagar E. Zhou. Gradient-based adaptive stochastic search for simulation optimization over continuous space. *ArXiv preprints*, arXiv:1608.00663, 2016.

M. Gallagher and M. Frean. Population-based continuous optimization, probabilistic modelling and mean shift. *Evol. Comput.*, 13(1):29–42, January 2005. ISSN 1063-6560. doi: 10.1162/1063656053583478. URL `http://dx.doi.org/10.1162/1063656053583478`.

Z. Ghahramani. Unsupervised learning. In O. Bousquet, U. von Luxburg, and G. Rätsch, editors, *Advanced Lectures on Machine Learning*, volume 3176 of *Lecture Notes in Computer Science*, pages 72–112. Springer Berlin / Heidelberg, 2004.

T. Glasmachers, T. Schaul, Y. Sun, D. Wierstra, and J. Schmidhuber. Exponential natural evolution strategies. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation GECCO'10*, pages 393–400. ACM, 2010.

J. Grahl, S. Minner, and F. Rothlauf. Behaviour of umda c with truncation selection on monotonous functions. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 3, pages 2553–2559. IEEE, 2005.

N. Hansen. An analysis of mutative σ-self-adaptation on linear fitness functions. *Evolutionary Computation*, 14(3):255–275, 2006a.

N. Hansen. The CMA evolution strategy: a comparing review. In J.A. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, editors, *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, pages 75–102. Springer, 2006b.

N. Hansen. Benchmarking a BI-population CMA-ES on the BBOB-2009 function testbed. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, GECCO '09, pages 2389–2396, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-505-5. doi: http://doi.acm.org/10.1145/1570256.1570333. URL `http://doi.acm.org/10.1145/1570256.1570333`.

N. Hansen and A. Auger. Principled design of continuous stochastic search: From theory to practice. In Y. Borenstein and A. Moraglio, editors, *Theory and Principled Methods for the Design of Metaheuristics*, Natural Computing Series, pages 145–180. Springer Berlin Heidelberg, 2014. ISBN 978-3-642-33205-0. doi: 10.1007/978-3-642-33206-7_8. URL `http://dx.doi.org/10.1007/978-3-642-33206-7_8`.

N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In X. Yao et al., editors, *Parallel Problem Solving from Nature PPSN VIII*, volume 3242 of *LNCS*, pages 282–291. Springer, 2004.

N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *ICEC96*, pages 312–317. IEEE Press, 1996.

N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

N. Hansen, S.D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, 11(1):1–18, 2003. ISSN 1063-6560.

G.R. Harik, F.G. Lobo, and D.E. Goldberg. The compact genetic algorithm. *Evolutionary Computation, IEEE Transactions on*, 3(4):287–297, 1999.

G.E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14:1771–1800, 2002.

G.E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Conputation*, 18:1527–1554, 2006.

R. Hooke and T.A. Jeeves. "Direct search" solution of numerical and statistical problems. *Journal of the ACM*, 8:212–229, 1961.

G.A. Jastrebski and D.V. Arnold. Improving evolution strategies through active covariance matrix adaptation. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 2814–2821. IEEE, 2006. ISBN 0780394879.

M. Jebalia and A. Auger. Log-linear convergence of the scale-invariant $(\mu/\mu_w, \lambda)$-ES and optimal μ for intermediate recombination for large population sizes. In R. Schaefer et al., editor, *Parallel Problem Solving from Nature (PPSN XI)*, volume 6239, pages 52–61. Springer, 2010. URL `http://hal.inria.fr/docs/00/49/44/78/PDF/ppsn2010JebaliaAuger.pdf`.

H. Jeffreys. An invariant form for the prior probability in estimation problems. *Proc. Roy. Soc. London. Ser. A.*, 186:453–461, 1946. ISSN 0962-8444.

H.K. Khalil. *Nonlinear Systems*. Nonlinear Systems. Prentice-Hall, Inc., second edition, 1996.

P.E. Kloeden and E. Platen. *Numerical solution of stochastic differential equations*, volume 23 of *Applications of Mathematics (New York)*. Springer-Verlag, Berlin, 1992. ISBN 3-540-54062-8.

S. Kullback. *Information theory and statistics*. Dover Publications Inc., Mineola, NY, 1997. ISBN 0-486-69684-7. Reprint of the second (1968) edition.

P. Larranaga and J.A. Lozano. *Estimation of distribution algorithms: A new tool for evolutionary computation*. Springer Netherlands, 2002. ISBN 0792374665.

N. Le Roux, P.-A. Manzagol, and Y. Bengio. Topmoumoute online natural gradient algorithm. In *NIPS*, 2007.

L. Malagò, M. Matteucci, and B. Dal Seno. An information geometry perspective on estimation of distribution algorithms: boundary analysis. In *GECCO (Companion)*, pages 2081–2088, 2008.

L. Malagò, M. Matteucci, and G. Pistone. Towards the geometry of estimation of distribution algorithms based on the exponential family. In H.-G. Beyer and W.B. Langdon, editors, *FOGA, Proceedings*, pages 230–242. ACM, 2011. ISBN 978-1-4503-0633-1.

G. Montavon and K.-R. Müller. Deep boltzmann machines and the centering trick. In G. Montavon, G.B. Orr, and K.-R. Müller, editors, *Neural Networks: Tricks of the Trade (2nd ed.)*, volume 7700 of *Lecture Notes in Computer Science*, pages 621–637. Springer, 2012. ISBN 978-3-642-35288-1.

J.J. Moré, B.S. Garbow, and K.E. Hillstrom. Testing unconstrained optimization software. *ACM Transactions on Mathematical Software (TOMS)*, 7(1):17–41, 1981.

J.A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, pages 308–313, 1965.

Y. Ollivier, L. Arnold, A. Auger, and N. Hansen. Information-geometric optimization: A unifying picture via invariance principles. *ArXiv preprints*, arXiv:1106.3708v2, 2011.

M. Pelikan, D.E. Goldberg, and F.G. Lobo. A survey of optimization by building and using probabilistic models. *Computational optimization and applications*, 21(1):5–20, 2002. ISSN 0926-6003.

P. Pošík. Preventing premature convergence in a simple eda via global step size setting. In *Parallel Problem Solving from Nature–PPSN X*, pages 549–558. Springer, 2008.

C.R. Rao. Information and the accuracy attainable in the estimation of statistical parameters. *Bull. Calcutta Math. Soc.*, 37:81–91, 1945. ISSN 0008-0659.

I. Rechenberg. *Evolutionstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog Verlag, Stuttgart, 1973.

I. Rechenberg. *Evolutionsstrategie '94*. Frommann-Holzboog Verlag, 1994.

R. Ros and N. Hansen. A simple modification in CMA-ES achieving linear time and space complexity. In G. Rudolph, T. Jansen, S. Lucas, C. Polini, and N. Beume, editors, *Proceedings of Parallel Problem Solving from Nature (PPSN X)*, volume 5199 of *Lecture Notes in Computer Science*, pages 296–305. Springer, 2008.

R.Y. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 1:127–190, 1999. ISSN 1387-5841. URL `http://dx.doi.org/10.1023/A:1010091220143`. 10.1023/A:1010091220143.

R.Y. Rubinstein and D.P. Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning*. Springer-Verlag New York Inc, 2004. ISBN 038721240X.

R. Salakhutdinov. Learning in Markov random fields using tempered transitions. In Y. Bengio, D. Schuurmans, J. Lafferty, C.K.I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1598–1606. MIT Press, 2009.

R. Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 872–879, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. URL `http://doi.acm.org/10.1145/1390156.1390266`.

B. Sareni and L. Krähenbühl. Fitness sharing and niching methods revisited. *IEEE Trans. Evolutionary Computation*, 2(3):97–106, 1998.

T. Schaul, T. Glasmachers, and J. Schmidhuber. High dimensions and heavy tails for natural evolution strategies. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, GECCO '11, pages 845–852, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0557-0. doi: 10.1145/2001576.2001692. URL `http://doi.acm.org/10.1145/2001576.2001692`.

R.L. Schilling. *Measures, integrals and martingales*. Cambridge University Press, New York, 2005. ISBN 978-0-521-61525-9; 0-521-61525-9. doi: 10.1017/CBO9780511810886. URL `http://dx.doi.org/10.1017/CBO9780511810886`.

L. Schwartz. *Analyse. II*, volume 43 of *Collection Enseignement des Sciences [Collection: The Teaching of Science]*. Hermann, Paris, 1992. Calcul différentiel et équations différentielles, With the collaboration of K. Zizi.

H.-P. Schwefel. *Evolution and Optimum Seeking*. Sixth-generation computer technology series. John Wiley & Sons, Inc. New York, NY, USA, 1995. ISBN 0471571482.

F. Silva and L. Almeida. Acceleration techniques for the backpropagation algorithm. *Neural Networks*, pages 110–119, 1990.

P. Smolensky. Information processing in dynamical systems: foundations of harmony theory. In D. Rumelhart and J. McClelland, editors, *Parallel Distributed Processing*, volume 1, chapter 6, pages 194–281. MIT Press, Cambridge, MA, USA, 1986. ISBN 0-262-68053-X.

Y. Sun, D. Wierstra, T. Schaul, and J. Schmidhuber. Efficient natural evolution strategies. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, GECCO '09, pages 539–546, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-325-9. doi: http://doi.acm.org/10.1145/1569901.1569976. URL `http://doi.acm.org/10.1145/1569901.1569976`.

T. Suttorp, N. Hansen, and C. Igel. Efficient covariance matrix update for variable metric evolution strategies. *Machine Learning*, 75(2):167–197, 2009.

H. Thorisson. *Coupling, Stationarity, and Regeneration*. Springer, 2000.

V. Torczon. On the convergence of pattern search algorithms. *SIAM Journal on optimization*, 7(1):1–25, 1997.

M. Toussaint. Notes on information geometry and evolutionary processes. *ArXiv preprints*, arXiv:nlin/0408040, 2004.

M. Wagner, A. Auger, and M. Schoenauer. EEDA : A new robust estimation of distribution algorithms. Research Report RR-5190, INRIA, 2004. URL `http://hal.inria.fr/inria-00070802/en/`.

D. Whitley. The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In *Proceedings of the third international conference on Genetic algorithms*, pages 116–121, 1989.

D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber. Natural evolution strategies. In *IEEE Congress on Evolutionary Computation*, pages 3381–3387, 2008.

D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber. Natural evolution strategies. *Journal of Machine Learning Research*, 15:949–980, 2014. URL `http://jmlr.org/papers/v15/wierstra14a.html`.

E. Zhou and J. Hu. Gradient-based adaptive stochastic search for non-differentiable optimization. *IEEE Transactions on Automatic Control*, 59(7):1818–1832, July 2014. ISSN 0018-9286. doi: 10.1109/TAC.2014.2310052.

Journal of Machine Learning Research 18 (2017) 1-53 Submitted 12/14; Revised 10/16; Published 4/17

Breaking the Curse of Dimensionality with Convex Neural Networks

Francis Bach FRANCIS.BACH@ENS.FR
INRIA - Sierra Project-team
Département d'Informatique de l'Ecole Normale Supérieure (UMR CNRS/ENS/INRIA)
2, rue Simone Iff
75012 Paris, France

Editor: Yoshua Bengio

Abstract

We consider neural networks with a single hidden layer and non-decreasing positively homogeneous activation functions like the rectified linear units. By letting the number of hidden units grow unbounded and using classical non-Euclidean regularization tools on the output weights, they lead to a convex optimization problem and we provide a detailed theoretical analysis of their generalization performance, with a study of both the approximation and the estimation errors. We show in particular that they are adaptive to unknown underlying linear structures, such as the dependence on the projection of the input variables onto a low-dimensional subspace. Moreover, when using sparsity-inducing norms on the input weights, we show that high-dimensional non-linear variable selection may be achieved, without any strong assumption regarding the data and with a total number of variables potentially exponential in the number of observations. However, solving this convex optimization problem in infinite dimensions is only possible if the non-convex subproblem of addition of a new unit can be solved efficiently. We provide a simple geometric interpretation for our choice of activation functions and describe simple conditions for convex relaxations of the finite-dimensional non-convex subproblem to achieve the same generalization error bounds, even when constant-factor approximations cannot be found. We were not able to find strong enough convex relaxations to obtain provably polynomial-time algorithms and leave open the existence or non-existence of such tractable algorithms with non-exponential sample complexities.

Keywords: Neural networks, non-parametric estimation, convex optimization, convex relaxation.

1. Introduction

Supervised learning methods come in a variety of ways. They are typically based on local averaging methods, such as k-nearest neighbors, decision trees, or random forests, or on optimization of the empirical risk over a certain function class, such as least-squares regression, logistic regression or support vector machine, with positive definite kernels, with model selection, structured sparsity-inducing regularization, or boosting (see, e.g., Györfi and Krzyzak, 2002; Hastie et al., 2009; Shalev-Shwartz and Ben-David, 2014, and references therein).

©2017 Francis Bach.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/14-546.html.

Most methods assume either explicitly or implicitly a certain class of models to learn from. In the non-parametric setting, the learning algorithms may adapt the complexity of the models as the number of observations increases: the sample complexity (i.e., the number of observations) to adapt to any particular problem is typically large. For example, when learning Lipschitz-continuous functions in $\mathbb{R}^d$, at least $n = \Omega(\varepsilon^{-\max\{d,2\}})$ samples are needed to learn a function with excess risk ε (von Luxburg and Bousquet, 2004, Theorem 15). The exponential dependence on the dimension d is often referred to as the *curse of dimensionality*: without any restrictions, exponentially many observations are needed to obtain optimal generalization performances.

At the other end of the spectrum, parametric methods such as linear supervised learning make strong assumptions regarding the problem and generalization bounds based on estimation errors typically assume that the model is well-specified, and the sample complexity to attain an excess risk of ε grows as $n = \Omega(d/\varepsilon^2)$, for linear functions in d dimensions and Lipschitz-continuous loss functions (Shalev-Shwartz and Ben-David, 2014, Chapter 9). While the sample complexity is much lower, when the assumptions are not met, the methods underfit and more complex models would provide better generalization performances.

Between these two extremes, there are a variety of models with structural assumptions that are often used in practice. For input data in $x \in \mathbb{R}^d$, prediction functions $f : \mathbb{R}^d \to \mathbb{R}$ may for example be parameterized as:

(a) *Affine functions*: $f(x) = w^\top x + b$, leading to potential severe underfitting, but easy optimization and good (i.e., non exponential) sample complexity.

(b) *Generalized additive models*: $f(x) = \sum_{j=1}^d f_j(x_j)$, which are generalizations of the above by summing functions $f_j : \mathbb{R} \to \mathbb{R}$ which may not be affine (Hastie and Tibshirani, 1990; Ravikumar et al., 2008; Bach, 2008a). This leads to less strong underfitting but cannot model interactions between variables, while the estimation may be done with similar tools than for affine functions (e.g., convex optimization for convex losses).

(c) *Nonparametric ANOVA models*: $f(x) = \sum_{A \in \mathcal{A}} f_A(x_A)$ for a set $\mathcal{A}$ of subsets of $\{1, \dots, d\}$, and non-linear functions $f_A : \mathbb{R}^A \to \mathbb{R}$. The set $\mathcal{A}$ may be either given (Gu, 2013) or learned from data (Lin and Zhang, 2006; Bach, 2008b). Multi-way interactions are explicitly included but a key algorithmic problem is to explore the $2^d - 1$ non-trivial potential subsets.

(d) *Single hidden-layer neural networks*: $f(x) = \sum_{j=1}^k \sigma(w_j^\top x + b_j)$, where k is the number of units in the hidden layer (see, e.g., Rumelhart et al., 1986; Haykin, 1994). The activation function σ is here assumed to be fixed. While the learning problem may be cast as a (sub)differentiable optimization problem, techniques based on gradient descent may not find the global optimum. If the number of hidden units is fixed, this is a parametric problem.

(e) *Projection pursuit* (Friedman and Stuetzle, 1981): $f(x) = \sum_{j=1}^k f_j(w_j^\top x)$ where k is the number of projections. This model combines both (b) and (d); the only difference with neural networks is that the non-linear functions $f_j : \mathbb{R} \to \mathbb{R}$ are learned from data. The optimization is often done sequentially and is harder than for neural networks.

(e) *Dependence on a unknown k-dimensional subspace*: $f(x) = g(W^\top x)$ with $W \in \mathbb{R}^{d\times k}$, where g is a non-linear function. A variety of algorithms exist for this problem (Li, 1991; Fukumizu et al., 2004; Dalalyan et al., 2008). Note that when the columns of W are assumed to be composed of a single non-zero element, this corresponds to *variable selection* (with at most k selected variables).

In this paper, our main aim is to answer the following question: **Is there a *single* learning method that can deal *efficiently* with all situations above with *provable adaptivity*?** We consider single-hidden-layer neural networks, with non-decreasing homogeneous activation functions such as

$$\sigma(u) = \max\{u, 0\}^\alpha = (u)_+^\alpha,$$

for $\alpha \in \{0, 1, \ldots\}$, with a particular focus on $\alpha = 0$ (with the convention that $0^0 = 0$), that is $\sigma(u) = 1_{u>0}$ (a threshold at zero), and $\alpha = 1$, that is, $\sigma(u) = \max\{u, 0\} = (u)_+$, the so-called *rectified linear unit* (Nair and Hinton, 2010; Krizhevsky et al., 2012). We follow the convexification approach of Bengio et al. (2006); Rosset et al. (2007), who consider potentially infinitely many units and let a sparsity-inducing norm choose the number of units automatically. This leads naturally to incremental algorithms such as forward greedy selection approaches, which have a long history for single-hidden-layer neural networks (see, e.g., Breiman, 1993; Lee et al., 1996).

We make the following contributions:

- We provide in Section 2 a review of functional analysis tools used for learning from continuously infinitely many basis functions, by studying carefully the similarities and differences between L_1- and L_2-penalties on the output weights. For L_2-penalties, this corresponds to a positive definite kernel and may be interpreted through random sampling of hidden weights. We also review incremental algorithms (i.e., forward greedy approaches) to learn from these infinite sets of basis functions when using L_1-penalties.

- The results are specialized in Section 3 to neural networks with a single hidden layer and activation functions which are positively homogeneous (such as the rectified linear unit). In particular, in Sections 3.2, 3.3 and 3.4, we provide simple geometric interpretations to the non-convex problems of additions of new units, in terms of separating hyperplanes or Hausdorff distance between convex sets. They constitute the core potentially hard computational tasks in our framework of learning from continuously many basis functions.

- In Section 4, we provide a detailed theoretical analysis of the approximation properties of (single hidden layer) convex neural networks with monotonic homogeneous activation functions, with explicit bounds. We relate these new results to the extensive literature on approximation properties of neural networks (see, e.g., Pinkus, 1999, and references therein) in Section 4.7, and show that these neural networks are indeed adaptive to linear structures, by replacing the exponential dependence in dimension by an exponential dependence in the dimension of the subspace of the data can be projected to for good predictions.

	Functional form	Generalization bound
No assumption		$n^{-1/(d+3)} \log n$
Affine function	$w^\top x + b$	$d^{1/2} \cdot n^{-1/2}$
Generalized additive model	$\sum_{j=1}^k f_j(w_j^\top x)$, $w_j \in \mathbb{R}^d$	$kd^{1/2} \cdot n^{-1/4} \log n$
Single-layer neural network	$\sum_{j=1}^k \eta_j (w_j^\top x + b_j)_+$	$kd^{1/2} \cdot n^{-1/2}$
Projection pursuit	$\sum_{j=1}^k f_j(w_j^\top x)$, $w_j \in \mathbb{R}^d$	$kd^{1/2} \cdot n^{-1/4} \log n$
Dependence on subspace	$f(W^\top x)$, $W \in \mathbb{R}^{d\times s}$	$d^{1/2} \cdot n^{-1/(s+3)} \log n$

Table 1: Summary of generalization bounds for various models. The bound represents the expected excess risk over the best predictor in the given class. When no assumption is made, the dependence in n goes to zero with an exponent proportional to $1/d$ (which leads to sample complexity exponential in d), while making assumptions removes the dependence of d in the exponent.

- In Section 5, we study the generalization properties under a standard supervised learning set-up, and show that these convex neural networks are adaptive to all situations mentioned earlier. These are summarized in Table 1 and constitute the main statistical results of this paper. When using an ℓ_1-norm on the input weights, we show in Section 5.3 that high-dimensional non-linear variable selection may be achieved, that is, the number of input variables may be much larger than the number of observations, without any strong assumption regarding the data (note that we do not present a polynomial-time algorithm to achieve this).

- We provide in Section 5.5 simple conditions for convex relaxations to achieve the same generalization error bounds, even when constant-factor approximation cannot be found (e.g., because it is NP-hard such as for the threshold activation function and the rectified linear unit). We present in Section 6 convex relaxations based on semi-definite programming, but we were not able to find strong enough convex relaxations (they provide only a provable sample complexity with a polynomial time algorithm which is exponential in the dimension d) and leave open the existence or non-existence of polynomial-time algorithms that preserve the non-exponential sample complexity.

2. Learning from Continuously Infinitely Many Basis Functions

In this section we present the functional analysis framework underpinning the methods presented in this paper, which learn for a potential continuum of features. While the formulation from Sections 2.1 and 2.2 originates from the early work on the approxima-

tion properties of neural networks (Barron, 1993; Kurkova and Sanguineti, 2001; Mhaskar, 2004), the algorithmic parts that we present in Section 2.5 have been studied in a variety of contexts, such as "convex neural networks" (Bengio et al., 2006), or ℓ_1-norm with infinite dimensional feature spaces (Rosset et al., 2007), with links with conditional gradient algorithms (Dunn and Harshbarger, 1978; Jaggi, 2013) and boosting (Rosset et al., 2004).

In the following sections, note that there will be two different notions of *infinity*: infinitely many inputs x and infinitely many basis functions $x \mapsto \varphi_v(x)$. Moreover, two orthogonal notions of *Lipschitz-continuity* will be tackled in this paper: the one of the prediction functions f, and the one of the loss ℓ used to measure the fit of these prediction functions.

2.1 Variation Norm

We consider an arbitrary measurable input space $\mathcal{X}$ (this will a sphere in $\mathbb{R}^{d+1}$ starting from Section 3), with a set of *basis functions* (a.k.a. *neurons* or *units*) $\varphi_v : \mathcal{X} \to \mathbb{R}$, which are parameterized by $v \in \mathcal{V}$, where $\mathcal{V}$ is a compact topological space (typically a sphere for a certain norm on $\mathbb{R}^d$ starting from Section 3). We assume that for any given $x \in \mathcal{X}$, the functions $v \mapsto \varphi_v(x)$ are continuous. These functions will be the hidden neurons in a single-hidden-layer neural network, and thus $\mathcal{V}$ will be $(d+1)$-dimensional for inputs of dimension d (to represent any affine function). Throughout Section 2, these features will be left unspecified as most of the tools apply more generally.

In order to define our space of functions from $\mathcal{X} \to \mathbb{R}$, we need real-valued Radon measures, which are continuous linear forms on the space of continuous functions from $\mathcal{V}$ to $\mathbb{R}$, equipped with the uniform norm (Rudin, 1987; Evans and Gariepy, 1991). For a continuous function $g : \mathcal{V} \to \mathbb{R}$ and a Radon measure μ, we will use the standard notation $\int_{\mathcal{V}} g(v) d\mu(v)$ to denote the action of the measure μ on the continuous function g. The norm of μ is usually referred to as its *total variation* (such finite total variation corresponds to having a continuous linear form on the space of continuous functions), and we denote it as $|\mu|(\mathcal{V})$, and is equal to the supremum of $\int_{\mathcal{V}} g(v) d\mu(v)$ over all continuous functions with values in $[-1, 1]$. As seen below, when μ has a density with respect to a probability measure, this is the L_1-norm of the density.

We consider the space $\mathcal{F}_1$ of functions f that can be written as

$$f(x) = \int_{\mathcal{V}} \varphi_v(x) d\mu(v),$$

where μ is a signed Radon measure on $\mathcal{V}$ with finite total variation $|\mu|(\mathcal{V})$.

When $\mathcal{V}$ is finite, this corresponds to

$$f(x) = \sum_{v \in \mathcal{V}} \mu_v \varphi_v(x),$$

with total variation $\sum_{v \in \mathcal{V}} |\mu_v|$, where the proper formalization for infinite sets $\mathcal{V}$ is done through measure theory.

The infimum of $|\mu|(\mathcal{V})$ over all decompositions of f as $f = \int_{\mathcal{V}} \varphi_v d\mu(v)$, turns out to be a norm γ_1 on $\mathcal{F}_1$, often called the *variation* norm of f with respect to the set of basis functions (see, e.g., Kurkova and Sanguineti, 2001; Mhaskar, 2004).

Given our assumptions regarding the compactness of $\mathcal{V}$, for any $f \in \mathcal{F}_1$, the infimum defining $\gamma_1(f)$ is in fact attained by a signed measure μ, as a consequence of the compactness of measures for the weak topology (see Evans and Gariepy, 1991, Section 1.9).

In the definition above, if we assume that the signed measure μ has a density with respect to a fixed *probability* measure τ with full support on $\mathcal{V}$, that is, $d\mu(v) = p(v) d\tau(v)$, then, the variation norm $\gamma_1(f)$ is also equal to the infimal value of

$$|\mu|(\mathcal{V}) = \int_{\mathcal{V}} |p(v)| d\tau(v),$$

over all integrable functions p such that $f(x) = \int_{\mathcal{V}} p(v) \varphi_v(x) d\tau(v)$. Note however that not all measures have densities, and that the two infimums are the same as all Radon measures are limits of measures with densities. Moreover, the infimum in the definition above is not attained in general (for example when the optimal measure is singular with respect to $d\tau$); however, it often provides a more intuitive definition of the variation norm, and leads to easier comparisons with Hilbert spaces in Section 2.3.

Finite number of neurons. If $f : \mathcal{X} \to \mathbb{R}$ is decomposable into k basis functions, that is, $f(x) = \sum_{j=1}^k \eta_j \varphi_{v_j}(x)$, then this corresponds to $\mu = \sum_{j=1}^k \eta_j \delta(v = v_j)$, and the total variation of μ is equal to the ℓ_1-norm $\|\eta\|_1$ of η. Thus the function f has variation norm less than $\|\eta\|_1$ or equal. This is to be contrasted with the number of basis functions, which is the ℓ_0-pseudo-norm of η.

2.2 Representation from Finitely Many Functions

When minimizing any functional J that depends only on the function values taken at a subset $\hat{\mathcal{X}}$ of values in $\mathcal{X}$, over the ball $\{f \in \mathcal{F}_1, \ \gamma_1(f) \leqslant \delta\}$, then we have a "representer theorem" similar to the reproducing kernel Hilbert space situation, but also with significant differences, which we now present.

The problem is indeed simply equivalent to minimizing a functional on functions restricted to $\mathcal{X}$, that is, to minimizing $J(f_{|\hat{\mathcal{X}}})$ over $f_{|\hat{\mathcal{X}}} \in \mathbb{R}^{\hat{\mathcal{X}}}$, such that $\gamma_{1|\hat{\mathcal{X}}}(f_{|\hat{\mathcal{X}}}) \leqslant \delta$, where

$$\gamma_{1|\hat{\mathcal{X}}}(f_{|\hat{\mathcal{X}}}) = \inf_{\mu} |\mu|(\mathcal{V}) \text{ such that } \forall x \in \hat{\mathcal{X}}, \ f_{|\hat{\mathcal{X}}}(x) = \int_{\mathcal{V}} \varphi_v(x) d\mu(v);$$

we can then build a function defined over all $\mathcal{X}$, through the optimal measure μ above.

Moreover, by Carathéodory's theorem for cones (Rockafellar, 1997), if $\hat{\mathcal{X}}$ is composed of only n elements (e.g., n is the number of observations in machine learning), the optimal function $f_{|\hat{\mathcal{X}}}$ above (and hence f) may be decomposed into at most n functions φ_v, that is, μ is supported by at most n points in $\mathcal{V}$, among a potential continuum of possibilities.

Note however that the identity of these n functions is not known in advance, and thus there is a significant difference with the representer theorem for positive definite kernels and Hilbert spaces (see, e.g., Shawe-Taylor and Cristianini, 2004), where the set of n functions are known from the knowledge of the points $x \in \hat{\mathcal{X}}$ (i.e., kernel functions evaluated at x).

2.3 Corresponding Reproducing Kernel Hilbert Space (RKHS)

We have seen above that if the real-valued measures μ are restricted to have density p with respect to a fixed probability measure τ with full support on $\mathcal{V}$, that is, $d\mu(v) = p(v) d\tau(v)$,

then, the norm $\gamma_1(f)$ is the infimum of the total variation $|\mu|(\mathcal{V}) = \int_{\mathcal{V}} |p(v)| d\tau(v)$, over all decompositions $f(x) = \int_{\mathcal{V}} p(v)\varphi_v(x) d\tau(v)$.

We may also define the infimum of $\int_{\mathcal{V}} |p(v)|^2 d\tau(v)$ over the same decompositions (squared L_2-norm instead of L_1-norm). It turns out that it defines a squared norm γ_2^2 and that the function space $\mathcal{F}_2$ of functions with finite norm happens to be a reproducing kernel Hilbert space (RKHS). When $\mathcal{V}$ is finite, then it is well-known (see, e.g., Berlinet and Thomas-Agnan, 2004, Section 4.1) that the infimum of $\sum_{v \in \mathcal{V}} \mu_v^2$ over all vectors μ such that $f = \sum_{v \in V} \mu_v \varphi_v$ defines a squared RKHS norm with positive definite kernel $k(x,y) = \sum_{v \in V} \varphi_v(x)\varphi_v(y)$.

We show in Appendix A that for any compact set $\mathcal{V}$, we have defined a squared RKHS norm γ_2^2 with positive definite kernel $k(x,y) = \displaystyle\int_{\mathcal{V}} \varphi_v(x)\varphi_v(y) d\tau(v)$.

Random sampling. Note that such kernels are well-adapted to approximations by sampling several basis functions φ_v sampled from the probability measure τ (Neal, 1995; Rahimi and Recht, 2007). Indeed, if we consider m i.i.d. samples $v_1, \dots, v_m$, we may define the approximation $\hat{k}(x,y) = \frac{1}{m} \sum_{i=1}^m \varphi_{v_i}(x)\varphi_{v_i}(y)$, which corresponds to an explicit feature representation. In other words, this corresponds to sampling units v_i, using prediction functions of the form $\frac{1}{m} \sum_{i=1}^m \eta_i \varphi_{v_i}(x)$ and then penalizing by the ℓ_2-norm of η.

When m tends to infinity, then $\hat{k}(x,y)$ tends to $k(x,y)$ and random sampling provides a way to work efficiently with explicit m-dimensional feature spaces. See Rahimi and Recht (2007) for a analysis of the number of units needed for an approximation with error ε, typically of order $1/\varepsilon^2$. See also Bach (2017) for improved results with a better dependence on ε when making extra assumptions on the eigenvalues of the associated covariance operator.

Relationship between $\mathcal{F}_1$ and $\mathcal{F}_2$. The corresponding RKHS norm is always greater than the variation norm (because of Jensen's inequality), and thus the RKHS $\mathcal{F}_2$ is included in $\mathcal{F}_1$. However, as shown in this paper, the two spaces $\mathcal{F}_1$ and $\mathcal{F}_2$ have very different properties; e.g., γ_2 may be computed easily in several cases, while γ_1 does not; also, learning with $\mathcal{F}_2$ may either be done by random sampling of sufficiently many weights or using kernel methods, while $\mathcal{F}_1$ requires dedicated convex optimization algorithms with potentially non-polynomial-time steps (see Section 2.5).

Moreover, for any $v \in \mathcal{V}$, $\varphi_v \in \mathcal{F}_1$ with a norm $\gamma_1(\varphi_v) \leqslant 1$, while in general $\varphi_v \notin \mathcal{F}_2$. This is a simple illustration of the fact that $\mathcal{F}_2$ is too small and thus will lead to a lack of adaptivity that will be further studied in Section 5.4 for neural networks with certain activation functions.

2.4 Supervised Machine Learning

Given some distribution over the pairs $(x,y) \in \mathcal{X} \times \mathcal{Y}$, a loss function $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$, our aim is to find a function $f : \mathcal{X} \to \mathbb{R}$ such that the functional $J(f) = \mathbb{E}[\ell(y, f(x))]$ is small, given some i.i.d. observations (x_i, y_i), $i = 1, \dots, n$. We consider the empirical risk minimization framework over a space of functions $\mathcal{F}$, equipped with a norm γ (in our situation, $\mathcal{F}_1$ and $\mathcal{F}_2$, equipped with γ_1 or γ_2). The empirical risk $\hat{J}(f) = \frac{1}{n} \sum_{i=1}^n \ell(y_i, f(x_i))$, is minimized either (a) by constraining f to be in the ball $\mathcal{F}^\delta = \{f \in \mathcal{F},\ \gamma(f) \leqslant \delta\}$ or (b) regularizing the empirical risk by $\lambda\gamma(f)$. Since this paper has a more theoretical nature, we focus on constraining, noting that in practice, penalizing is often more robust (see, e.g., Harchaoui et al., 2013) and leaving its analysis in terms of learning rates for future work. Since the functional $\hat{J}$ depends only on function values taken at finitely many points, the results from Section 2.2 apply and we expect the solution f to be spanned by only n functions $\varphi_{v_1}, \dots, \varphi_{v_n}$ (but we ignore in advance which ones among all φ_v, $v \in \mathcal{V}$, and the algorithms in Section 2.5 will provide approximate such representations with potentially less or more than n functions).

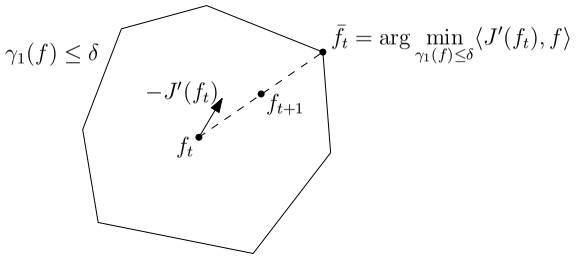


Figure 1: Conditional gradient algorithm for minimizing a smooth functional J on $\mathcal{F}_1^\delta = \{f \in \mathcal{F}_1,\ \gamma_1(f) \leqslant \delta\}$: going from f_t to f_{t+1}; see text for details.

Approximation error vs. estimation error. We consider an ε-approximate minimizer of $\hat{J}(f) = \frac{1}{n} \sum_{i=1}^n \ell(y_i, f(x_i))$ on the convex set $\mathcal{F}^\delta$, that is a certain $\hat{f} \in \mathcal{F}^\delta$ such that $\hat{J}(\hat{f}) \leqslant \varepsilon + \inf_{f \in \mathcal{F}^\delta} \hat{J}(f)$. We thus have, using standard arguments (see, e.g., Shalev-Shwartz and Ben-David, 2014):

$$J(\hat{f}) - \inf_{f \in \mathcal{F}} J(f) \leqslant \left[\inf_{f \in \mathcal{F}^\delta} J(f) - \inf_{f \in \mathcal{F}} J(f) \right] + 2 \sup_{f \in \mathcal{F}^\delta} |\hat{J}(f) - J(f)| + \varepsilon,$$

that is, the excess risk $J(\hat{f}) - \inf_{f \in \mathcal{F}} J(f)$ is upper-bounded by a sum of an *approximation error* $\inf_{f \in \mathcal{F}^\delta} J(f) - \inf_{f \in \mathcal{F}} J(f)$, an *estimation error* $2 \sup_{f \in \mathcal{F}^\delta} |\hat{J}(f) - J(f)|$ and an *optimization error* ε (see also Bottou and Bousquet, 2008). In this paper, we will deal with all three errors, starting from the optimization error which we now consider for the space $\mathcal{F}_1$ and its variation norm.

2.5 Incremental Conditional Gradient Algorithms

In this section, we review algorithms to minimize a smooth functional $J : L_2(d\rho) \to \mathbb{R}$, where ρ is a probability measure on $\mathcal{X}$. This may typically be the expected risk or the empirical risk above. When minimizing $J(f)$ with respect to $f \in \mathcal{F}_1$ such that $\gamma_1(f) \leqslant \delta$, we need algorithms that can efficiently optimize a convex function over an infinite-dimensional space of functions. Conditional gradient algorithms allow to incrementally build a set of elements of $\mathcal{F}_1^\delta = \{f \in \mathcal{F}_1,\ \gamma_1(f) \leqslant \delta\}$; see, e.g., Frank and Wolfe (1956); Dem'yanov and Rubinov (1967); Dudik et al. (2012); Harchaoui et al. (2013); Jaggi (2013); Bach (2015).

Conditional gradient algorithm. We assume the functional J is convex and L-smooth, that is for all $h \in L_2(d\rho)$, there exists a gradient $J'(h) \in L_2(d\rho)$ such that for all $f \in L_2(d\rho)$,

$$0 \leqslant J(f) - J(h) - \langle f - h, J'(h) \rangle_{L_2(d\rho)} \leqslant \frac{L}{2} \|f - h\|^2_{L_2(d\rho)}.$$

When $\mathcal{X}$ is finite, this corresponds to the regular notion of smoothness from convex optimization (Nesterov, 2004).

The conditional gradient algorithm (a.k.a. Frank-Wolfe algorithm) is an iterative algorithm, starting from any function $f_0 \in \mathcal{F}_1^\delta$ and with the following recursion, for $t \geqslant 0$:

$$\begin{aligned} \bar{f}_t &\in \arg\min_{f \in \mathcal{F}_1^\delta} \langle f, J'(f_t) \rangle_{L_2(d\rho)} \\ f_{t+1} &= (1 - \rho_t) f_t + \rho_t \bar{f}_t. \end{aligned}$$

See an illustration in Figure 1. We may choose either $\rho_t = \frac{2}{t+1}$ or perform a line search for $\rho_t \in [0, 1]$. For all of these strategies, the t-th iterate is a convex combination of the functions $\bar{f}_0, \dots, \bar{f}_{t-1}$, and is thus an element of $\mathcal{F}_1^\delta$. It is known that for these two strategies for ρ_t, we have the following convergence rate (see, e.g. Jaggi, 2013):

$$J(f_t) - \inf_{f \in \mathcal{F}_1^\delta} J(f) \leqslant \frac{2L}{t+1} \sup_{f, g \in \mathcal{F}_1^\delta} \|f - g\|^2_{L_2(d\rho)}.$$

When, $r^2 = \sup_{v \in \mathcal{V}} \|\varphi_v\|^2_{L_2(d\rho)}$ is finite, we have $\|f\|^2_{L_2(d\rho)} \leqslant r^2 \gamma_1(f)^2$ and thus we get a convergence rate of $\frac{2Lr^2\delta^2}{t+1}$.

Moreover, the basic Frank-Wolfe (FW) algorithm may be extended to handle the regularized problem as well (Harchaoui et al., 2013; Bach, 2013; Zhang et al., 2012), with similar convergence rates in $O(1/t)$. Also, the second step in the algorithm, where the function f_{t+1} is built in the segment between f_t and the newly found extreme function, may be replaced by the optimization of J over the convex hull of all functions $\bar{f}_0, \dots, \bar{f}_t$, a variant which is often referred to as *fully corrective*. Moreover, in our context where $\mathcal{V}$ is a space where local search techniques may be considered, there is also the possibility of "fine-tuning" the vectors v as well (Bengio et al., 2006), that is, we may optimize the function $(v_1, \dots, v_t, \alpha_1, \dots, \alpha_t) \mapsto J(\sum_{i=1}^t \alpha_i \varphi_{v_i})$, through local search techniques, starting from the weights (α_i) and points (v_i) obtained from the conditional gradient algorithm.

Adding a new basis function. The conditional gradient algorithm presented above relies on solving at each iteration the "Frank-Wolfe step":

$$\max_{\gamma(f) \leqslant \delta} \langle f, g \rangle_{L_2(d\rho)}.$$

for $g = -J'(f_t) \in L_2(d\rho)$. For the norm γ_1 defined through an L_1-norm, we have for $f = \int_{\mathcal{V}} \varphi_v d\mu(v)$ such that $\gamma_1(f) = |\mu|(\mathcal{V})$:

$$\begin{aligned} \langle f, g \rangle_{L_2(d\rho)} &= \int_{\mathcal{X}} f(x) g(x) d\rho(x) = \int_{\mathcal{X}} \left(\int_{\mathcal{V}} \varphi_v(x) d\mu(v) \right) g(x) d\rho(x) \\ &= \int_{\mathcal{V}} \left(\int_{\mathcal{X}} \varphi_v(x) g(x) d\rho(x) \right) d\mu(v) \\ &\leqslant \gamma_1(f) \cdot \max_{v \in \mathcal{V}} \left| \int_{\mathcal{X}} \varphi_v(x) g(x) d\rho(x) \right|, \end{aligned}$$

with equality if and only if $\mu = \mu_+ - \mu_-$ with μ_+ and μ_- two non-negative measures, with μ_+ (resp. μ_-) supported in the set of maximizers v of $|\int_{\mathcal{X}} \varphi_v(x) g(x) d\rho(x)|$ where the value is positive (resp. negative).

This implies that:

$$\max_{\gamma_1(f) \leqslant \delta} \langle f, g \rangle_{L_2(d\rho)} = \delta \max_{v \in \mathcal{V}} \left| \int_{\mathcal{X}} \varphi_v(x) g(x) d\rho(x) \right|, \tag{1}$$

with the maximizers f of the first optimization problem above (left-hand side) obtained as δ times convex combinations of φ_v and $-\varphi_v$ for maximizers v of the second problem (right-hand side).

A common difficulty in practice is the hardness of the Frank-Wolfe step, that is, the optimization problem above over $\mathcal{V}$ may be difficult to solve. See Section 3.2, 3.3 and 3.4 for neural networks, where this optimization is usually difficult.

Finitely many observations. When $\mathcal{X}$ is finite (or when using the result from Section 2.2), the Frank-Wolfe step in Eq. (1) becomes equivalent to, for some vector $g \in \mathbb{R}^n$:

$$\sup_{\gamma_1(f) \leqslant \delta} \frac{1}{n} \sum_{i=1}^n g_i f(x_i) = \delta \max_{v \in \mathcal{V}} \left| \frac{1}{n} \sum_{i=1}^n g_i \varphi_v(x_i) \right|, \tag{2}$$

where the set of solutions of the first problem is in the convex hull of the solutions of the second problem.

Non-smooth loss functions. In this paper, in our theoretical results, we consider non-smooth loss functions for which conditional gradient algorithms do not converge in general. One possibility is to smooth the loss function, as done by Nesterov (2005): an approximation error of ε may be obtained with a smoothness constant proportional to $1/\varepsilon$. By choosing ε as $1/\sqrt{t}$, we obtain a convergence rate of $O(1/\sqrt{t})$ after t iterations. See also Lan (2013).

Approximate oracles. The conditional gradient algorithm may deal with approximate oracles; however, what we need in this paper is not the additive errors situations considered by Jaggi (2013), but multiplicative ones on the computation of the dual norm (similar to ones derived by Bach (2013) for the regularized problem).

Indeed, in our context, we minimize a function $J(f)$ on $f \in L_2(d\rho)$ over a norm ball $\{\gamma_1(f) \leqslant \delta\}$. A multiplicative approximate oracle outputs for any $g \in L_2(d\rho)$, a vector $\hat{f} \in L_2(d\rho)$ such that $\gamma_1(\hat{f}) = 1$, and

$$\langle \hat{f}, g \rangle \leqslant \max_{\gamma_1(f) \leqslant 1} \langle f, g \rangle \leqslant \kappa \langle \hat{f}, g \rangle,$$

for a fixed $\kappa \geqslant 1$. In Appendix B, we propose a modification of the conditional gradient algorithm that converges to a certain $h \in L_2(d\rho)$ such that $\gamma_1(h) \leqslant \delta$ and for which $\inf_{\gamma_1(f) \leqslant \delta} J(f) \leqslant J(h) \leqslant \inf_{\gamma_1(f) \leqslant \delta/\kappa} J(f)$.

Such approximate oracles are not available in general, because they require uniform bounds over all possible values of $g \in L_2(d\rho)$. In Section 5.5, we show that a weaker form of oracle is sufficient to preserve our generalization bounds from Section 5.

Approximation of any function by a finite number of basis functions. The Frank-Wolfe algorithm may be applied in the function space $\mathcal{F}_1$ with $J(f) = \frac{1}{2}\mathbb{E}[(f(x)-g(x))^2]$, we get a function f_t, supported by t basis functions such that $\mathbb{E}[(f_t(x) - g(x))^2] = O(\gamma(g)^2/t)$. Hence, any function in $\mathcal{F}_1$ may be approximated with averaged error ε with $t = O([\gamma(g)/\varepsilon]^2)$ units. Note that the conditional gradient algorithm is one among many ways to obtain such approximation with ε^{-2} units (Barron, 1993; Kurkova and Sanguineti, 2001; Mhaskar, 2004). See Section 4.1 for a (slightly) better dependence on ε for convex neural networks.

3. Neural Networks with Non-decreasing Positively Homogeneous Activation Functions

In this paper, we focus on a specific family of basis functions, that is, of the form

$$x \mapsto \sigma(w^\top x + b),$$

for specific activation functions σ. We assume that σ is non-decreasing and positively homogeneous of some integer degree, i.e., it is equal to $\sigma(u) = (u)_+^\alpha$, for some $\alpha \in \{0, 1, \dots\}$. We focus on these functions for several reasons:

- Since they are not polynomials, linear combinations of these functions can approximate any measurable function (Leshno et al., 1993).
- By homogeneity, they are invariant by a change of scale of the data; indeed, if all observations x are multiplied by a constant, we may simply change the measure μ defining the expansion of f by the appropriate constant to obtain exactly the same function. This allows us to study functions defined on the unit-sphere.
- The special case $\alpha = 1$, often referred to as the *rectified linear unit*, has seen considerable recent empirical success (Nair and Hinton, 2010; Krizhevsky et al., 2012), while the case $\alpha = 0$ (hard thresholds) has some historical importance (Rosenblatt, 1958).

The goal of this section is to specialize the results from Section 2 to this particular case and show that the "Frank-Wolfe" steps have simple geometric interpretations.

We first show that the positive homogeneity of the activation functions allows to transfer the problem to a unit sphere.

Boundedness assumptions. For the theoretical analysis, we assume that our data inputs $x \in \mathbb{R}^d$ are almost surely bounded by R in ℓ_q-norm, for some $q \in [2, \infty]$ (typically $q = 2$ and $q = \infty$). We then build the augmented variable $z \in \mathbb{R}^{d+1}$ as $z = (x^\top, R)^\top \in \mathbb{R}^{d+1}$ by appending the constant R to $x \in \mathbb{R}^d$. We therefore have $\|z\|_q \leqslant \sqrt{2}R$. By defining the vector $v = (w^\top, b/R)^\top \in \mathbb{R}^{d+1}$, we have:

$$\varphi_v(x) = \sigma(w^\top x + b) = \sigma(v^\top z) = (v^\top z)_+^\alpha,$$

which now becomes a function of $z \in \mathbb{R}^{d+1}$.

Without loss of generality (and by homogeneity of σ), we may assume that the ℓ_p-norm of each vector v is equal to $1/R$, that is $\mathcal{V}$ will be the $(1/R)$-sphere for the ℓ_p-norm, where $1/p + 1/q = 1$ (and thus $p \in [1, 2]$, with corresponding typical values $p = 2$ and $p = 1$).

This implies by Hölder's inequality that $\varphi_v(x)^2 \leqslant 2^\alpha$. Moreover this leads to functions in $\mathcal{F}_1$ that are bounded everywhere, that is, $\forall f \in \mathcal{F}_1$, $f(x)^2 \leqslant 2^\alpha \gamma_1(f)^2$. Note that the functions in $\mathcal{F}_1$ are also Lipschitz-continuous for $\alpha \geqslant 1$.

Since all ℓ_p-norms (for $p \in [1, 2]$) are equivalent to each other with constants of at most $\sqrt{d}$ with respect to the ℓ_2-norm, all the spaces $\mathcal{F}_1$ defined above are equal, but the norms γ_1 are of course different and they differ by a constant of at most $d^{\alpha/2}$—this can be seen by computing the dual norms like in Eq. (2) or Eq. (1).

Homogeneous reformulation. In our study of approximation properties, it will be useful to consider the the space of function $\mathcal{G}_1$ defined for z in the unit sphere $\mathbb{S}^d \subset \mathbb{R}^{d+1}$ of the Euclidean norm, such that $g(z) = \int_{\mathbb{S}^d} \sigma(v^\top z) d\mu(v)$, with the norm $\gamma_1(g)$ defined as the infimum of $|\mu|(\mathbb{S}^d)$ over all decompositions of g. Note the slight overloading of notations for γ_1 (for norms in $\mathcal{G}_1$ and $\mathcal{F}_1$) which should not cause any confusion.

In order to prove the approximation properties (with unspecified constants depending only on d), we may assume that $p = 2$, since the norms $\|\cdot\|_p$ for $p \in [1, \infty]$ are equivalent to $\|\cdot\|_2$ with a constant that grows at most as $d^{\alpha/2}$ with respect to the ℓ_2-norm. We thus focus on the ℓ_2-norm in all proofs in Section 4.

We may go from $\mathcal{G}_1$ (a space of real-valued functions defined on the unit ℓ_2-sphere in $d+1$ dimensions) to the space $\mathcal{F}_1$ (a space of real-valued functions defined on the ball of radius R for the ℓ_2-norm) as follows (this corresponds to sending a ball in $\mathbb{R}^d$ into a spherical cap in dimension $d+1$, as illustrated in Figure 2).

- Given $g \in \mathcal{G}_1$, we define $f \in \mathcal{F}_1$, with $f(x) = \left(\frac{\|x\|_2^2}{R^2} + 1\right)^{\alpha/2} g\left(\frac{1}{\sqrt{\|x\|_2^2 + R^2}} \begin{pmatrix} x \\ R \end{pmatrix}\right)$. If g may be represented as $\int_{\mathbb{S}^d} \sigma(v^\top z) d\mu(v)$, then the function f that we have defined may be represented as

$$\begin{aligned} f(x) &= \left(\frac{\|x\|_2^2}{R^2} + 1\right)^{\alpha/2} \int_{\mathbb{S}^d} \left(v^\top \frac{1}{\sqrt{\|x\|_2^2 + R^2}} \begin{pmatrix} x \\ R \end{pmatrix}\right)_+^\alpha d\mu(v) \\ &= \int_{\mathbb{S}^d} \left(v^\top \begin{pmatrix} x/R \\ 1 \end{pmatrix}\right)_+^\alpha d\mu(v) = \int_{\mathbb{S}^d} \sigma(w^\top x + b) d\mu(Rw, b), \end{aligned}$$

  that is $\gamma_1(f) \leqslant \gamma_1(g)$, because we have assumed that $(w^\top, b/R)^\top$ is on the $(1/R)$-sphere.

- Conversely, given $f \in \mathcal{F}_1$, for $z = (t^\top, a)^\top \in \mathbb{S}^d$, we define $g(z) = g(t, a) = f(\frac{Rt}{a}) a^\alpha$, which we define as such on the set of $z = (t^\top, a)^\top \in \mathbb{R}^d \times \mathbb{R}$ (of unit norm) such that $a \geqslant \frac{1}{\sqrt{2}}$. Since we always assume $\|x\|_2 \leqslant R$, we have $\sqrt{\|x\|_2^2 + R^2} \leqslant \sqrt{2}R$, and the value of $g(z, a)$ for $a \geqslant \frac{1}{\sqrt{2}}$ is enough to recover f from the formula above.

  On that portion $\{a \geqslant 1/\sqrt{2}\}$ of the sphere $\mathbb{S}^d$, this function exactly inherits the differentiability properties of f. That is, (a) if f is bounded by 1 and f is $(1/R)$-Lipschitz-continuous, then g is Lipschitz-continuous with a constant that only depends on d and α and (b), if all derivatives of order less than k are bounded by R^{-k}, then all derivatives of the same order of g are bounded by a constant that only depends on d and α. Precise notions of differentiability may be defined on the sphere, using

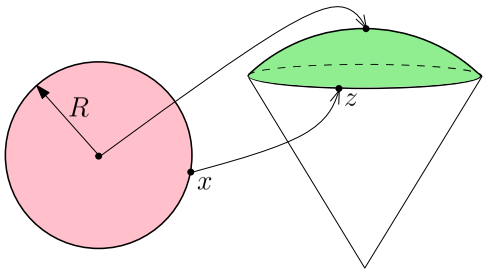


Figure 2: Sending a ball to a spherical cap.

the manifold structure (see, e.g., Absil et al., 2009) or through polar coordinates (see, e.g., Atkinson and Han, 2012, Chapter 3). See these references for more details.

The only remaining important aspect is to define g on the entire sphere, so that (a) its regularity constants are controlled by a constant times the ones on the portion of the sphere where it is already defined, (b) g is either even or odd (this will be important in Section 4). Ensuring that the regularity conditions can be met is classical when extending to the full sphere (see, e.g., Whitney, 1934). Ensuring that the function may be chosen as odd or even may be obtained by multiplying the function g by an infinitely differentiable function which is equal to one for $a \geqslant 1/\sqrt{2}$ and zero for $a \leqslant 0$, and extending by $-g$ or g on the hemi-sphere $a < 0$.

In summary, we may consider in Section 4 functions defined on the sphere, which are much easier to analyze. In the rest of the section, we specialize some of the general concepts reviewed in Section 2 to our neural network setting with specific activation functions, namely, in terms of corresponding kernel functions and geometric reformulations of the Frank-Wolfe steps.

3.1 Corresponding Positive-definite Kernels

In this section, we consider the ℓ_2-norm on the input weight vectors w (that is $p = 2$). We may compute for $x, x' \in \mathbb{R}^d$ the kernels defined in Section 2.3:

$$k_\alpha(x, x') = \mathbb{E}[(w^\top x + b)_+^\alpha (w^\top x' + b)_+^\alpha],$$

for (Rw, b) distributed uniformly on the unit ℓ_2-sphere $\mathbb{S}^d$, and $x, x' \in \mathbb{R}^{d+1}$. Given the angle $\varphi \in [0, \pi]$ defined through $\frac{x^\top x'}{R^2} + 1 = (\cos\varphi)\sqrt{\frac{\|x\|_2^2}{R^2} + 1}\sqrt{\frac{\|x'\|_2^2}{R^2} + 1}$, we have explicit expressions (Le Roux and Bengio, 2007; Cho and Saul, 2009):

$$\begin{aligned} k_0(z, z') &= \frac{1}{2\pi}(\pi - \varphi) \\ k_1(z, z') &= \frac{\sqrt{\frac{\|x\|_2^2}{R^2} + 1}\sqrt{\frac{\|x'\|_2^2}{R^2} + 1}}{2(d+1)\pi}((\pi - \varphi)\cos\varphi + \sin\varphi) \end{aligned}$$

$$k_2(z, z') = \frac{\left(\frac{\|x\|_2^2}{R^2} + 1\right)\left(\frac{\|x'\|_2^2}{R^2} + 1\right)}{2\pi[(d+1)^2 + 2(d+1)]}(3\sin\varphi\cos\varphi + (\pi - \varphi)(1 + 2\cos^2\varphi)).$$

There are key differences and similarities between the RKHS $\mathcal{F}_2$ and our space of functions $\mathcal{F}_1$. The RKHS is smaller than $\mathcal{F}_1$ (i.e., the norm in the RKHS is larger than the norm in $\mathcal{F}_1$); this implies that approximation properties of the RKHS are transferred to $\mathcal{F}_1$. In fact, our proofs rely on this fact.

However, the RKHS norm does not lead to any adaptivity, while the function space $\mathcal{F}_1$ does (see more details in Section 5). This may come as a paradox: both the RKHS $\mathcal{F}_2$ and $\mathcal{F}_1$ have similar properties, but one is adaptive while the other one is not. A key intuitive difference is as follows: given a function f expressed as $f(x) = \int_{\mathcal{V}} \varphi_v(x) p(v) d\tau(v)$, then $\gamma_1(f) = \int_{\mathcal{V}} |p(v)| d\tau(v)$, while the squared RKHS norm is $\gamma_2(f)^2 = \int_{\mathcal{V}} |p(v)|^2 d\tau(v)$. For the L_1-norm, the measure $p(v)d\tau(v)$ may tend to a singular distribution with a bounded norm, while this is not true for the L_2-norm. For example, the function $(w^\top x + b)_+^\alpha$ is in $\mathcal{F}_1$, while it is not in $\mathcal{F}_2$ in general.

3.2 Incremental Optimization Problem for $\alpha = 0$

We consider the problem in Eq. (2) for the special case $\alpha = 0$. For $z_1, \dots, z_n \in \mathbb{R}^{d+1}$ and a vector $y \in \mathbb{R}^n$, the goal is to solve (as well as the corresponding problem with y replaced by $-y$):

$$\max_{v \in \mathbb{R}^{d+1}} \sum_{i=1}^n y_i 1_{v^\top z_i > 0} = \max_{v \in \mathbb{R}^{d+1}} \sum_{i \in I_+} |y_i| 1_{v^\top z_i > 0} - \sum_{i \in I_-} |y_i| 1_{v^\top z_i > 0},$$

where $I_+ = \{i, y_i \geqslant 0\}$ and $I_- = \{i, y_i < 0\}$. As outlined by Bengio et al. (2006), this is equivalent to finding an hyperplane parameterized by v that minimizes a weighted misclassification rate (when doing linear classification). Note that the norm of v has no effect.

NP-hardness. This problem is NP-hard in general. Indeed, if we assume that all y_i are equal to -1 or 1 and with $\sum_{i=1}^n y_i = 0$, then we have a balanced binary classification problem (we need to assume n even). The quantity $\sum_{i=1}^n y_i 1_{v^\top z_i > 0}$ is then $\frac{n}{2}(1 - 2e)$ where e is the corresponding classification error for a problem of classifying at positive (resp. negative) the examples in I_+ (resp. I_-) by thresholding the linear classifier $v^\top z$. Guruswami and Raghavendra (2009) showed that for all (ε, δ), it is NP-hard to distinguish between instances (i.e., configurations of points x_i), where a halfspace with classification error at most ε exists, and instances where all half-spaces have an error of at least $1/2 - \delta$. Thus, it is NP-hard to distinguish between instances where there exists $v \in \mathbb{R}^{d+1}$ such that $\sum_{i=1}^n y_i 1_{v^\top z_i > 0} \geqslant \frac{n}{2}(1 - 2\varepsilon)$ and instances where for all $v \in \mathbb{R}^{d+1}$, $\sum_{i=1}^n y_i 1_{v^\top z_i > 0} \leqslant n\delta$. Thus, it is NP-hard to distinguish instances where $\max_{v \in \mathbb{R}^{d+1}} \sum_{i=1}^n y_i 1_{v^\top z_i > 0} \geqslant \frac{n}{2}(1 - 2\varepsilon)$ and ones where it is less than $\frac{n}{2}\delta$. Since this is valid for all δ and ε, this rules out a constant-factor approximation.

Convex relaxation. Given linear binary classification problems, there are several algorithms to approximately find a good half-space. These are based on using convex surrogates (such as the hinge loss or the logistic loss). Although some theoretical results do exist regarding the classification performance of estimators obtained from convex surrogates (Bartlett et al., 2006), they do not apply in the context of linear classification.

3.3 Incremental Optimization Problem for $\alpha = 1$

We consider the problem in Eq. (2) for the special case $\alpha = 1$. For $z_1, \dots, z_n \in \mathbb{R}^{d+1}$ and a vector $y \in \mathbb{R}^n$, the goal is to solve (as well as the corresponding problem with y replaced by $-y$):

$$\max_{\|v\|_p \leqslant 1} \sum_{i=1}^n y_i (v^\top z_i)_+ = \max_{\|v\|_p \leqslant 1} \sum_{i \in I_+} (v^\top |y_i| z_i)_+ - \sum_{i \in I_-} (v^\top |y_i| z_i)_+,$$

where $I_+ = \{i, y_i \geqslant 0\}$ and $I_- = \{i, y_i < 0\}$. We have, with $t_i = |y_i| z_i \in \mathbb{R}^{d+1}$, using convex duality:

$$\begin{aligned}
\max_{\|v\|_p \leqslant 1} \sum_{i=1}^n y_i (v^\top z_i)_+ &= \max_{\|v\|_p \leqslant 1} \sum_{i \in I_+} (v^\top t_i)_+ - \sum_{i \in I_-} (v^\top t_i)_+ \\
&= \max_{\|v\|_p \leqslant 1} \sum_{i \in I_+} \max_{b_i \in [0,1]} b_i v^\top t_i - \sum_{i \in I_-} \max_{b_i \in [0,1]} b_i v^\top t_i \\
&= \max_{b_+ \in [0,1]^{I_+}} \max_{\|v\|_p \leqslant 1} \min_{b_- \in [0,1]^{I_-}} v^\top [T_+^\top b_+ - T_-^\top b_-] \\
&= \max_{b_+ \in [0,1]^{I_+}} \min_{b_- \in [0,1]^{I_-}} \max_{\|v\|_p \leqslant 1} v^\top [T_+^\top b_+ - T_-^\top b_-] \text{ by Fenchel duality,} \\
&= \max_{b_+ \in [0,1]^{I_+}} \min_{b_- \in [0,1]^{I_-}} \|T_+^\top b_+ - T_-^\top b_-\|_q,
\end{aligned}$$

where $T_+ \in \mathbb{R}^{n_+ \times d}$ has rows t_i, $i \in I_+$ and $T_- \in \mathbb{R}^{n_- \times d}$ has rows t_i, $i \in I_-$, with $v \in \arg\max_{\|v\|_p \leqslant 1} v^\top (T_+^\top b_+ - T_-^\top b_-)$. The problem thus becomes

$$\max_{b_+ \in [0,1]^{n_+}} \min_{b_- \in [0,1]^{n_-}} \|T_+^\top b_+ - T_-^\top b_-\|_q.$$

For the problem of maximizing $|\sum_{i=1}^n y_i (v^\top z_i)_+|$, then this corresponds to

$$\max \left\{ \max_{b_+ \in [0,1]^{n_+}} \min_{b_- \in [0,1]^{n_-}} \|T_+^\top b_+ - T_-^\top b_-\|_q, \max_{b_- \in [0,1]^{n_-}} \min_{b_+ \in [0,1]^{n_+}} \|T_+^\top b_+ - T_-^\top b_-\|_q \right\}.$$

This is exactly the Hausdorff distance between the two convex sets $\{T_+^\top b_+,\ b_+ \in [0,1]^{n_+}\}$ and $\{T_-^\top b_-,\ b_- \in [0,1]^{n_-}\}$ (referred to as zonotopes, see below).

Given the pair (b_+, b_-) achieving the Hausdorff distance, then we may compute the optimal v as $v = \arg\max_{\|v\|_p \leqslant 1} v^\top (T_+^\top b_+ - T_-^\top b_-)$. Note this has not changed the problem at all, since it is equivalent. It is still NP-hard in general (König, 2014). But we now have a geometric interpretation with potential approximation algorithms. See below and Section 6.

Zonotopes. A *zonotope* A is the Minkowski sum of a finite number of segments from the origin, that is, of the form

$$A = [0, t_1] + \cdots + [0, t_r] = \left\{ \sum_{i=1}^r b_i t_i,\ b \in [0,1]^r \right\},$$

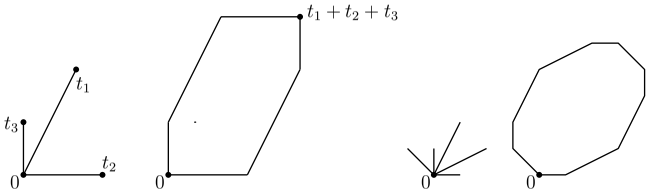


Figure 3: Two zonotopes in two dimensions: (left) vectors, and (right) their Minkowski sum (represented as a polygone).

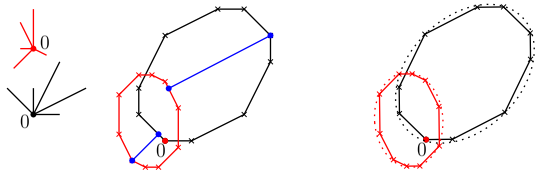


Figure 4: Left: two zonotopes (with their generating segments) and the segments achieving the two sides of the Haussdorf distance. Right: approximation by ellipsoids.

for some vectors t_i, $i = 1, \dots, r$ (Bolker, 1969). See an illustration in Figure 3. They appear in several areas of computer science (Edelsbrunner, 1987; Guibas et al., 2003) and mathematics (Bolker, 1969; Bourgain et al., 1989). In machine learning, they appear naturally as the affine projection of a hypercube; in particular, when using a higher-dimensional distributed representation of points in $\mathbb{R}^d$ with elements in $[0,1]^r$, where r is larger than d (see, e.g., Hinton and Ghahramani, 1997), the underlying polytope that is modelled in $\mathbb{R}^d$ happens to be a zonotope.

In our context, the two convex sets $\{T_+^\top b_+,\ b_+ \in [0,1]^{n_+}\}$ and $\{T_-^\top b_-,\ b_- \in [0,1]^{n_-}\}$ defined above are thus zonotopes. See an illustration of the Hausdorff distance computation in Figure 4 (middle plot), which is the core computational problem for $\alpha = 1$.

Approximation by ellipsoids. Centrally symmetric convex polytopes (w.l.o.g. centered around zero) may be approximated by ellipsoids. In our set-up, we could use the minimum volume enclosing ellipsoid (see, e.g. Barvinok, 2002), which can be computed exactly when the polytope is given through its vertices, or up to a constant factor when the polytope is such that quadratic functions may be optimized with a constant factor approximation. For zonotopes, the standard semi-definite relaxation of Nesterov (1998) leads to such constant-factor approximations, and thus the minimum volume inscribed ellipsoid may be computed up to a constant. Given standard results (see, e.g. Barvinok, 2002), a $(1/\sqrt{d})$-scaled version of the ellipsoid is inscribed in this polytope, and thus the ellipsoid is a provably good approximation of the zonotope with a factor scaling as $\sqrt{d}$. However, the approximation

ratio is not good enough to get any relevant bound for our purpose (see Section 5.5), as for computing the Haussdorff distance, we care about potentially vanishing differences that are swamped by constant factor approximations.

Nevertheless, the ellipsoid approximation may prove useful in practice, in particular because the ℓ_2-Haussdorff distance between two ellipsoids may be computed in polynomial time (see Appendix E).

NP-hardness. Given the reduction of the case $\alpha = 1$ (rectified linear units) to $\alpha = 0$ (exact thresholds) (Livni et al., 2014), the incremental problem is also NP-hard, so as obtaining a constant-factor approximation. However, this does not rule out convex relaxations with non-constant approximation ratios (see Section 6 for more details).

3.4 Incremental Optimization Problem for $\alpha \geqslant 2$

We consider the problem in Eq. (2) for the remaining cases $\alpha \geqslant 2$. For $z_1, \dots, z_n \in \mathbb{R}^{d+1}$ and a vector $y \in \mathbb{R}^n$, the goal is to solve (as well as the corresponding problem with y replaced by $-y$):

$$\max_{\|v\|_p \leqslant 1} \frac{1}{\alpha} \sum_{i=1}^n y_i (v^\top z_i)_+^\alpha = \max_{\|v\|_p \leqslant 1} \sum_{i \in I_+} \frac{1}{\alpha} (v^\top |y_i|^{1/\alpha} z_i)_+^\alpha - \sum_{i \in I_-} \frac{1}{\alpha} (v^\top |y_i|^{1/\alpha} z_i)_+^\alpha,$$

where $I_+ = \{i, y_i \geqslant 0\}$ and $I_- = \{i, y_i < 0\}$. We have, with $t_i = |y_i|^{1/\alpha} z_i \in \mathbb{R}^{d+1}$, and $\beta \in (1, 2]$ defined by $1/\beta + 1/\alpha = 1$ (we use the fact that the function $u \mapsto u^\alpha/\alpha$ and $v \mapsto v^\beta/\beta$ are Fenchel-dual to each other):

$$\begin{aligned}
\max_{\|v\|_p \leqslant 1} \frac{1}{\alpha} \sum_{i=1}^n y_i (v^\top z_i)_+^\alpha &= \max_{\|v\|_p \leqslant 1} \sum_{i \in I_+} \frac{1}{\alpha} (v^\top t_i)_+^\alpha - \sum_{i \in I_-} \frac{1}{\alpha} (v^\top t_i)_+^\alpha \\
&= \max_{\|v\|_p \leqslant 1} \sum_{i \in I_+} \max_{b_i \geqslant 0} \Big\{ b_i v_i^\top t_i - \frac{1}{\beta} b_i^\beta \Big\} - \sum_{i \in I_-} \max_{b_i \geqslant 0} \Big\{ b_i v^\top t_i - \frac{1}{\beta} b_i^\beta \Big\} \\
&= \max_{b_+ \in \mathbb{R}_+^{I_+}} \min_{b_- \in \mathbb{R}_+^{I_-}} \max_{\|v\|_p \leqslant 1} v^\top [T_+^\top b_+ - T_-^\top b_-] - \frac{1}{\beta} \|b_+\|_\beta^\beta + \frac{1}{\beta} \|b_-\|_\beta^\beta \\
&\qquad \text{by Fenchel duality,} \\
&= \max_{b_+ \in [0,1]^{I_+}} \min_{b_- \in [0,1]^{I_-}} \|T_+^\top b_+ - T_-^\top b_-\|_q - \frac{1}{\beta} \|b_+\|_\beta^\beta + \frac{1}{\beta} \|b_-\|_\beta^\beta, \quad (3)
\end{aligned}$$

where $T_+ \in \mathbb{R}^{n_+ \times d}$ has rows t_i, $i \in I_+$ and $T_- \in \mathbb{R}^{n_- \times d}$ has rows t_i, $i \in I_-$, with $v \in \arg\max_{\|v\|_p \leqslant 1} (T_+^\top b_+ - T_-^\top b_-)^\top v$. Contrary to the case $\alpha = 1$, we do not obtain exactly a formulation as a Hausdorff distance. However, if we consider the convex sets $K_\lambda^+ = \{T_+^\top b_+,\ b_+ \geqslant 0,\ \|b_+\|_\beta \leqslant \lambda\}$ and $K_\mu^- = \{T_-^\top b_-,\ b_- \geqslant 0,\ \|b_-\|_\beta \leqslant \mu\}$, then, a solution of Eq. (3) may be obtained from Hausdorff distance computations between K_λ^+ and K_μ^-, for certain λ and μ.

Note that, while for $\alpha = 1$ we can use the identity $2u_+ = u + |u|$ to replace the rectified linear unit by the absolute value and obtain the same function space, this is not possible for $\alpha = 2$, as $(u_+)^2$ and u^2 do not differ by a linear function. This implies that the results from Livni et al. (2014), which state that for the quadratic activation function, the incremental problems is equivalent to an eigendecomposition (and hence solvable in polynomial time), do not apply.

4. Approximation Properties

In this section, we consider the approximation properties of the set $\mathcal{F}_1$ of functions defined on $\mathbb{R}^d$. As mentioned earlier, the norm used to penalize input weights w or v is irrelevant for approximation properties as all norms are equivalent. Therefore, we focus on the case $q = p = 2$ and ℓ_2-norm constraints.

Because we consider homogeneous activation functions, we start by studying the set $\mathcal{G}_1$ of functions defined on the unit ℓ_2-sphere $\mathbb{S}^d \subset \mathbb{R}^{d+1}$. We denote by τ_d the uniform probability measure on $\mathbb{S}^d$. The set $\mathcal{G}_1$ is defined as the set of functions on the sphere such that $g(z) = \int_{\mathbb{S}^d} \sigma(v^\top z) p(z) d\tau_d(z)$, with the norm $\gamma_1(g)$ equal to the smallest possible value of $\int_{\mathbb{S}^d} |p(z)| d\tau_d(z)$. We may also define the corresponding squared RKHS norm by the smallest possible value of $\int_{\mathbb{S}^d} |p(z)|^2 d\tau_d(z)$, with the corresponding RKHS $\mathcal{G}_2$.

In this section, we first consider approximation properties of functions in $\mathcal{G}_1$ by a finite number of neurons (only for $\alpha = 1$). We then study approximation properties of functions on the sphere by functions in $\mathcal{G}_1$. It turns out that all our results are based on the approximation properties of the corresponding RKHS $\mathcal{G}_2$: we give sufficient conditions for being in $\mathcal{G}_2$, and then approximation bounds for functions which are not in $\mathcal{G}_2$. Finally we transfer these to the spaces $\mathcal{F}_1$ and $\mathcal{F}_2$, and consider in particular functions which only depend on projections on a low-dimensional subspace, for which the properties of $\mathcal{G}_1$ and $\mathcal{G}_2$ (and of $\mathcal{F}_1$ and $\mathcal{F}_2$) differ. This property is key to obtaining generalization bounds that show adaptivity to linear structures in the prediction functions (as done in Section 5).

Approximation properties of neural networks with finitely many neurons have been studied extensively (see, e.g., Petrushev, 1998; Pinkus, 1999; Makovoz, 1998; Burger and Neubauer, 2001). In Section 4.7, we relate our new results to existing work from the literature on approximation theory, by showing that our results provide an explicit control of the various weight vectors which are needed for bounding the estimation error in Section 5.

4.1 Approximation by a Finite Number of Basis Functions

A key quantity that drives the approximability by a finite number of neurons is the variation norm $\gamma_1(g)$. As shown in Section 2.5, any function g such that $\gamma_1(g)$ is finite, may be approximated in $L_2(\mathbb{S}^d)$-norm with error ε with $n = O(\gamma_1(g)^2 \varepsilon^{-2})$ units. For $\alpha = 1$ (rectified linear units), we may improve the dependence in ε, through the link with zonoids and zonotopes, as we now present.

If we decompose the signed measure μ as $\mu = \mu_+ - \mu_-$ where μ_+ and μ_- are positive measures, then, for $g \in \mathcal{G}_1$, we have $g(z) = \int_{\mathbb{S}^d} (v^\top z)_+ d\mu_+(v) - \int_{\mathbb{S}^d} (v^\top z)_+ d\mu_-(v) = g_+(z) - g_-(z)$, which is a decomposition of g as a difference of positively homogenous convex functions.

Positively homogenous convex functions h may be written as the *support function* of a compact convex set K (Rockafellar, 1997), that is, $h(z) = \max_{y \in K} y^\top z$, and the set K characterizes the function h. The functions g_+ and g_- defined above are not *any* convex positively homogeneous functions, as we now describe.

If the measure μ_+ is supported by finitely many points, that is, $\mu_+(v) = \sum_{i=1}^r \eta_i \delta(v - v_i)$ with $\eta \geqslant 0$, then $g_+(z) = \sum_{i=1}^t \eta_i (v_i^\top z)_+ = \sum_{i=1}^t (\eta_i v_i^\top z)_+ = \sum_{i=1}^t (t_i^\top z)_+$ for $t_i = \eta_i v_i$. Thus the corresponding set K_+ is the *zonotope* $[0, t_1] + \cdots + [0, t_r] = \{ \sum_{i=1}^r b_i t_i, \ b \in [0,1]^r \}$ already defined in Section 3.3. Thus the functions $g_+ \in \mathcal{G}_1$ and $g_- \in \mathcal{G}_1$ for finitely supported measures μ are support functions of zonotopes.

When the measure μ is not constrained to have finite support, then the sets K_+ and K_- are limits of zonotopes, and thus, by definition, *zonoids* (Bolker, 1969), and thus functions in $\mathcal{G}_1$ are differences of support functions of zonoids. Zonoids are a well-studied set of convex bodies. They are centrally symmetric, and in two dimensions, all centrally symmetric compact convexs sets are (up to translation) zonoids, which is not true in higher dimensions (Bolker, 1969). Moreover, the problem of approximating a zonoid by a zonotope with a small number of segments (Bourgain et al., 1989; Matoušek, 1996) is essentially equivalent to the approximation of a function g by finitely many neurons. The number of neurons directly depends on the norm γ_1, as we now show.

Proposition 1 (Number of units - $\alpha = 1$) *Let $\varepsilon \in (0, 1/2)$. For any function g in $\mathcal{G}_1$, there exists a measure μ supported on at most r points in $\mathcal{V}$, so that for all $z \in \mathbb{S}^d$. $|g(z) - \int_{\mathbb{S}^d} (v^\top z)_+ d\mu(v)| \leqslant \varepsilon \gamma_1(g)$, with $r \leqslant C(d) \varepsilon^{-2d/(d+3)}$, for some constant $C(d)$ that depends only on d.*

Proof Without loss of generality, we assume $\gamma(g) = 1$. It is shown by Matoušek (1996) that for any probability measure μ (positive and with finite mass) on the sphere $\mathbb{S}^d$, there exists a set of r points $v_1, \dots, v_r$, so that for all $z \in \mathbb{S}^d$,

$$\left| \int_{\mathbb{S}^d} |v^\top z| d\mu(v) - \frac{1}{r} \sum_{i=1}^r |v_i^\top z| \right| \leqslant \varepsilon, \tag{4}$$

with $r \leqslant C(d) \varepsilon^{-2+6/(d+3)} = C(d) \varepsilon^{-2d/(d+3)}$, for some constant $C(d)$ that depends only on d. We may then simply write

$$g(z) = \int_{\mathbb{S}^d} (v^\top z)_+ d\mu(v) = \frac{1}{2} \int_{\mathbb{S}^d} (v^\top z) d\mu(v) + \frac{\mu_+(\mathbb{S}^d)}{2} \int_{\mathbb{S}^d} |v^\top z| \frac{d\mu_+(v)}{\mu_+(\mathbb{S}^d)} - \frac{\mu_-(\mathbb{S}^d)}{2} \int_{\mathbb{S}^d} |v^\top z| \frac{d\mu_-(v)}{\mu_-(\mathbb{S}^d)},$$

and approximate the last two terms with error $\varepsilon \mu_\pm(\mathbb{S}^d)$ with r terms, leading to an approximation of $\varepsilon \mu_+(\mathbb{S}^d) + \varepsilon \mu_-(\mathbb{S}^d) = \varepsilon \gamma_1(g) = \varepsilon$, with a remainder that is a linear function $q^\top z$ of z, with $\|q\|_2 \leqslant 1$. We may then simply add two extra units with vectors $q/\|q\|_2$ and weights $-\|q\|_2$ and $\|q\|_2$. We thus obtain, with $2r + 2$ units, the desired approximation result.

Note that Bourgain et al. (1989, Theorem 6.5) showed that the scaling in ε in Eq. (4) is not improvable, if the measure is allowed to have non equal weights on all points and the proof relies on the non-approximability of the Euclidean ball by centered zonotopes. This results does not apply here, because we may have different weights $\mu_-(\mathbb{S}^d)$ and $\mu_+(\mathbb{S}^d)$. ■

Note that the proposition above is slightly improved in terms of the scaling of the number of neurons with respect to the approximation error ε (improved exponent), compared to conditional gradient bounds (Barron, 1993; Kurkova and Sanguineti, 2001). Indeed, the simple use of conditional gradient leads to $r \leqslant \varepsilon^{-2} \gamma_1(g)^2$, with a better constant (independent of d) but a worse scaling in ε—also with a result in $L_2(\mathbb{S}^d)$-norm and not uniformly on the ball $\{\|x\|_q \leqslant R\}$. Note also that the conditional gradient algorithm gives a constructive way of building the measure. Moreover, the proposition above is related to the result from Makovoz (1998, Theorem 2), which applies for $\alpha = 0$ but with a number of neurons growing as $\varepsilon^{-2d/(d+1)}$, or to the one of Burger and Neubauer (2001, Example 3.1), which applies to a piecewise affine sigmoidal function but with a number of neurons growing as $\varepsilon^{-2(d+1)/(d+3)}$ (both slightly worse than ours).

Finally, the number of neurons needed to express a function with a bound on the γ_2-norm can be estimated from general results on approximating reproducing kernel Hilbert space described in Section 2.3, whose kernel can be expressed as an expectation. Indeed, Bach (2017) shows that with k neurons, one can approximate a function in $\mathcal{F}_2$ with unit γ_2-norm with an error measured in L_2 of $\varepsilon = k^{-(d+3)/(2d)}$. When inverting the relationship between k and ε, we get a number of neurons scaling as $\varepsilon^{-2d/(d+3)}$, which is the same as in Prop. 1 but with an error in L_2-norm instead of L^∞-norm.

4.2 Sufficient Conditions for Finite Variation

In this section and the next one, we study more precisely the RKHS $\mathcal{G}_2$ (and thus obtain similar results for $\mathcal{G}_1 \supset \mathcal{G}_2$). The kernel $k(x, y) = \int_{\mathbb{S}^d} (v^\top x)_+ (v^\top y)_+ d\tau_d(v)$ defined on the sphere $\mathbb{S}^d$ belongs to the family of dot-product kernels (Smola et al., 2001) that only depends on the dot-product $x^\top y$, although in our situation, the function is not particularly simple (see formulas in Section 3.1). The analysis of these kernels is similar to one of translation-invariant kernels; for $d = 1$, i.e., on the 2-dimensional sphere, it is done through Fourier series; while for $d > 1$, *spherical harmonics* have to be used as the expansion of functions in series of spherical harmonics make the computation of the RKHS norm explicit (see a review of spherical harmonics in Appendix D.1 with several references therein). Since the calculus is tedious, all proofs are put in appendices, and we only present here the main results. In this section, we provide simple sufficient conditions for belonging to $\mathcal{G}_2$ (and hence $\mathcal{G}_1$) based on the existence and boundedness of derivatives, while in the next section, we show how any Lipschitz-function may be approximated by functions in $\mathcal{G}_2$ (and hence $\mathcal{G}_1$) with precise control of the norm of the approximating functions.

The derivatives of functions defined on $\mathbb{S}^d$ may be defined in several ways, using the manifold structure (see, e.g., Absil et al., 2009) or through polar coordinates (see, e.g., Atkinson and Han, 2012, Chapter 3). For $d = 1$, the one-dimensional sphere $\mathbb{S}^1 \subset \mathbb{R}^2$ may be parameterized by a single angle and thus the notion of derivatives and the proof of the following result is simpler and based on Fourier series (see Appendix C.2). For the general proof based on spherical harmonics, see Appendix D.2.

Proposition 2 (Finite variation on the sphere) *Assume that $g : \mathbb{S}^d \to \mathbb{R}$ is such that all i-th order derivatives exist and are upper-bounded in absolute value by η for $i \in \{0, \dots, s\}$, where s is an integer such that $s \geqslant (d-1)/2 + \alpha + 1$. Assume g is even if α is odd (and vice-versa); then $g \in \mathcal{G}_2$ and $\gamma_2(g) \leqslant C(d, \alpha) \eta$, for a constant $C(d, \alpha)$ that depends only on d and α.*

We can make the following observations:

– *Tightness of conditions*: as shown in Appendix D.5, there are functions g, which have bounded first s derivatives and do not belong to $\mathcal{G}_2$ while $s \leqslant \frac{d}{2} + \alpha$ (at least when $s - \alpha$ is even). Therefore, when $s - \alpha$ is even, the scaling in $(d-1)/2 + \alpha$ is optimal.

– *Dependence on α*: for any d, the higher the α, the stricter the sufficient condition. Given that the estimation error grows slowly with α (see Section 5.1), low values of α would be preferred in practice.

– *Dependence on d*: a key feature of the sufficient condition is the dependence on d, that is, as d increases the number of derivatives has to increase in $d/2$—like for Sobolev spaces in dimension d (Adams and Fournier, 2003). This is another instantiation of the curse of dimensionality: only very smooth functions in high dimensions are allowed.

– *Special case $d = 1$, $\alpha = 0$*: differentiable functions on the sphere in $\mathbb{R}^2$, with bounded derivatives, belong to $\mathcal{G}_2$, and thus all Lipschitz-continuous functions, because Lipschitz-continuous functions are almost everywhere differentiable with bounded derivative (Adams and Fournier, 2003).

4.3 Approximation of Lipschitz-continuous Functions

In order to derive generalization bounds for target functions which are not sufficiently differentiable (and may not be in $\mathcal{G}_2$ or $\mathcal{G}_1$), we need to approximate any Lipschitz-continuous function, with a function $g \in \mathcal{G}_2$ with a norm $\gamma_2(g)$ that will grow as the approximation gets tighter. We give precise rates in the proposition below. Note the requirement for parity of the function g. The result below notably shows the density of $\mathcal{G}_1$ in uniform norm in the space of Lipschitz-continuous functions of the given parity, which is already known since our activation functions are not polynomials (Leshno et al., 1993).

Proposition 3 (Approximation of Lipschitz-continuous functions on the sphere) *For δ greater than a constant depending only on d and α, for any function $g : \mathbb{S}^d \to \mathbb{R}$ such that for all $x, y \in \mathbb{S}^d$, $g(x) \leqslant \eta$ and $|g(x) - g(y)| \leqslant \eta \|x - y\|_2$, and g is even if α is odd (and vice-versa), there exists $h \in \mathcal{G}_2$, such that $\gamma_2(h) \leqslant \delta$ and*

$$\sup_{x \in \mathbb{S}^d} |h(x) - g(x)| \leqslant C(d, \alpha) \eta \Big(\frac{\delta}{\eta} \Big)^{-1/(\alpha + (d-1)/2)} \log \Big(\frac{\delta}{\eta} \Big).$$

This proposition is shown in Appendix C.3 for $d = 1$ (using Fourier series) and in Appendix D.4 for all $d \geqslant 1$ (using spherical harmonics). We can make the following observations:

– *Dependence in δ and η*: as expected, the main term in the error bound $(\delta/\eta)^{-1/(\alpha+(d-1)/2)}$ is a decreasing function of δ/η, that is when the norm $\gamma_2(h)$ is allowed to grow, the approximation gets tighter, and when the Lipschitz constant of g increases, the approximation is less tight.

– *Dependence on d and α*: the rate of approximation is increasing in d and α. In particular the approximation properties are better for low α.

– *Special case $d = 1$ and $\alpha = 0$*: up to the logarithmic term we recover the result of Prop. 2, that is, the function g is in $\mathcal{G}_2$.

– *Tightness*: in Appendix D.5, we provide a function which is not in the RKHS and for which the tightest possible approximation scales as $\delta^{-2/(d/2+\alpha-2)}$. Thus the linear scaling of the rate as $d/2 + \alpha$ is not improvable (but constants are).

4.4 Linear Functions

In this section, we consider a linear function on $\mathbb{S}^d$, that is $g(x) = v^\top x$ for a certain $v \in \mathbb{S}^d$, and compute its norm (or upper-bound thereof) both for $\mathcal{G}_1$ and $\mathcal{G}_2$, which is independent of v and finite. In the following propositions, the notation $\approx$ means asymptotic equivalents when $d \to \infty$.

Proposition 4 (Norms of linear functions on the sphere) *Assume that $g : \mathbb{S}^d \to \mathbb{R}$ is such $g(x) = v^\top x$ for a certain $v \in \mathbb{S}^d$. If $\alpha = 0$, then $\gamma_1(g) \leqslant \gamma_2(g) = \frac{2d\pi}{d-1} \approx 2\pi$. If $\alpha = 1$, then $\gamma_1(g) \leqslant 2$, and for all $\alpha \geqslant 1$, $\gamma_1(g) \leqslant \gamma_2(g) = \frac{d}{d-1} \frac{4\pi}{\alpha} \frac{\Gamma(\alpha/2 + d/2 + 1)}{\Gamma(\alpha/2)\Gamma(d/2+1)} \approx C d^{\alpha/2}$.*

We see that for $\alpha = 1$, the γ_1-norm is less than a constant, and is much smaller than the γ_2-norm (which scales as $\sqrt{d}$). For $\alpha \geqslant 2$, we were not able to derive better bounds for γ_1 (other than the value of γ_2).

4.5 Functions of Projections

If $g(x) = \varphi(w^\top x)$ for some unit-norm $w \in \mathbb{R}^{d+1}$ and φ a function defined on the real-line, then the value of the norms γ_2 and γ_1 differ significantly. Indeed, for γ_1, we may consider a new variable $\tilde{x} \in \mathbb{S}^1 \subset \mathbb{R}^2$, with its first component $\tilde{x}_1 = w^\top x$, and the function $\tilde{g}(\tilde{x}) = \varphi(\tilde{x}_1)$. We may then apply Prop. 2 to $\tilde{g}$ with $d = 1$. That is, if φ is $(\alpha+1)$-times differentiable with bounded derivatives, there exists a decomposition $\tilde{g}(\tilde{x}) = \int_{\mathbb{S}^1} \tilde{\mu}(\tilde{v}) \sigma(\tilde{v}^\top \tilde{x}) d\tilde{\mu}$, with $\gamma_1(\tilde{g}) = |\tilde{\mu}|(\mathbb{S}^1)$, which is *not* increasing in d. If we consider any vector $t \in \mathbb{R}^{d+1}$ which is orthogonal to w in $\mathbb{R}^{d+1}$, then, we may define a measure μ supported in the circle defined by the two vectors w and t and which is equal to $\tilde{\mu}$ on that circle. The total variation of μ is the one of $\tilde{\mu}$ while g can be decomposed using μ and thus $\gamma_1(g) \leqslant \gamma_1(\tilde{g})$. Similarly, Prop. 3 could also be applied (and will for obtaining generalization bounds), also our reasoning works for any low-dimensional projections: the dependence on a lower-dimensional projection allows to reduce smoothness requirements.

However, for the RKHS norm γ_2, this reasoning does not apply. For example, a certain function φ exists, which is s-times differentiable, as shown in Appendix D.5, for $s \leqslant \frac{d}{2} + \alpha$ (when $s - \alpha$ is even), and is not in $\mathcal{G}_2$. Thus, given Prop. 2, the dependence on a uni-dimensional projection does not make a difference regarding the level of smoothness which is required to belong to $\mathcal{G}_2$.

4.6 From the Unit-sphere $\mathbb{S}^d$ to $\mathbb{R}^{d+1}$

We now extend the results above to functions defined on $\mathbb{R}^d$, to be approximated by functions in $\mathcal{F}_1$ and $\mathcal{F}_2$. More precisely, we first extend Prop. 2 and Prop. 3, and then consider norms of linear functions and functions of projections.

Proposition 5 (Finite variation) *Assume that $f : \mathbb{R}^d \to \mathbb{R}$ is such that all i-th order derivatives exist and are upper-bounded on the ball $\{\|x\|_q \leqslant R\}$ by η/R^i for $i \in \{0, \dots, k\}$, where s is the smallest integer such that $s \geqslant (d-1)/2 + \alpha + 1$; then $f \in \mathcal{F}_2$ and $\gamma_2(f) \leqslant C(d,\alpha)\eta$, for a constant $C(d,\alpha)$ that depends only on d and α.*

Proof By assumption, the function $x \mapsto f(Rx)$ has all its derivatives bounded by a constant times η. Moreover, we have defined $g(t,a) = f(\frac{Rt}{a})a^\alpha$ so that all derivatives are bounded by η. The result then follows immediately from Prop. 2. ■

Proposition 6 (Approximation of Lipschitz-continuous functions) *For δ larger than a constant that depends only on d and α, for any function $f : \mathbb{R}^d \to \mathbb{R}$ such that for all x, y such that $\|x\|_q \leqslant R$ and $\|y\|_q \leqslant R$, $|f(x)| \leqslant \eta$ and $|f(x) - f(y)| \leqslant \eta R^{-1}\|x - y\|_q$, there exists $g \in \mathcal{F}_2$ such that $\gamma_2(g) \leqslant \delta$ and*

$$\sup_{\|x\|_q \leqslant R} |f(x) - g(x)| \leqslant C(d,\alpha)\eta\Big(\frac{\delta}{\eta}\Big)^{-1/(\alpha+(d-1)/2)} \log\Big(\frac{\delta}{\eta}\Big).$$

Proof With the same reasoning as above, we obtain that g is Lipschitz-continuous with constant η, we thus get the desired approximation error from Prop. 3. ■

Linear functions. If $f(x) = w^\top x + b$, with $\|w\|_2 \leqslant \eta$ and $b \leqslant \eta R$, then for $\alpha = 1$, it is straightforward that $\gamma_1(f) \leqslant 2R\eta$. Moreover, we have $\gamma_2(f) \sim CR\eta$. For other values of α, we also have γ_1-norms less than a constant (depending *only* of α) times $R\eta$. The RKHS norms are bit harder to compute since linear functions for f leads to linear functions for g only for $\alpha = 1$.

Functions of projections. If $f(x) = \varphi(w^\top x)$ where $\|w\|_2 \leqslant \eta$ and $\varphi : \mathbb{R} \to \mathbb{R}$ is a function, then the norm of f is the same as the norm of the function φ on the interval $[-R\eta, R\eta]$, and it thus does not depend on d. This is a consequence of the fact that the total mass of a Radon measure remains bounded even when the support has measure zero (which might not be the case for the RKHS defined in Section 2.3). For the RKHS, there is no such results and it is in general not adaptive.

More generally, if $f(x) = \Phi(W^\top x)$ for $W \in \mathbb{R}^{d\times s}$ with the largest singular value of W less than η, and Φ a function from $\mathbb{R}^s$ to $\mathbb{R}$, then for $\|x\|_2 \leqslant R$, we have $\|W^\top x\|_2 \leqslant R\eta$, and thus we may apply our results for $d = s$.

ℓ_1-penalty on input weights (p=1). When using an ℓ_1-penalty on input weights instead of an ℓ_2-penalty, the results in Prop. 5 and 6 are unchanged (only the constants that depend on d are changed). Moreover, when $\|x\|_\infty \leqslant 1$ almost surely, functions of the form $f(x) = \varphi(w^\top x)$ where $\|w\|_1 \leqslant \eta$ and $\varphi : \mathbb{R} \to \mathbb{R}$ is a function, will also inherit from properties of φ (without any dependence on dimension). Similarly, for functions of the form $f(x) = \Phi(W^\top x)$ for $W \in \mathbb{R}^{d\times s}$ with all columns of ℓ_1-norm less than η, we have $\|W^\top x\|_\infty \leqslant R\eta$ and we can apply the s-dimensional result.

4.7 Related Work

In this section, we show how our results from the previous sections relate to existing work on neural network approximation theory.

Approximation of Lipschitz-continuous functions with finitely many neurons. In this section, we only consider the case $\alpha = 1$, for which we have two approximation bounds: Prop. 6 which approximates any η-Lipschitz-continuous function by a function with finite γ_1-norm less than δ and uniform error less than $\eta(\delta/\eta)^{-2/(d+1)}\log(\delta/\eta)$, and Prop. 1 which shows that a function with γ_1-norm less than δ, may be approximated with r neurons with uniform error $\delta r^{-(d+3)/(2d)}$.

Thus, given r neurons, we get an approximation of the original function with uniform error

$$\eta(\delta/\eta)^{-2/(d+1)}\log(\delta/\eta) + \delta r^{-(d+3)/(2d)}.$$

We can optimize over δ, and use $\delta = \eta n^{(d+1)/(2d)}$, to obtain a uniform approximation bound proportional to $\eta(\log n)n^{-1/d}$, for approximating an η-Lipschitz-continuous function with n neurons.

Approximation by ridge functions. The approximation properties of single hidden layer neural networks have been studied extensively, where they are often referred to as "ridge function" approximations. As shown by Pinkus (1999, Corollary 6.10)—based on a result from Petrushev (1998), the approximation order of $n^{-1/d}$ for the rectified linear unit was already known, but only in L_2-norm (and without the factor $\log n$), and without any constraints on the input and output weights. In this paper, we provide an explicit control of the various weights, which is needed for computing estimation errors. Moreover, while the two proof techniques use spherical harmonics, the proof of Petrushev (1998) relies on quadrature formulas for the associated Legendre polynomials, while ours relies on the relationship with the associated positive definite kernels, is significantly simpler, and offers additional insights into the problem (relationship with convex neural networks and zonoids). Maiorov (2006, Theorem 2.3) also derives a similar result, but in L_2-norm (rather than uniform norm), and for sigmoidal activation functions (which are bounded). Note finally, that the order $O(n^{-1/d})$ cannot be improved (DeVore et al., 1989, Theorem 4.2). Also, Maiorov and Meir (2000, Theorem 5) derive similar upper and lower bounds based on a random sampling argument which is close to using random features in the RKHS setting described in Section 2.3.

Relationship to hardness results for Boolean-valued functions. In this paper, we consider a particular view of the curse of dimensionality and ways of circumventing it, that is, our distribution over inputs is arbitrary, but our aim is to approximate a real-valued function. Thus, all hardness results depending on functions with values in $\{0,1\}$ do not apply there directly—see, e.g., Shalev-Shwartz and Ben-David (2014, Chapter 20), for the need of exponentially many hidden units for approximating most of the functions from $\{0,1\}^d$ to $\{0,1\}$.

Our approximation bounds show that, without any assumption beyond Lipschitz-continuity of the target function, it sufficient to have a number of hidden units which is still exponential in dimension (hence we also suffer from the curse of dimensionality), but a soon as the target function depends on linear low-dimensional structure, then we lose this exponential

dependence. It would be interesting to study an extension to $\{0,1\}$-valued functions, and also to relate our results to the number of linear regions delimited by neural networks with rectified linear units (Montufar et al., 2014).

5. Generalization Bounds

Our goal is to derive the generalization bounds outlined in Section 2.4 for neural networks with a single hidden layer. The main results that we obtain are summarized in Table 2 and show adaptivity to assumptions that avoid the curse of dimensionality.

More precisely, given some distribution over the pairs $(x,y) \in \mathcal{X} \times \mathcal{Y}$, a loss function $\ell : \mathcal{Y} \times \mathbb{R} \to \mathbb{R}$, our aim is to find a function $f : \mathbb{R}^d \to \mathbb{R}$ such that $J(f) = \mathbb{E}[\ell(y, f(x))]$ is small, given some i.i.d. observations (x_i, y_i), $i = 1, \dots, n$. We consider the empirical risk minimization framework over a space of functions $\mathcal{F}$, equipped with a norm γ (in our situations, $\mathcal{F}_1$ and $\mathcal{F}_2$, equipped with γ_1 or γ_2). The empirical risk $\hat{J}(f) = \frac{1}{n}\sum_{i=1}^n \ell(y_i, f(x_i))$, is minimized by constraining f to be in the ball $\mathcal{F}^\delta = \{f \in \mathcal{F},\ \gamma(f) \leqslant \delta\}$.

We assume that almost surely, $\|x\|_q \leqslant R$, that for all y the function $u \mapsto \ell(y,u)$ is G-Lipschitz-continuous on $\{|u| \leqslant \sqrt{2}\delta\}$, and that almost surely, $\ell(y,0) \leqslant G\delta$. As before z denotes $z = (x^\top, R)^\top$ so that $\|z\|_q \leqslant \sqrt{2}R$. This corresponds to the following examples:

- Logistic regression and support vector machines: we have $G = 1$.
- Least-squares regression: we take $G = \max\{\sqrt{2}\delta + \|y\|_\infty, \frac{\|y\|_\infty^2}{\sqrt{2}\delta}\}$.

Approximation errors $\inf_{f\in\mathcal{F}^\delta} J(f) - \inf_{f\in\mathcal{F}} J(f)$ will be obtained from the approximation results from Section 4 by assuming that the optimal target function f_* has a specific form. Indeed, we have:

$$\inf_{f\in\mathcal{F}^\delta} J(f) - J(f_*) \leqslant G \inf_{f\in\mathcal{F}^\delta} \Big\{ \sup_{\|x\|_q \leqslant R} |f(x) - f_*(x)| \Big\}.$$

We now deal with estimation errors $\sup_{f\in\mathcal{F}^\delta} |\hat{J}(f) - J(f)|$ using Rademacher complexities.

5.1 Estimation Errors and Rademacher Complexity

The following proposition bounds the uniform deviation between J and its empirical counterpart $\hat{J}$. This result is standard (see, e.g., Koltchinskii, 2001; Bartlett and Mendelson, 2003) and may be extended in bounds that hold with high-probability.

Proposition 7 (Uniform deviations) *We have the following bound on the expected uniform deviation:*

$$\mathbb{E}\Big[\sup_{\gamma_1(f)\leqslant\delta} |J(f) - \hat{J}(f)|\Big] \leqslant 4\frac{G\delta}{\sqrt{n}} C(p,d,\alpha),$$

with the following constants:

- *for $\alpha \geqslant 1$, $C(p,d,\alpha) \leqslant \alpha\sqrt{2\log(d+1)}$ for $p = 1$ and $C(p,d,\alpha) \leqslant \frac{\alpha}{\sqrt{p-1}}$ for $p \in (1,2]$*
- *for $\alpha = 0$, $C(p,d,\alpha) \leqslant C\sqrt{d+1}$, where C is a universal constant.*

Proof We use the standard framework of Rademacher complexities and get:

$$\begin{aligned}
&\mathbb{E}\sup_{\gamma_1(f)\leqslant\delta} |J(f) - \hat{J}(f)| \\
\leqslant\ & 2\mathbb{E}\sup_{\gamma_1(f)\leqslant\delta} \Big|\frac{1}{n}\sum_{i=1}^n \tau_i \ell(y_i, f(x_i))\Big| \text{ using Rademacher random variables } \tau_i, \\
\leqslant\ & 2\mathbb{E}\sup_{\gamma_1(f)\leqslant\delta} \Big|\frac{1}{n}\sum_{i=1}^n \tau_i \ell(y_i, 0)\Big| + 2\mathbb{E}\sup_{\gamma_1(f)\leqslant\delta} \Big|\frac{1}{n}\sum_{i=1}^n \tau_i [\ell(y_i, f(x_i)) - \ell(y_i,0)]\Big| \\
\leqslant\ & 2\frac{G\delta}{\sqrt{n}} + 2G\mathbb{E}\sup_{\gamma(f)\leqslant\delta} \Big|\frac{1}{n}\sum_{i=1}^n \tau_i f(x_i)\Big| \text{ using the Lipschitz-continuity of the loss,} \\
\leqslant\ & 2\frac{G\delta}{\sqrt{n}} + 2G\delta\mathbb{E}\sup_{\|v\|_p\leqslant 1/R} \Big|\frac{1}{n}\sum_{i=1}^n \tau_i (v^\top z_i)_+^\alpha\Big| \text{ using Eq. (2).}
\end{aligned}$$

We then take different routes for $\alpha \geqslant 1$ and $\alpha = 0$.

For $\alpha \geqslant 1$, we have the upper-bound

$$\begin{aligned}
\mathbb{E}\sup_{\gamma_1(f)\leqslant\delta} |J(f) - \hat{J}(f)| &\leqslant 2\frac{G\delta}{\sqrt{n}} + 2G\delta\alpha\mathbb{E}\sup_{\|v\|_p\leqslant 1/R} \Big|\frac{1}{n}\sum_{i=1}^n \tau_i v^\top z_i\Big| \\
&\qquad \text{using the } \alpha\text{-Lipschitz-cont. of } (\cdot)_+^\alpha \text{ on } [-1,1], \\
&\leqslant 2\frac{G\delta}{\sqrt{n}} + 2\frac{G\alpha\delta}{Rn}\mathbb{E}\Big\|\sum_{i=1}^n \tau_i z_i\Big\|_q.
\end{aligned}$$

From Kakade et al. (2009), we get the following bounds on Rademacher complexities:

- If $p \in (1,2]$, then $q \in [2,\infty)$, and $\mathbb{E}\|\sum_{i=1}^n \tau_i z_i\|_q \leqslant \sqrt{q-1}R\sqrt{n} = \frac{1}{\sqrt{p-1}}R\sqrt{n}$
- If $p = 1$, then $q = \infty$, and $\mathbb{E}\|\sum_{i=1}^n \tau_i z_i\|_q \leqslant R\sqrt{n}\sqrt{2\log(d+1)}$.

Overall, we have $\mathbb{E}\|\sum_{i=1}^n \tau_i z_i\|_q \leqslant \sqrt{n}RC(p,d)$ with $C(p,d)$ defined above, and thus

$$\mathbb{E}\sup_{\gamma(f)\leqslant\delta} |J(f) - \hat{J}(f)| \leqslant 2\frac{G\delta}{\sqrt{n}}(1 + \alpha C(p,d)) \leqslant 4\frac{G\delta\alpha}{\sqrt{n}} C(p,d).$$

For $\alpha = 0$, we can simply go through the VC-dimension of half-hyperplanes, which is equal to d, and Theorem 6 from Bartlett and Mendelson (2003), that shows that $\mathbb{E}\sup_{v\in\mathbb{R}^{d+1}} \Big|\frac{1}{n}\sum_{i=1}^n \tau_i 1_{v^\top z_i}\Big| \leqslant C\frac{\sqrt{d+1}}{\sqrt{n}}$, where C is a universal constant.

Note that using standard results from Rademacher complexities, we have, with probability greater than $1-u$, $\sup_{\gamma_1(f)\leqslant\delta} |J(f) - \hat{J}(f)| \leqslant \mathbb{E}\sup_{\gamma_1(f)\leqslant\delta} |J(f) - \hat{J}(f)| + \frac{2G\delta}{\sqrt{n}}\sqrt{\log\frac{2}{u}}$. ■

5.2 Generalization Bounds for ℓ_2-norm Constraints on Input Weights ($p = 2$)

We now provide generalization bounds for the minimizer of the empirical risk given the contraint that $\gamma_1(f) \leqslant \delta$ for a well chosen δ, that will depend on the assumptions regarding the target function f_*, listed in Section 1. In this section, we consider an ℓ_2-norm on input weights w, while in the next section, we consider the ℓ_1-norm. The two situations are summarized and compared in Table 2, where we consider that $\|x\|_\infty \leqslant r$ almost surely, which implies that our bound R will depend on dimension as $R \leqslant r\sqrt{d}$.

Our generalization bounds are expected values of the excess expected risk for a our estimator (where the expectation is taken over the data).

Affine functions. We assume $f_*(x) = w^\top x + b$, with $\|w\|_2 \leqslant \eta$ and $|b| \leqslant R\eta$. Then, as seen in Section 4.6, $f_* \in \mathcal{F}_1$ with $\gamma_1(f_*) \leqslant C(\alpha)\eta R$ (the constant is independent of d because we approximate an affine function). From Prop. 7, we thus get a generalization bound proportional to $\frac{GR\eta}{\sqrt{n}}$ times a constant (that may depend on α), which is the same as assuming directly that we optimize over linear predictors only. The chosen δ is then a constant times $R\eta$, and does not grow with n, like in parametric estimation (although we do use a non-parametric estimation procedure).

Projection pursuit. We assume $f_*(x) = \sum_{j=1}^k f_j(w_j^\top x)$, with $\|w_j\|_2 \leqslant \eta$ and each f_j bounded by ηR and 1-Lipschitz continuous. From Prop. 6, we may approach each $x \mapsto f_j(w_j^\top x)$ by a function with γ_1-norm less than $\delta\eta R$ and uniform approximation $C(\alpha)\eta R\delta^{-1/\alpha}\log\delta$. This leads to a total approximation error of $kC(\alpha)G\eta R\delta^{-1/\alpha}\log\delta$ for a norm less than $k\delta\eta R$ (the constant is independent of d because we approximate a function of one-dimensional projection).

For $\alpha \geqslant 1$, from Prop. 7, the estimation error is $\frac{kGR\eta\delta}{\sqrt{n}}$, with an overall bound of $C(\alpha)kGR\eta(\frac{\delta}{\sqrt{n}}+\delta^{-1/\alpha}\log\delta)$. With $\delta = n^{\alpha/2(\alpha+1)}$ (which grows with n), we get an optimized generalization bound of $C(\alpha)kGR\eta\frac{\log n}{n^{1/(2\alpha+2)}}$, with a scaling independent of the dimension d (note however that R typically grow with $\sqrt{d}$, i.e., $r\sqrt{d}$, if we have a bound in ℓ_∞-norm for all our inputs x).

For $\alpha = 0$, from Prop. 5, the target function belongs to $\mathcal{F}_1$ with a norm less than $kGR\eta$, leading to an overall generalization bound of $\frac{kGR\eta\sqrt{d}}{\sqrt{n}}$.

Note that when the functions f_j are exactly the activation functions, the bound is better, as these functions directly belong to the space $\mathcal{F}_1$.

Multi-dimensional projection pursuit. We extend the situation above, by assuming $f_*(x) = \sum_{j=1}^k F_j(W_j^\top x)$ with each $W_j \in \mathbb{R}^{d\times s}$ having all singular values less than η and each F_j bounded by ηR and 1-Lipschitz continuous. From Prop. 6, we may approach each $x \mapsto F_j(W_j^\top x)$ by a function with γ_1-norm less than $\delta\eta R$ and uniform approximation $C(\alpha,s)\eta R\delta^{-1/(\alpha+(s-1)/2)}\log\delta$. This leads to a total approximation error of $kC(\alpha,s)G\eta R\delta^{-1/(\alpha+(s-1)/2)}\log\delta$.

For $\alpha \geqslant 1$, the estimation error is $kGR\eta\delta/\sqrt{n}$, with an overall bound of $C(\alpha,s)kGR\eta(\delta/\sqrt{n}+\delta^{-1/(\alpha+(s-1)/2)}\log\delta)$. With $\delta = n^{(\alpha+(s-1)/2)/(2\alpha+s-1)}$, we get an optimized bound of $\frac{C(\alpha,s)kGR\eta}{n^{1/(2\alpha+s+1)}}\log n$.

For $\alpha = 0$, we have an overall bound of $C(s)kGR\eta(\delta^{-2/(s-1)}\log\delta + \frac{\delta\sqrt{d}}{\sqrt{n}})$, and with $\delta = (n/d)^{(s-1)/(s+1)}$, we get a generalization bound scaling as $\frac{C(s)kGR\eta}{(n/d)^{1/(s+1)}}\log(n/d)$.

function space	$\|\cdot\|_2$, $\alpha \geqslant 1$	$\|\cdot\|_1$, $\alpha \geqslant 1$	$\alpha = 0$
$w^\top x + b$	$\frac{d^{1/2}}{n^{1/2}}$	$\sqrt{q}(\frac{\log d}{n})^{1/2}$	$\frac{(dq)^{1/2}}{n^{1/2}}$
$\sum_{j=1}^k f_j(w_j^\top x)$, $w_j \in \mathbb{R}^d$	$\frac{kd^{1/2}}{n^{1/(2\alpha+2)}}\log n$	$\frac{kq^{1/2}(\log d)^{1/(\alpha+1)}}{n^{1/(2\alpha+2)}}\log n$	$\frac{k(dq)^{1/2}}{n^{1/2}}$
$\sum_{j=1}^k f_j(W_j^\top x)$, $W_j \in \mathbb{R}^{d\times s}$	$\frac{kd^{1/2}}{n^{1/(2\alpha+s+1)}}\log n$	$\frac{kq^{1/2}(\log d)^{1/(\alpha+(s+1)/2)}}{n^{1/(2\alpha+s+1)}}\log n$	$\frac{(dq)^{1/2}d^{1/(s+1)}}{n^{1/(s+1)}}\log n$

Table 2: Summary of generalization bounds with different settings. See text for details.

Note that for $s = d$ and $k = 1$, we recover the usual Lipschitz-continuous assumption, with a rate of $\frac{C(\alpha,d)kGR\eta}{n^{1/(2\alpha+d+1)}}\log n$.

We can make the following observations:

- *Summary table*: when we know a bound r on all dimensions of x, then we may take $R = r\sqrt{d}$; this is helpful in comparisons in Table 2, where R is replaced by $r\sqrt{d}$ and the dependence in r is removed as it is the same for all models.
- *Dependence on d*: when making only a global Lipschitz-continuity assumption, the generalization bound has a bad scaling in n, i.e., as $n^{-1/(2\alpha+d+1)}$, which goes down to zero slowly when d increases. However, when making structural assumptions regarding the dependence on unknown lower-dimensional subspaces, the scaling in d disappears.
- *Comparing different values of α*: the value $\alpha = 0$ always has the best scaling in n, but constants are better for $\alpha \geqslant 1$ (among which $\alpha = 1$ has the better scaling in n).
- *Bounds for $\mathcal{F}_2$*: The simplest upper bound for the penalization by the space $\mathcal{F}_2$ depends on the approximation properties of $\mathcal{F}_2$. For linear functions and $\alpha = 1$, it is less than $\sqrt{d}\eta R$, with a bound $\frac{GR\eta\sqrt{d}}{\sqrt{n}}$. For the other values of α, there is a constant $C(d)$. Otherwise, there is no adaptivity and all other situations only lead to upper-bounds of $O(n^{-1/(2\alpha+d+1)})$. See more details in Section 5.4.
- *Sample complexity*: Note that the generalization bounds above may be used to obtain sample complexity results such as $d\varepsilon^{-2}$ for affine functions, $(\varepsilon k^{-1}d^{-1/2})^{-2\alpha-2}$ for projection pursuit, and $(\varepsilon k^{-1}d^{-1/2})^{-s-1-2\alpha}$ for the generalized version (up to logarithmic terms).
- *Relationship to existing work*: Maiorov (2006, Theorem 1.1) derives similar results for neural networks with sigmoidal activation functions (that tend to one at infinity) and the square loss only, and for a level of smoothness of the target function which grows with dimension (in this case, once can get easily rates of $n^{-1/2}$). Our result holds for problems where only bounded first-order derivatives are assumed, but by using Prop. 2, we would get similar rate by ensuring the target function belongs to $\mathcal{F}_2$ and hence to $\mathcal{F}_1$.

Lower bounds. In the sections above, we have only provided generalization bounds. Although interesting, deriving lower-bounds for the generalization performance when the target function belongs to certain function classes is out of the scope of this paper. Note however, that results from Sridharan (2012) suggest that the Rademacher complexities of the associated function classes provide such lower-bounds. For general Lipschitz-functions, these Rademacher complexities decreases as $n^{-\max\{d,2\}}$ (von Luxburg and Bousquet, 2004).

5.3 Generalization Bounds for ℓ_1-norm Constraints on Input Weights ($p = 1$)

We consider the same three situations, assuming that linear predictors have at most q non-zero elements. We assume that each component of x is almost surely bounded by r (i.e., a bound in ℓ_∞-norm).

Affine functions. We assume $f_*(x) = w^\top x + b$, with $\|w\|_2 \leqslant \eta$ and $|b| \leqslant R\eta$. Given that we have assumed that w has at most q non-zeros, we have $\|w\|_1 \leqslant \sqrt{q}\eta$.

Then, $f_* \in \mathcal{F}_1$ with $\gamma_1(f) \leqslant C(\alpha)\eta r\sqrt{q}$, with a constant that is independent of d because we have an affine function.

From Prop. 7, we thus get a rate of $\frac{Gr\eta\sqrt{q\log(d)}}{\sqrt{n}}$ times a constant (that may depend on α), which is the same as assuming directly that we optimize over linear predictors only (see, for example, Bühlmann and Van De Geer, 2011). We recover a high-dimensional phenomenon (although with a slow rate in $1/\sqrt{n}$), where d may be much larger than n, as long as $\log d$ is small compared to n. The chosen δ is then a constant times $r\eta\sqrt{q}$ (and does not grow with n).

Projection pursuit. We assume $f_*(x) = \sum_{j=1}^k f_j(w_j^\top x)$, with $\|w_j\|_2 \leqslant \eta$ (which implies $\|w_j\|_1 \leqslant \sqrt{q}\eta$ given our sparsity assumption) and each f_j bounded by $\eta r\sqrt{q}$ and 1-Lipschitz continuous. We may approach each $x \mapsto f_j(w_j^\top x)$ by a function with γ_1-norm less than $\delta\eta r\sqrt{q}$ and uniform approximation $C(\alpha)\eta r\sqrt{q}\delta^{-1/\alpha}\log\delta$, with a constant that is independent of d because we have a function of one-dimensional projection. This leads to a total approximation error of $kC(\alpha)G\eta r\sqrt{q}\delta^{-1/\alpha}\log\delta$ for a norm less than $k\delta\eta r\sqrt{q}$.

For $\alpha \geqslant 1$, the estimation error is $\frac{kGr\eta\delta\sqrt{q\log d}}{\sqrt{n}}$, with an overall bound of $C(\alpha)kGr\sqrt{q}\eta(\delta^{-1/\alpha}\log\delta + \frac{\delta\sqrt{\log d}}{\sqrt{n}})$. With $\delta = (n/\log d)^{\alpha/2(\alpha+1)}$, we get an optimized bound of $C(\alpha)kGr\sqrt{q}\eta\frac{\log n(\log d)^{1/(2\alpha+2)}}{n^{1/(2\alpha+2)}}$, with a scaling only dependent in d with a logarithmic factor.

For $\alpha = 0$, the target function belongs to $\mathcal{F}_1$ with a norm less than $kGr\sqrt{q}\eta$, leading to an overal bound of $\frac{kGr\eta\sqrt{q\log d}}{\sqrt{n}}$ (the sparsity is not helpful in this case).

Multi-dimensional projection pursuit. We assume $f_*(x) = \sum_{j=1}^k F_j(W_j^\top x)$ with each $W_j \in \mathbb{R}^{d\times s}$, having all columns with ℓ_2-norm less than η (note that this is a weaker requirement than having all singular values that are less than η). If we assume that each of these columns has at most q non-zeros, then the ℓ_1-norms are less than $r\sqrt{q}$ and we may use the approximation properties described at the end of Section 4.6. We also assume that each F_j is bounded by $\eta r\sqrt{q}$ and 1-Lipschitz continuous (with respect to the ℓ_2-norm).

We may approach each $x \mapsto F_j(W_j^\top x)$ by a function with γ_1-norm less than $\delta\eta r\sqrt{q}$ and uniform approximation $C(\alpha,s)\eta r\sqrt{q}\delta^{-1/(\alpha+(s-1)/2)}\log\delta$. This leads to a total approximation error of $kC(\alpha,s)G\eta r\sqrt{q}\delta^{-1/(\alpha+(s-1)/2)}\log\delta$.

For $\alpha \geqslant 1$, the estimation error is $kGr\sqrt{q}\eta\delta\sqrt{\log d}/\sqrt{n}$, with an overall bound which is equal to $C(\alpha,s)kGr\sqrt{q}\eta(\delta^{-1/(\alpha+(s-1)/2)}\log\delta + \frac{\delta\sqrt{\log d}}{\sqrt{n}})$. With $\delta = (n/\log d)^{(\alpha+(s-1)/2)/(2\alpha+s-1)}$, we get an optimized bound of $\frac{C(\alpha,s)kGr\sqrt{q}\eta(\log d)^{1/(2\alpha+s+1)}}{n^{1/(2\alpha+s+1)}}\log n$.

For $\alpha = 0$, we have the bound $\frac{C(s)kGr\sqrt{q}\eta}{(n/d)^{1/(s+1)}}\log(n/d)$, that is we cannot use the sparsity as the problem is invariant to the chosen norm on hidden weights.

We can make the following observations:

- *High-dimensional variable selection*: when $k = 1$, $s = q$ and W_1 is a projection onto q variables, then we obtain a bound proportional to $\frac{\sqrt{q}\eta(\log d)^{1/(2\alpha+s+1)}}{n^{1/(2\alpha+s+1)}}\log n$, which exhibits a high-dimensional scaling in a non-linear setting. Note that beyond sparsity, no assumption is made (in particular regarding correlations between input variables), and we obtain a high-dimensional phenomenon where d may be much larger than n.

- *Group penalties*: in this paper, we only consider ℓ_1-norm on input weights; when doing joint variable selection for all basis functions, it may be worth using a group penalty (Yuan and Lin, 2006; Bach, 2008a).

5.4 Relationship to Kernel Methods and Random Sampling

The results presented in the two sections above were using the space $\mathcal{F}_1$, with an L_1-norm on the outputs weights (and either an ℓ_1- or ℓ_2-norm on input weights). As seen in Sections 2.3 and 3.1, when using an L_2-norm on output weights, we obtain a reproducing kernel Hilbert space $\mathcal{F}_2$.

As shown in Section 6, the space $\mathcal{F}_2$ is significantly smaller than $\mathcal{F}_1$, and in particular is not adaptive to low-dimensional linear structures, which is the main advantage of the space $\mathcal{F}_1$. However, algorithms for $\mathcal{F}_2$ are significantly more efficient, and there is no need for the conditional gradient algorithms presented in Section 2.5. The first possibility is to use the usual RKHS representer theorem with the kernel functions computed in Section 3.1, leading to a computation complexity of $O(n^2)$. Alternatively, as shown by Rahimi and Recht (2007), one may instead sample m basis functions that is m different hidden units, keep the input weights fixed and optimize only the output layer with a squared ℓ_2-penalty. This will quickly (i.e., the error goes down as $1/\sqrt{m}$) approach the non-parametric estimator based on penalizing by the RKHS norm γ_2. Note that this argument of random sampling has been used to study approximation bounds for neural networks with finitely many units (Maiorov and Meir, 2000).

Given the usage of random sampling with L_2-penalties, it is thus tempting to sample weights, but now optimize an ℓ_1-penalty, in order to get the non-parametric estimator obtained from penalizing by γ_1. When the number of samples m tends to infinity, we indeed obtain an approximation that converges to γ_1 (this is simply a uniform version of the law of large numbers). However, the rate of convergence does depend on the dimension d, and in general exponentially many samples would be needed for a good approximation—see Bach (2013, Section 6) for a more precise statement in the related context of convex matrix factorizations.

5.5 Sufficient Condition for Polynomial-time Algorithms

In order to preserve the generalization bounds presented above, it is sufficient to be able to solve the following problem, for any $y \in \mathbb{R}^n$ and $z_1, \dots, z_n \in \mathbb{R}^{d+1}$:

$$\sup_{\|v\|_p=1} \left| \frac{1}{n} \sum_{i=1}^n y_i (v^\top z_i)_+^\alpha \right|, \tag{5}$$

up to a constant factor. That is, there exists $\kappa \geqslant 1$, such that for all y and z, we may compute $\hat{v}$ such that $\|\hat{v}\|_p = 1$ and

$$\left| \frac{1}{n} \sum_{i=1}^n y_i (\hat{v}^\top z_i)_+^\alpha \right| \geqslant \frac{1}{\kappa} \sup_{\|v\|_p=1} \left| \frac{1}{n} \sum_{i=1}^n y_i (v^\top z_i)_+^\alpha \right|.$$

This is provably NP-hard for $\alpha = 0$ (see Section 3.2), and for $\alpha = 1$ (see Section 3.3). If such an algorithm is available, the approximate conditional gradient presented in Section 2.5 leads to an estimator with the same generalization bound. Moreover, given the strong hardness results for improper learning in the situation $\alpha = 0$ (Klivans and Sherstov, 2006; Livni et al., 2014), a convex relaxation that would consider a larger set of predictors (e.g., by relaxing $vv^\top$ into a symmetric positive-definite matrix), and obtained a constant approximation guarantee, is also ruled out.

However, this is only a sufficient condition, and a simpler sufficient condition may be obtained. In the following, we consider $\mathcal{V} = \{v \in \mathbb{R}^{d+1}, \ \|v\|_2 = 1\}$ and basis functions $\varphi_v(z) = (v^\top z)_+^\alpha$ (that is we specialize to the ℓ_2-norm penalty on weight vectors). We consider a new variation norm $\hat{\gamma}_1$ which has to satisfy the following assumptions:

- *Lower-bound on* γ_1: It is defined from functions $\hat{\varphi}_{\hat{v}}$, for $\hat{v} \in \hat{\mathcal{V}}$, where for any $v \in \mathcal{V}$, there exists $\hat{v} \in \hat{\mathcal{V}}$ such that $\varphi_v = \hat{\varphi}_{\hat{v}}$. This implies that the corresponding space $\hat{\mathcal{F}}_1$ is larger than $\mathcal{F}_1$ and that if $f \in \mathcal{F}_1$, then $\hat{\gamma}_1(f) \leqslant \gamma_1(f)$.

- *Polynomial-time algorithm for dual norm*: The dual norm $\sup_{\hat{v} \in \hat{\mathcal{V}}} \left| \frac{1}{n} \sum_{i=1}^n y_i \hat{\varphi}_{\hat{v}}(z_i) \right|$ may be computed in polynomial time.

- *Performance guarantees for random direction*: There exists $\kappa > 0$, such that for any vectors $z_1, \dots, z_n \in \mathbb{R}^{d+1}$ with ℓ_2-norm less than R, and random standard Gaussian vector $y \in \mathbb{R}^n$,

$$\sup_{\hat{v} \in \hat{\mathcal{V}}} \left| \frac{1}{n} \sum_{i=1}^n y_i \hat{\varphi}_{\hat{v}}(x_i) \right| \leqslant \kappa \frac{R}{\sqrt{n}}. \tag{6}$$

  We may also replace the standard Gaussian vectors by Rademacher random variables.

We can then penalize by $\hat{\gamma}$ instead of γ. Since $\hat{\gamma}_1 \leqslant \gamma_1$, approximation properties are transferred, and because of the result above, the Rademacher complexity for $\hat{\gamma}_1$-balls scales as well as for γ_1-balls. In the next section, we show convex relaxations which cannot achieve these and leave the existence or non-existence of such norm $\hat{\gamma}_1$ as an open problem.

6. Convex Relaxations of the Frank-Wolfe Step

In this section, we provide approximation algorithms for the following problem of maximizing, for a given $y \in \mathbb{R}^n$ and vectors $z_1, \dots, z_n$:

$$\sup_{\|v\|_p=1} \frac{1}{n} \sum_{i=1}^n y_i (v^\top z_i)_+^\alpha$$

These approximation algorithms may be divided in three families, as they may be based on (a) geometric interpretations as linear binary classification or computing Haussdorff distances (see Section 3.2 and Section 3.3), (b) on direct relaxations, on (c) relaxations of sign vectors. For simplicity, we only focus on the case $p = 2$ (that is ℓ_2-constraint on weights) and on $\alpha = 1$ (rectified linear units). As described in Section 5.5, constant-factor approximation ratios are not possible, while approximation ratios that increases with n are possible (but as of now, we only obtain scalings in n that provide a provable sample complexity with a polynomial time algorithm which is exponential in the dimension d.

6.1 Semi-definite Programming Relaxations

We present two relaxations, which are of the form described in Section 5.5 (leading to potential generalization bounds) but do not attain the proper approximation scaling (as was checked empirically).

Note that all relaxations that end up being Lipschitz-continuous functions of z, will have at least the same scaling than the set of these functions. The Rademacher complexity of such functions is well-known, that is $1/\sqrt{n}$ for $d = 1$, $\sqrt{\frac{\log n}{n}}$ for $d = 2$ and $n^{-1/d}$ for larger d (von Luxburg and Bousquet, 2004). Unfortunately, the decay in n is too slow to preserve generalization bounds (which would require a scaling in $1/\sqrt{n}$).

d-dimensional relaxation. We denote $u_i = (v^\top z_i)_+ = \frac{1}{2} v^\top z_i + \frac{1}{2} |v^\top z_i|$. We may then use $2u_i - v^\top z_i = |v^\top z_i|$ and, for $\|v\|_2 = 1$, $\|vv^\top z_i\|_2 = |v^\top z_i| = \sqrt{z_i^\top vv^\top z_i}$. By denoting $V = vv^\top$, the constraint that $u_i = (v^\top z_i)_+ = \frac{1}{2} v^\top z_i + \frac{1}{2} |v^\top z_i|$ is equivalent to

$$\|Vz_i\|_2 \leqslant 2u_i - v^\top z_i \leqslant \sqrt{z_i^\top V z_i} \ \text{ and } \ V \succcurlyeq 0, \ \operatorname{tr} V = 1, \ \operatorname{rank}(V) = 1.$$

We obtain a convex relaxation when removing the rank constraint, that is

$$\sup_{V \succcurlyeq 0, \ \operatorname{tr} V = 1, \ u \in \mathbb{R}^n} u^\top y \ \text{ such that } \ \forall i \in \{1, \dots, n\}, \ \|Vz_i\|_2 \leqslant 2u_i - v^\top z_i \leqslant \sqrt{z_i^\top V z_i}.$$

$(n+d)$-dimensional relaxation. We may go further by also considering quadratic forms in $u \in \mathbb{R}^n$ defined above. Indeed, we have:

$$(2u_i - v^\top z_i)(2u_j - v^\top z_j) = |v^\top z_i| \cdot |v^\top z_j| = |v^\top z_i z_j^\top v| = |\operatorname{tr} V z_i z_j^\top|,$$

which leads to a convex program in $U = uu^\top$, $V = vv^\top$ and $J = uv^\top$, that is a semidefinite program with $d + n$ dimensions, with the constraints

$$4U_{ij} + x_j^\top V z_i - 2\delta_i^\top J z_j - 2\delta_j^\top J z_i \geqslant |\operatorname{tr} V z_i z_j^\top|,$$

and the usual semi-definite contraints $\begin{pmatrix} U & J \\ J^\top & V \end{pmatrix} \succcurlyeq \begin{pmatrix} u \\ v \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix}^\top$, with the additional constraint that $4U_{ii} + z_i^\top V z_i - 4\delta_i^\top J z_i = \operatorname{tr} V z_i z_i^\top$.

If we add these constraints on top of the ones above, we obtain a tighter relaxation. Note that for this relaxation, we must have $[(2u_i - v^\top z_i) - (2u_j - v^\top z_j)]$ less than a constant times $\|z_i - z_j\|_2$. Hence, the result mentioned above regarding Lipschitz-continuous functions and the scaling of the upper-bound for random y holds (with the dependence on n which is not good enough to preserve the generalization bounds with a polynomial-time algorithm).

6.2 Relaxation of Sign Vectors

By introducing a sign vector $s \in \mathbb{R}^n$ such that $s_i \in \{-1, 1\}$ and $s_i v^\top x_i = |v^\top x_i|$, we have the following relaxation with $S = ss^\top$, $V = vv^\top$ and $J = sv^\top$:

- Usual semi-definite constraint: $\begin{pmatrix} S & J \\ J^\top & V \end{pmatrix} \succcurlyeq \begin{pmatrix} s \\ v \end{pmatrix}\begin{pmatrix} s \\ v \end{pmatrix}^\top$,
- Unit/trace constraints: $\operatorname{diag}(S) = 1$ and $\operatorname{tr} V = 1$,
- Sign constraint: $\delta_i^\top J x_i \geqslant \max_{j \neq i} |\delta_j^\top J x_i|$.
- Additional constraint: $(x_i^\top V x_i)^{1/2} \leqslant \delta_i^\top J x_i$.

We then need to maximize $\frac{1}{2n}\sum_{i=1}^n y_i \delta_i^\top J x_i + \frac{1}{2n}\sum_{i=1}^n y_i v^\top x_i$, which leads to a semidefinte program. Again empirically, it did not lead to the correct scaling as a function of n for random Gaussian vectors $y \in \mathbb{R}^n$.

7. Conclusion

In this paper, we have provided a detailed analysis of the generalization properties of convex neural networks with positively homogenous non-decreasing activation functions. Our main new result is the adaptivity of the method to underlying linear structures such as the dependence on a low-dimensional subspace, a setting which includes non-linear variable selection in presence of potentially many input variables.

All our current results apply to estimators for which no polynomial-time algorithm is known to exist and we have proposed sufficient conditions under which convex relaxations could lead to the same bounds, leaving open the existence or non-existence of such algorithms. Interestingly, these problems have simple geometric interpretations, either as binary linear classification, or computing the Haussdorff distance between two zonotopes.

In this work, we have considered a single real-valued output; the functional analysis framework readily extends to outputs in a finite-dimensional vector-space where vector-valued measures could be used, and then apply to multi-task or multi-class problems. However, the extension to multiple hidden layers does not appear straightforward as the units of the last hidden layers share the weights of the first hidden layers, which should require a new functional analysis framework.

Acknowledgements

The author was partially supported by the European Research Council (SIERRA Project), and thanks Nicolas Le Roux for helpful discussions. The author also thanks Varun Kanade for pointing the NP-hardness linear classification result. The main part of this work was carried through while visiting the Centre de Recerca Matemàtica (CRM) in Barcelona.

Appendix A. Reproducing Kernel Hilbert Spaces for ℓ_2-norm Penalization

In this section, we consider a Borel probability measure τ on the compact space $\mathcal{V}$, and functions $\varphi_v : \mathcal{X} \to \mathbb{R}$ such that the functions $v \mapsto \varphi_v(x)$ are measurable for all $x \in \mathcal{X}$. We study the set $\mathcal{F}_2$ of functions f such that there exists a squared-integrable function $p : \mathcal{X} \to \mathbb{R}$ with $f(x) = \int_{\mathcal{V}} p(v)\varphi_v(x) d\tau(v)$ for all $x \in \mathcal{X}$. For $f \in \mathcal{F}_2$, we define $\gamma_2^2(f)$ as the infimum of $\int_{\mathcal{V}} p(v)^2 d\tau(v)$ over all decompositions of f. We now show that $\mathcal{F}_2$ is an RKHS with kernel $k(x, y) = \int_{\mathcal{V}} \varphi_v(x)\varphi_v(y) d\tau(v)$.

We follow the proof of Berlinet and Thomas-Agnan (2004, Section 4.1) and extend it to integrals rather than finite sums. We consider the linear mapping $T : L_2(d\tau) \to \mathcal{F}_2$ defined by $(Tp)(x) = \int_{\mathcal{V}} p(v)\varphi_v(x) d\tau(v)$, with null space $\mathcal{K}$. When restricted to the orthogonal complement $\mathcal{K}^\perp$, we obtain a bijection U from $\mathcal{K}^\perp$ to $\mathcal{F}_2$. We then define a dot-product on $\mathcal{F}_2$ as $\langle f, g\rangle = \int_{\mathcal{V}} (U^{-1}f)(v)(U^{-1}g)(v) d\tau(v)$.

We first show that this defines an RKHS with kernel k defined above. For this, we trivially have $k(\cdot, y) \in \mathcal{F}_2$ for all $y \in \mathcal{X}$. Moreover, for any $y \in \mathcal{X}$, we have with $p = U^{-1}k(\cdot, y) \in \mathcal{K}^\perp$ and $q : v \mapsto \varphi_v(y)$, $p - q \in \mathcal{K}$, which implies that $\langle f, k(\cdot, y)\rangle = \int_{\mathcal{V}} (U^{-1}f)(v) p(v) d\tau(v) = \int_{\mathcal{V}} (U^{-1}f)(v) q(v) d\tau(v) = \int_{\mathcal{V}} (U^{-1}f)(v)\varphi_v(y) d\tau(v) = T(U^{-1}f)(y) = f(y)$, hence the reproducing property is satisfied. Thus, $\mathcal{F}_2$ is an RKHS.

We now need to show that the RKHS norm which we have defined is actually γ_2. For any $f \in \mathcal{F}_2$ such that $f = Tp$, for $p \in L_2(d\tau)$, we have $p = U^{-1}f + q$, where $q \in \mathcal{K}$. Thus, $\int_{\mathcal{V}} p(v)^2 d\tau(v) = \|p\|^2_{L_2(d\tau)} = \|U^{-1}f\|^2_{L_2(d\tau)} + \|q\|^2_{L_2(d\tau)} = \|f\|^2 + \|q\|^2_{L_2(d\tau)}$. This implies that $\int_{\mathcal{V}} p(v)^2 d\tau(v) \geqslant \|f\|^2$ with equality if and only if $q = 0$. This shows that $\gamma_2(f) = \|f\|$.

Appendix B. Approximate Conditional Gradient with Multiplicative Oracle

In this section, we wish to minimize a smooth convex functional $J(h)$ on for h in a Hilbert-space over a norm ball $\{\gamma(h) \leqslant \delta\}$. A multiplicative approximate oracle outputs for any $g \in \mathbb{R}^n$, $\hat{h}$ such that $\gamma(\hat{h}) = 1$, and

$$\langle \hat{h}, g\rangle \leqslant \max_{\gamma(h) \leqslant 1} \langle h, g\rangle \leqslant \kappa \langle \hat{h}, g\rangle,$$

for a fixed $\kappa \geqslant 1$. We now propose a modification of the conditional gradient algorithm that converges to a certain h such that $\gamma(h) \leqslant \delta$ and for which $\inf_{\gamma(h) \leqslant \delta} J(h) \leqslant J(\hat{h}) \leqslant \inf_{\gamma(h) \leqslant \delta/\kappa} J(h)$.

We assume the smoothness of the function J with respect to the norm γ, that is, for a certain $L > 0$, for all h, h' such that $\gamma(h) \leqslant \delta$, then

$$J(h') \leqslant J(h) + \langle J'(h), h' - h \rangle + \frac{L}{2}\gamma(h - h')^2. \tag{7}$$

We consider the following recursion

$$\begin{aligned} \hat{h}_t &= -\delta \times \text{ output of approximate oracle at } -J'(h_t) \\ h_{t+1} &\in \arg\min_{\rho \in [0,1]} J((1-\rho)h_t + \rho \hat{h}_t). \end{aligned}$$

In the previous recursion, one may replace the minimization of J on the segment $[h_t, \hat{h}_t]$ with the minimization of its upper-bound of Eq. (7) taken at $h = h_t$. From the recursion, all iterates are in the γ-ball of radius δ. Following the traditional convergence proof for the conditional gradient method (Dunn and Harshbarger, 1978; Jaggi, 2013), we have, for any ρ in $[0, 1]$:

$$\begin{aligned} J(h_{t+1}) &\leqslant J(h_t) - \rho \langle J'(h_t), h_t - \hat{h}_t \rangle + 2L\rho^2\delta^2 \\ &= J(h_t) - \rho J'(h_t)^\top h_t + \kappa \langle J'(h_t), \frac{\hat{h}_t}{\kappa} \rangle + 2L\rho^2\delta^2 \\ &\leqslant J(h_t) - \rho J'(h_t)^\top h_t - \max_{\gamma(h) \leqslant \delta/\kappa} \{ - \langle J'(h_t), h \rangle \} + 2L\rho^2\delta^2. \end{aligned}$$

If we take h_* the minimizer of J on $\{\gamma(h) \leqslant \delta/\kappa\}$, we get:

$$J(h_{t+1}) \leqslant J(h_t) - \rho \langle J'(h_t), h_t - h_* \rangle + 2L\rho^2\delta^2.$$

Then, by using $J(h_t) \geqslant J(h_*) + \langle J'(h_t), h_* - h_t \rangle$, we get:

$$J(h_{t+1}) - J(h_*) \leqslant (1 - \rho)[J(h_t) - J(h_*)] + 2L\rho^2\delta^2.$$

This is valid for any $\rho \in [0, 1]$. If $J(h_t) - J(h_*) \leqslant 0$ for some t, then by taking $\rho = 0$ it remains the same of all greater t. Therefore, up to (the potentially never happening) point where $J(h_t) - J(h_*) \leqslant 0$, we can apply the regular proof of the conditional gradien to obtain: $J(h_t) \leqslant \inf_{\gamma(h) \leqslant \delta/\kappa} J(h) + \frac{4L\rho^2\delta^2}{t}$, which leads to the desired result. Note that a similar reasoning may be used for $\rho = 2/(t + 1)$.

Appendix C. Proofs for the 2-dimensional Sphere ($d = 1$)

In this section, we consider only the case $d = 1$, where the sphere $\mathbb{S}^d$ is isomorphic to $[0, 2\pi]$ (with periodic boundary conditions). We may then compute the norm γ_2 in closed form. Indeed, if we can decompose g as $g(\theta) = \frac{1}{2\pi}\int_0^{2\pi} p(\varphi)\sigma(\cos(\varphi - \theta))d\varphi$, then the decomposition of g into the k-th frequency elements (the combination of the two k-th elements of the Fourier series) is equal to, for $\sigma(u) = (u)_+^\alpha$, and for $k > 0$:

$$\begin{aligned} g_k(\theta) &= \frac{1}{\pi}\int_0^{2\pi} g(\eta)\cos k(\theta - \eta) d\eta \\ &= \frac{1}{\pi}\int_0^{2\pi} \frac{1}{2\pi}\Big(\int_0^{2\pi} p(\varphi)\sigma(\cos(\eta - \varphi))d\varphi\Big)\cos k(\theta - \eta) d\eta \text{ through the decomposition of } g, \\ &= \frac{1}{2\pi^2}\int_0^{2\pi} p(\varphi)\Big(\int_0^{2\pi} \sigma(\cos(\eta - \varphi))\cos k(\theta - \eta) d\eta\Big) d\varphi \\ &= \frac{1}{2\pi^2}\int_0^{2\pi} p(\varphi)\Big(\int_0^{2\pi} \sigma(\cos\eta)\cos k(\theta - \varphi - \eta) d\eta\Big) d\varphi \text{ by a change of variable,} \\ &= \frac{1}{2\pi^2}\int_0^{2\pi} p(\varphi)\Big(\cos k(\theta - \varphi)\int_0^{2\pi}\sigma(\cos\eta)\cos k\eta\, d\eta \\ &\qquad + \sin k(\theta - \varphi)\int_0^{2\pi}\sigma(\cos\eta)\sin k\eta\, d\eta\Big) d\varphi \text{ by expanding the cosine,} \\ &= \Big(\frac{1}{2\pi}\int_0^{2\pi}\sigma(\cos\eta)\cos k\eta\, d\eta\Big)\Big(\frac{1}{\pi}\int_0^{2\pi} p(\varphi)\cos k(\theta - \varphi)\Big) + 0 \text{ by a parity argument,} \\ &= \lambda_k p_k(\theta) \text{ with } \lambda_k = \frac{1}{2\pi}\int_0^{2\pi}\sigma(\cos\eta)\cos k\eta\, d\eta. \end{aligned}$$

For $k = 0$, the same equality holds (except that the two coefficients g_0 and p_0 are divided by 2π except of π).

Thus we may express $\|p\|^2_{L_2(\mathbb{S}^d)}$ as

$$\begin{aligned} \|p\|^2_{L_2(\mathbb{S}^d)} &= \sum_{k \geqslant 0}\|p_k\|^2_{L_2(\mathbb{S}^d)} = \sum_{\lambda_k \neq 0}\|p_k\|^2_{L_2(\mathbb{S}^d)} + \sum_{\lambda_k = 0}\|p_k\|^2_{L_2(\mathbb{S}^d)} \\ &= \sum_{\lambda_k \neq 0}\frac{1}{\lambda_k^2}\|g_k\|^2_{L_2(\mathbb{S}^d)} + \sum_{\lambda_k = 0}\|p_k\|^2_{L_2(\mathbb{S}^d)}. \end{aligned}$$

If we minimize over p, we thus need to have $\|p_k\|^2_{L_2(\mathbb{S}^d)} = 0$ for $\lambda_k = 0$, and we get

$$\gamma_2(g)^2 = \sum_{\lambda_k \neq 0}\frac{1}{\lambda_k^2}\|g_k\|^2_{L_2(\mathbb{S}^d)}. \tag{8}$$

We thus simply need to compute λ_k and its decay for all values of α, and then relate them to the smoothness properties of g, which is standard for Fourier series.

C.1 Computing λ_k

We now detail the computation of $\lambda_k = \frac{1}{2\pi}\int_0^{2\pi}\sigma(\cos\eta)\cos k\eta\, d\eta$ for the different functions $\sigma = (\cdot)_+^\alpha$. We have for $\alpha = 0$:

$$\frac{1}{2\pi}\int_0^{2\pi} 1_{\cos\eta \geqslant 0}\cos k\eta\, d\eta = \frac{1}{2\pi}\int_{-\pi/2}^{\pi/2}\cos k\eta\, d\eta = \frac{1}{\pi k}\sin\frac{k\pi}{2} \text{ if } k \neq 0.$$

For $k = 0$ it is equal to $\frac{1}{2}$. It is equal to zero for all other even k, and different from zero for all odd k, with λ_k going to zero as $1/k$.

We have for $\alpha = 1$:

$$\frac{1}{2\pi}\int_0^{2\pi}(\cos\eta)_+\cos k\eta\, d\eta = \frac{1}{2\pi}\int_{-\pi/2}^{\pi/2}\cos\eta\cos k\eta\, d\eta$$

$$
\begin{aligned}
&= \frac{1}{2\pi}\int_{-\pi/2}^{\pi/2}[\frac{1}{2}\cos(k+1)\eta + \frac{1}{2}\cos(k-1)\eta]\,d\eta \\
&= \frac{1}{4\pi}\left(\frac{2}{k+1}\sin(k+1)\frac{\pi}{2} + \frac{2}{k-1}\sin(k-1)\frac{\pi}{2}\right) \\
&= \frac{\cos\frac{k\pi}{2}}{2\pi}\left(\frac{1}{k+1} - \frac{1}{k-1}\right) = \frac{-\cos\frac{k\pi}{2}}{\pi(k^2-1)} \text{ for } k \neq 1.
\end{aligned}
$$

For $k=1$, it is equal to $1/4$. It is equal to zero for all other odd k, and different from zero for all even k, with λ_k going to zero as $1/k^2$.

For $\alpha = 2$, we have:

$$
\begin{aligned}
\frac{1}{2\pi}\int_0^{2\pi}(\cos\eta)_+^2\cos k\eta\,d\eta &= \frac{1}{2\pi}\int_{-\pi/2}^{\pi/2}(\cos\eta)^2\cos k\eta\,d\eta = \frac{1}{2\pi}\int_{-\pi/2}^{\pi/2}\frac{1+\cos 2\eta}{2}\cos k\eta\,d\eta \\
&= \frac{1}{2\pi}\int_{-\pi/2}^{\pi/2}[\frac{1}{2}\cos k\eta + \frac{1}{4}\cos(k+2)\eta + \frac{1}{4}\cos(k-2)\eta]\,d\eta \\
&= \frac{1}{4\pi}\left(\frac{2}{k}\sin k\frac{\pi}{2} + \frac{1}{k+2}\sin(k+2)\frac{\pi}{2} + \frac{1}{k-2}\sin(k-2)\frac{\pi}{2}\right) \\
&= \frac{\sin(k\frac{\pi}{2})}{4\pi}\left(\frac{2}{k} - \frac{1}{k+2} - \frac{1}{k-2}\right) \\
&= \frac{\sin(k\frac{\pi}{2})}{4\pi}\left(\frac{2k^2-8-k^2+2k-k^2-2k}{k(k^2-4)}\right) \\
&= \frac{-8\sin(k\frac{\pi}{2})}{4\pi k(k^2-4)} \text{ for } k \notin \{0,2\}.
\end{aligned}
$$

For $k=0$, it is equal to $1/4$, and for $k=2$, it is equal to $1/8$. It is equal to zero for all other even k, and different from zero for all odd k, with λ_k going to zero as $1/k^3$.

The general case for $\alpha \geqslant 2$ will be shown for for all d in Appendix D.2: for all $\alpha \in \mathbb{N}$, λ_k is different from zero for k having the opposite parity of α, with a decay as $1/k^{\alpha+1}$. All values from $k=0$ to α are also different from zero. All larger values with the same parity as α are equal to zero.

C.2 Proof of Prop. 2 for $d = 1$

We only consider the proof for $d=1$. For the proof for general d, see Appendix D.3.

Given the zero values of λ_k given above, if g has the opposite parity than α (that is, is even when α is odd, and vice-versa), then we may define p through its Fourier series, which is obtained by multiplying the one of g by a strictly positive sequence growing as $k^{\alpha+1}$.

Thus, if g is such that its $(\alpha+1)$-th order derivative is squared-integrable, then p defined above is squared-integrable, that is, $g \in \mathcal{G}_2$. Moreover, if all derivatives of order less than $(\alpha+1)$ are bounded by η, p is squared-integrable and $\|p\|^2_{L_2(\mathbb{S}^d)}$ is upper-bounded by a constant times η^2, i.e., $\gamma_2(g)^2 \leqslant C(\alpha)^2\eta^2$.

Note that we could relax the assumption that g is even (resp. odd) by adding all trigonometric polynomials of order less than α.

C.3 Proof of Prop. 3 for $d = 1$

Again, we only consider the proof for $d=1$. For the proof for general d, see Appendix D.4.

Without loss of generality, we assume that $\eta = 1$. For $d=1$, we essentially want to approximate a Lipschitz-continuous function by a function which is $(\alpha+1)$-times differentiable.

For $\alpha = 0$, then the function g is already in $\mathcal{G}_2$ with a norm less than one, because Lipschitz-continuous functions are almost everywhere differentiable with bounded derivative (Adams and Fournier, 2003). We thus now consider $\alpha > 0$.

Given λ_k defined above and $r \in (0,1)$, we define $\hat{p}$ through

$$
\hat{p}_k(\theta) = \sum_{k,\lambda_k \neq 0} \lambda_k^{-1} g_k(\theta) r^k.
$$

Our goal is to show that for r chosen close enough to 1, then the function $\hat{g}$ defined from $\hat{p}$ has small enough norm $\gamma_2(\hat{g}) \leqslant \|\hat{p}\|_{L_2(\mathbb{S}^d)}$, and is close to g.

Computation of norm. We have

$$
\|\hat{p}\|^2_{L_2(\mathbb{S}^d)} = \sum_{k,\lambda_k\neq 0} \lambda_k^{-2} r^{2k} \|g_k\|^2_{L_2(\mathbb{S}^d)}.
$$

Since g is 1-Lipschitz-continuous with constant 1, then it has a squared-integrable derivative $f = g'$ with norm less than 1 (Adams and Fournier, 2003), so that

$$
\|f\|^2_{L_2(\mathbb{S}^d)} = \sum_{k\geqslant 0} \|f_k\|^2_{L_2(\mathbb{S}^d)} \leqslant 1.
$$

This implies that using $\lambda_k^{-1} = O(k^{\alpha+1})$:

$$
\|\hat{p}\|^2_{L_2(\mathbb{S}^d)} \leqslant \lambda_0^{-2}\|g_0\|^2_{L_2(\mathbb{S}^d)} + \|g'\|^2_{L_2(\mathbb{S}^d)} \max_{k\geqslant 1,\lambda_k\neq 0} r^{2k}\lambda_k^{-2}k^{-2} \leqslant C + C\|g'\|^2_{L_2(\mathbb{S}^d)} \max_{k\geqslant 1} r^{2k}k^{2\alpha},
$$

because $\|g_0\|^2_{L_2(\mathbb{S}^d)}$ and $\|f\|^2_{L_2(\mathbb{S}^d)}$ are bounded by 1.

We may now compute the derivative of $k \mapsto r^{2k}k^{2\alpha}$ with respect to k (now considered a real number), that is $2\alpha k^{2\alpha-1}r^{2k} + k^{2\alpha}r^{2k}2\log r$, which is equal to zero for $\frac{\alpha}{k} = \log\frac{1}{r}$, that is $k = \frac{\alpha}{\log\frac{1}{r}}$, the maximum being then $e^{-2\alpha}(\frac{\alpha}{\log\frac{1}{r}})^{2\alpha} = O((1-r)^{-2\alpha})$, by using the concavity of the logarithm. Thus $\|\hat{p}\|_{L_2(\mathbb{S}^d)} \leqslant C(1-r)^{-\alpha}$. This defines $\hat{g}$ with $\gamma(\hat{g}) \leqslant C(1-r)^{-\alpha}$.

Computing distance between $\hat{g}$ and g. We have:

$$
\begin{aligned}
\hat{g}(\theta) &= \sum_{k\geqslant 0} g_k(\theta)r^k = \sum_{k>0}\frac{1}{\pi}\int_0^{2\pi} g(\eta)r^k\cos k(\theta-\eta)d\eta + \frac{1}{2\pi}\int_0^{2\pi} g(\eta)d\eta \\
&= \frac{1}{\pi}\int_0^{2\pi}\left(\sum_{k\geqslant 0} r^k\cos k(\theta-\eta)\right)g(\eta)d\eta - \frac{1}{2\pi}\int_0^{2\pi} g(\eta)d\eta \\
&= \frac{1}{\pi}\int_0^{2\pi}\mathrm{Real}\left(\frac{1}{1-re^{i(\theta-\eta)}}\right)g(\eta)d\eta - \frac{1}{2\pi}\int_0^{2\pi} g(\eta)d\eta
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{\pi}\int_0^{2\pi}\Big(\frac{1-r\cos(\theta-\eta)}{(1-r\cos(\theta-\eta))^2+r^2(\sin(\theta-\eta))^2}\Big)g(\eta)d\eta - \frac{1}{2\pi}\int_0^{2\pi} g(\eta)d\eta \\
&= \frac{1}{\pi}\int_0^{2\pi}\Big(\frac{1-r\cos(\theta-\eta)}{1+r^2-2r\cos(\theta-\eta)}\Big)g(\eta)d\eta - \frac{1}{2\pi}\int_0^{2\pi} g(\eta)d\eta \\
&= \frac{1}{2\pi}\int_0^{2\pi}\Big(\frac{1-r^2+1+r^2-2r\cos(\theta-\eta)}{1+r^2-2r\cos(\theta-\eta)}\Big)g(\eta)d\eta - \frac{1}{2\pi}\int_0^{2\pi} g(\eta)d\eta \\
&= \frac{1}{2\pi}\int_0^{2\pi}\Big(\frac{1-r^2}{1+r^2-2r\cos(\theta-\eta)}\Big)g(\eta)d\eta.
\end{aligned}
$$

We have, for any $\theta \in [0, 2\pi]$

$$
\begin{aligned}
|\hat{g}(\theta) - g(\theta)| &= \Big|\frac{1}{2\pi}\int_0^{2\pi}\Big(\frac{1-r^2}{1+r^2-2r\cos(\theta-\eta)}\Big)[g(\eta)-g(\theta)]d\eta\Big| \\
&\leqslant \frac{1}{2\pi}\int_0^{2\pi}\Big(\frac{1-r^2}{1+r^2-2r\cos(\theta-\eta)}\Big)|g(\eta)-g(\theta)|d\eta \\
&= \frac{1}{2\pi}\int_\pi^{\pi}\Big(\frac{1-r^2}{1+r^2-2r\cos\eta}\Big)|g(\theta)-g(\theta+\eta)|d\eta \ \text{ by periodicity,} \\
&= \frac{1}{\pi}\int_{\pi/2}^{\pi/2}\Big(\frac{1-r^2}{1+r^2-2r\cos\eta}\Big)|g(\theta)-g(\theta+\eta)|d\eta \ \text{ by parity of } g, \\
&\leqslant \frac{1}{\pi}\int_{\pi/2}^{\pi/2}\Big(\frac{1-r^2}{1+r^2-2r\cos\eta}\Big)\sqrt{2}|\sin\eta|d\eta \\
&\qquad \text{because the distance on the sphere is bounded by the sine,} \\
&\leqslant \frac{2}{\pi}\int_0^{\pi}\Big(\frac{1-r^2}{1+r^2-2r\cos\eta}\Big)\sin\eta\, d\eta \\
&= \frac{1}{\pi}\int_0^{1}\Big(\frac{1-r^2}{1+r^2-2rt}\Big)dt \ \text{ by the change of variable } t=\cos\theta, \\
&\leqslant C(1-r)\int_0^1\Big(\frac{1}{1+r^2-2rt}\Big)dt \\
&= C(1-r)\Big[\frac{-1}{2r}\log(1+r^2-2rt)\Big]_0^1 = C(1-r)\frac{1}{2r}\log\frac{1+r^2}{(1-r)^2}.
\end{aligned}
$$

It can be easily checked that for any $r \in (1/2, 1)$, the last function is less than a constant times $\frac{5}{2}C(1-r)\log\frac{1}{1-r}$. We thus get for δ large enough, by taking $r = 1-(C/\delta)^{1/\alpha} \in (1/2, 1)$, an error of

$$(C/\delta)^{1/\alpha}\log(C/\delta)^{-1/\alpha} = O(\delta^{-1/\alpha}\log\delta).$$

This leads to the desired result.

Appendix D. Approximations on the d-dimensional Sphere

In this appendix, we first review tools from spherical harmonic analysis, before proving the two main propositions regarding the approximation properties of the Hilbert space $\mathcal{G}_2$. Using spherical harmonics in our set-up is natural and is common in the analysis of ridge functions (Petrushev, 1998) and zonotopes (Bourgain and Lindenstrauss, 1988).

D.1 Review of Spherical Harmonics Theory

In this section, we review relevant concepts from spherical harmonics. See Frye and Efthimiou (2012); Atkinson and Han (2012) for more details. Spherical harmonics may be seen as extension of Fourier series to spheres in dimensions more than 2 (i.e., with our convention $d \geqslant 1$).

For $d \geqslant 1$, we consider the sphere $\mathbb{S}^d = \{x \in \mathbb{R}^{d+1},\ \|x\|_2 = 1\} \subset \mathbb{R}^{d+1}$, as well as its normalized rotation-invariant measure τ_d (with mass 1). We denote by $\omega_d = \frac{2\pi^{(d+1)/2}}{\Gamma((d+1)/2)}$ the surface area of the sphere $\mathbb{S}^d$.

Definition and links with Laplace-Beltrami operator. For any $k \geqslant 1$ (for $k = 0$, the constant function is the corresponding basis element), there is an orthonormal basis of spherical harmonics, $Y_{kj} : \mathbb{S}^d \to \mathbb{R}$, $1 \leqslant j \leqslant N(d,k) = \frac{2k+d-1}{k}\binom{k+d-2}{d-1}$. They are such $\langle Y_{ki}, Y_{si}\rangle_{\mathbb{S}^d} = \int_{\mathbb{S}^d} Y_{ki}(x)Y_{sj}d\tau_d(x) = \delta_{ij}\delta_{sk}$.

Each of these harmonics may be obtained from homogeneous polynomials in $\mathbb{R}^d$ with an Euclidean Laplacian equal to zero, that is, if we define a function $H_k(y) = Y_{ki}(y/\|y\|_2)\|y\|_2^k$ for $y \in \mathbb{R}^{d+1}$, then H_k is a homogeneous polynomial of degree k with zero Laplacian. From the relationship between the Laplacian in $\mathbb{R}^{d+1}$ and the Laplace-Beltrami operator Δ on $\mathbb{S}^d$, Y_{ki} is an eigenfunction of Δ with eigenvalue $-k(k+d-1)$. Like in Euclidean spaces, the Laplace-Beltrami operator may be used to characterize differentiability of functions defined on the sphere (Frye and Efthimiou, 2012; Atkinson and Han, 2012).

Legendre polynomials. We have the addition formula

$$\sum_{j=1}^{N(d,k)} Y_{kj}(x)Y_{kj}(y) = N(d,k)P_k(x^\top y),$$

where P_k is a Legendre polynomial of degree k and dimension $d+1$, defined as (Rodrigues' formula):

$$P_k(t) = (-1/2)^k\frac{\Gamma(d/2)}{\Gamma(k+d/2)}(1-t^2)^{(2-d)/2}\Big(\frac{d}{dt}\Big)^k(1-t^2)^{k+(d-2)/2}.$$

They are also referred to as Gegenbauer polynomials. For $d = 1$, P_k is the k-th Chebyshev polynomial, such that $P_k(\cos\theta) = \cos(k\theta)$ for all θ (and we thus recover the Fourier series framework of Appendix C). For $d = 2$, P_k is the usual Legendre polynomial.

The polynomial P_k is even (resp. odd) when k is even (resp. odd), and we have

$$\int_{-1}^1 P_k(t)P_j(k)(1-t^2)^{(d-2)/2}dt = \delta_{jk}\frac{\omega_d}{\omega_{d-1}}\frac{1}{N(d,k)}.$$

For small k, we have $P_0(t) = 1$, $P_1(t) = t$, and $P_2(t) = \frac{(d+1)t^2-1}{d}$.

The Hecke-Funk formula leads to, for any linear combination Y_k of Y_{kj}, $j \in \{1, \dots, N(d,k)\}$:

$$\int_{\mathbb{S}^d} f(x^\top y)Y_k(y)d\tau_d(y) = \frac{\omega_{d-1}}{\omega_d}Y_k(x)\int_{-1}^1 f(t)P_k(t)(1-t^2)^{(d-2)/2}dt.$$

Decomposition of functions in $L_2(\mathbb{S}^d)$. Any function $g : \mathbb{S}^d \to \mathbb{R}$, such that we have $\int_{\mathbb{S}^d} g(x) d\tau_d(x) = 0$ may be decomposed as

$$\begin{aligned} g(x) &= \sum_{k=1}^{\infty} \sum_{j=1}^{N(d,k)} \langle Y_{kj}, g\rangle Y_{kj}(x) = \sum_{k=1}^{\infty} \sum_{j=1}^{N(d,k)} \int_{\mathbb{S}^d} Y_{kj}(y) Y_{kj}(x) g(y) d\tau_d(y) \\ &= \sum_{k=1}^{\infty} g_k(x) \text{ with } g_k(x) = N(d,k) \int_{\mathbb{S}^d} g(y) P_k(x^\top y) d\tau_d(y). \end{aligned}$$

This is the decomposition in harmonics of degree k. Note that

$$g_1(x) = x^\top \Big[d \int_{\mathbb{S}^d} y g(y) d\tau_d(y) \Big]$$

is the linear part of g (i.e., if $g(x) = w^\top x$, $g_1 = g$). Moreover, if g does not have zero mean, we may define $g_0(x) = \int_{\mathbb{S}^d} g(y) d\tau_d(y)$ as the average value of g. Since the harmonics of different degrees are orthogonal to each other, we have the Parseval formula:

$$\|g\|^2_{\mathbb{S}^d} = \sum_{k \geqslant 0} \|g_k\|^2_{\mathbb{S}^d}.$$

Decomposition of functions of one-dimensional projections. If $g(x) = \varphi(x^\top v)$ for $v \in \mathbb{S}^d$ and $\varphi : [-1,1] \to \mathbb{R}$, then

$$\begin{aligned} g_k(x) &= N(d,k) \int_{\mathbb{S}^d} \varphi(v^\top y) P_k(x^\top y) d\tau(y) \\ &= N(d,k) \frac{\omega_{d-1}}{\omega_d} P_k(x^\top v) \int_{-1}^1 \varphi(t) P_k(t) (1-t^2)^{(d-2)/2} dt \\ &= \Big(\frac{\omega_{d-1}}{\omega_d} P_k(x^\top v) \int_{-1}^1 \varphi(t) P_k(t) (1-t^2)^{(d-2)/2} dt \Big) \sum_{j=1}^{N(d,k)} Y_{kj}(x) Y_{kj}(y), \end{aligned}$$

and

$$\begin{aligned} \|g_k\|^2_{L_2(\mathbb{S}^d)} &= \Big(\frac{\omega_{d-1}}{\omega_d} P_k(x^\top v) \int_{-1}^1 \varphi(t) P_k(t) (1-t^2)^{(d-2)/2} dt \Big)^2 \sum_{j=1}^{N(d,k)} Y_{kj}(y)^2 \\ &= \Big(\frac{\omega_{d-1}}{\omega_d} P_k(x^\top v) \int_{-1}^1 \varphi(t) P_k(t) (1-t^2)^{(d-2)/2} dt \Big)^2 N(d,k) P_k(1) \\ &= \Big(\frac{\omega_{d-1}}{\omega_d} P_k(x^\top v) \int_{-1}^1 \varphi(t) P_k(t) (1-t^2)^{(d-2)/2} dt \Big)^2 N(d,k). \end{aligned}$$

D.2 Computing the RKHS Norm γ_2

Like for the case $d = 1$, we may compute the RKHS norm γ_2 of a function g in closed form given its decomposition in the basis of spherical harmonics $g = \sum_{k \geqslant 0} g_k$. If we can decompose $g(x) = \int_{\mathbb{S}^d} p(w) \sigma(w^\top x) d\tau_d(w)$ for a certain function $p : \mathbb{S}^d \to \mathbb{R}$, then we have, for $k \geqslant 0$:

$$\begin{aligned} g_k(x) &= N(d,k) \int_{\mathbb{S}^d} g(y) P_k(x^\top y) d\tau_d(y) \\ &= N(d,k) \int_{\mathbb{S}^d} \int_{\mathbb{S}^d} p(w) \sigma(w^\top y) P_k(x^\top y) d\tau_d(y) d\tau_d(w) \\ &= N(d,k) \int_{\mathbb{S}^d} p(w) \Big(\int_{\mathbb{S}^d} \sigma(w^\top y) P_k(x^\top y) d\tau_d(y) \Big) d\tau_d(w) \\ &= \frac{\omega_{d-1}}{\omega_d} N(d,k) \int_{\mathbb{S}^d} p(w) P_k(x^\top w) \Big(\int_{-1}^1 \sigma(t) P_k(t) (1-t^2)^{(d-2)/2} dt \Big) d\tau_d(w) \\ &\qquad \text{using the Hecke-Funk formula,} \\ &= \lambda_k p_k(x) \text{ with } \lambda_k = \frac{\omega_{d-1}}{\omega_d} \int_{-1}^1 \sigma(t) P_k(t) (1-t^2)^{(d-2)/2} dt. \end{aligned}$$

If $k \equiv \alpha \bmod 2$, then $\lambda_k \propto \frac{1}{2} \int_{-1}^1 t^\alpha P_k(t) (1-t^2)^{(d-2)/2} dt = 0$, for $k > \alpha$ since P_k is orthogonal to all polynomials of degree strictly less than k for that dot-product. Otherwise, $\lambda_k \neq 0$, since t^α may be decomposed as combination with non-zero coefficients of polynomials P_j for $j \equiv \alpha \bmod 2$, $j \leqslant \alpha$.

We now provide an explicit formula extending the proof technique (for $\alpha = 1$) of Schneider (1967) and Bourgain and Lindenstrauss (1988) to all values of α. See also Mhaskar (2006).

We have, by α successive integration by parts, for $k \geqslant \alpha + 1$:

$$\begin{aligned} &\int_0^1 t^\alpha \Big(\frac{d}{dt} \Big)^k (1-t^2)^{k+(d-2)/2} dt \\ &= (-1)^\alpha \alpha! \int_0^1 \Big(\frac{d}{dt} \Big)^{k-\alpha} (1-t^2)^{k+(d-2)/2} dt = -(-1)^\alpha \alpha! \Big(\frac{d}{dt} \Big)^{k-\alpha-1} (1-t^2)^{k+(d-2)/2} \Big|_{t=0} \\ &= -(-1)^\alpha \alpha! \Big(\frac{d}{dt} \Big)^{k-\alpha-1} \sum_{j \geqslant 0} \binom{k+(d-2)/2}{j} (-1)^j t^{2j} \Big|_{t=0} \text{ using the binomial formula,} \\ &= -(-1)^\alpha \alpha! \Big(\frac{d}{dt} \Big)^{k-\alpha-1} \binom{k+(d-2)/2}{j} (-1)^j t^{2j} \Big|_{t=0} \text{ for } 2j = k-\alpha-1, \\ &= -(-1)^\alpha \alpha! \binom{k+(d-2)/2}{j} (-1)^j (2j)! \text{ for } 2j = k-\alpha-1. \end{aligned}$$

Thus

$$\begin{aligned} \lambda_k &= -\frac{\omega_{d-1}}{\omega_d} (-1/2)^k \frac{\Gamma(d/2)}{\Gamma(k+d/2)} (-1)^\alpha \alpha! \binom{k+(d-2)/2}{j} (-1)^j (2j)! \text{ for } 2j = k-\alpha-1, \\ &= -\frac{\omega_{d-1}}{\omega_d} (-1/2)^k \frac{\Gamma(d/2)}{\Gamma(k+d/2)} (-1)^\alpha \alpha! \frac{\Gamma(k+\frac{d}{2})}{\Gamma(j+1)\Gamma(k+\frac{d}{2}-j)} (-1)^j \Gamma(2j+1) \\ &= -\frac{\omega_{d-1}}{\omega_d} (-1/2)^k \frac{\Gamma(d/2)}{\Gamma(k+d/2)} (-1)^\alpha \alpha! \frac{\Gamma(k+\frac{d}{2}) (-1)^{(k-\alpha-1)/2} \Gamma(k-\alpha)}{\Gamma(\frac{k}{2}-\frac{\alpha}{2}+\frac{1}{2}) \Gamma(\frac{k}{2}+\frac{d}{2}+\frac{\alpha}{2}+\frac{1}{2})} \\ &= \frac{d-1}{2\pi} \frac{\alpha! (-1)^{(k-\alpha-1)/2}}{2^k} \frac{\Gamma(d/2)\Gamma(k-\alpha)}{\Gamma(\frac{k}{2}-\frac{\alpha}{2}+\frac{1}{2}) \Gamma(\frac{k}{2}+\frac{d}{2}+\frac{\alpha}{2}+\frac{1}{2})}. \end{aligned}$$

By using Stirling formula $\Gamma(x) \approx x^{x-1/2}e^{-x}\sqrt{2\pi}$, we get an equivalent when k or d tends to infinity as a constant (that depends on α) times

$$d^{d/2+1/2}k^{k/2-\alpha/2+1/2}(k+d)^{-k/2-d/2-\alpha/2}.$$

Note that all exponential terms cancel out. Moreover, when k tends to infinity and d is considered constant, then we get the equivalent $k^{-d/2-\alpha-1/2}$, which we need for the following sections. Finally, when d tends to infinity and k is considered constant, then we get the equivalent $d^{-\alpha/2-k/2+1/2}$.

We will also need expressions of λ_k for $k=0$ and $k=1$. For $k=0$, we have:

$$\begin{aligned}\int_0^1 t^\alpha(1-t^2)^{d/2-1}dt &= \int_0^1 (1-u)^{\alpha/2}u^{d/2-1}\frac{du}{2\sqrt{1-u}} \text{ with } t=\sqrt{1-u},\\ &= \frac{1}{2}\int_0^1 (1-u)^{\alpha/2+1/2-1}u^{d/2-1}du = \frac{1}{2}\frac{\Gamma(\alpha/2+1/2)\Gamma(d/2)}{\Gamma(\alpha/2+1/2+d/2)},\end{aligned}$$

using the normalization factor of the Beta distribution. This leads to

$$\lambda_0 = \frac{\omega_{d-1}}{\omega_d}\frac{1}{2}\frac{\Gamma(\alpha/2+1/2)\Gamma(d/2)}{\Gamma(\alpha/2+1/2+d/2)} = \frac{d-1}{2\pi}\frac{1}{2}\frac{\Gamma(\alpha/2+1/2)\Gamma(d/2)}{\Gamma(\alpha/2+1/2+d/2)},$$

which is equivalent to $d^{1/2-\alpha/2}$ as d tends to infinity.

Moreover, for $k=1$, we have (for $\alpha>0$):

$$\begin{aligned}\int_0^1 t^\alpha\Big(\frac{d}{dt}\Big)(1-t^2)^{d/2}dt &= -\alpha\int_0^1 t^{\alpha-1}(1-t^2)^{d/2}dt = -\alpha\int_0^1 (1-u)^{\alpha/2-1/2}u^{d/2}\frac{du}{2\sqrt{1-u}}\\ &= -\alpha/2\int_0^1 (1-u)^{\alpha/2-1}u^{d/2+1-1}du = -\alpha/2\frac{\Gamma(\alpha/2)\Gamma(d/2+1)}{\Gamma(\alpha/2+d/2+1)}.\end{aligned}$$

This leads to, for $\alpha>0$:

$$\lambda_1 = (-1/2)\frac{2}{d}\frac{d-1}{2\pi}(-\alpha/2)\frac{\Gamma(\alpha/2)\Gamma(d/2+1)}{\Gamma(\alpha/2+d/2+1)} = \frac{d-1}{d}\frac{\alpha}{4\pi}\frac{\Gamma(\alpha/2)\Gamma(d/2+1)}{\Gamma(\alpha/2+d/2+1)},$$

which is equivalent to $d^{-\alpha/2}$ as d tends to infinity.

Finally, for $\alpha=0$, $\lambda_1 = \frac{d-1}{2d\pi}$. More generally, we have $|\lambda_k| \sim C(d)k^{-(d-1)/2-\alpha-1}$.

Computing the RKHS norm. Given g with the correct parity, then we have

$$\gamma_2(g)^2 = \sum_{k\geqslant 0}\lambda_k^{-2}\|g_k\|^2_{L_2(\mathbb{S}^d)}.$$

D.3 Proof of Prop. 2 for $d>1$

Given the expression of λ_k from the section above, the proof is essentially the same than for $d=1$ in Appendix C.3. If g is s-times differentiable with all derivatives bounded uniformly by η, then is equal to $g=\Delta^{s/2}f$ for a certain function f such that $\|f\|_{L_2(\mathbb{S}^d)}\leqslant\eta$ (where Δ is the Laplacian on the sphere) (Frye and Efthimiou, 2012; Atkinson and Han, 2012).

Moreover, since g has the correct parity,

$$\gamma_2(g)^2 \leqslant \|p\|^2_{L_2(\mathbb{S}^d)} \leqslant \sum_{k\geqslant 1,\lambda_k\neq 0}\lambda_k^{-2}\|g_k\|^2_{L_2(\mathbb{S}^d)}$$

Also, g_k are eigenfunctions of the Laplacian with eigenvalues $k(k+d-1)$. Thus, we have

$$\|g_k\|_2^2 \leqslant \|f_k\|^2_{L_2(\mathbb{S}^d)}\frac{1}{[k(k+d-1)]^s} \leqslant \|f_k\|^2_{L_2(\mathbb{S}^d)}/k^{2s},$$

leading to $\gamma_2(g)^2 \leqslant \max_{k\geqslant 2}\lambda_k^{-2}k^{-2s}\|f\|^2_{L_2(\mathbb{S}^d)} \leqslant \max_{k\geqslant 2}k^{d-1+2\alpha+2}k^{-2s}\|f\|^2_{L_2(\mathbb{S}^d)} \leqslant C(d)\eta^2$, if $s\geqslant (d-1)/2+\alpha+1$, which is the desired result.

D.4 Proof of Prop. 3 for $d>1$

Without loss of generality we assume that $\eta=1$, and we follow the same proof as for $d=1$ in Appendix C.3. We have assumed that for all $x,y\in\mathbb{S}^d$, $|g(x)-g(y)|\leqslant\eta\|x-y\|_2 = \eta\sqrt{2}\sqrt{1-x^\top y}$. Given the decomposition in the k-th harmonics, with

$$g_k(x) = N(d,k)\int_{\mathbb{S}^d} g(y)P_k(x^\top y)d\tau_d(y),$$

we may now define, for $r<1$:

$$\hat{p}(x) = \sum_{k,\lambda_k\neq 0}\lambda_k^{-1}r^k g_k(x),$$

which is always defined when $r\in(0,1)$ because the series is absolutely convergent. This defines a function $\hat{g}$ that will have a finite γ_2-norm and be close to g.

Computing the norm. Given our assumption regarding the Lipschitz-continuity of g, we have $g=\Delta^{1/2}f$ with $f\in L_2(\mathbb{S}^d)$ with norm less than 1 (Atkinson and Han, 2012). Moreover $\|g_k\|^2_{L_2(\mathbb{S}^d)}\leqslant Ck^2\|f_k\|^2_{L_2(\mathbb{S}^d)}$. We have

$$\begin{aligned}\|\hat{p}\|^2_{L_2(\mathbb{S}^d)} &= \sum_{k,\lambda_k\neq 0}\lambda_k^{-2}r^{2k}\|g_k\|^2_{L_2(\mathbb{S}^d)}\\ &\leqslant C(d,\alpha)\max_{k\geqslant 0}k^{d-1+2\alpha}r^{2k}\|f\|^2_{L_2(\mathbb{S}^d)} \text{ because } \lambda_k=\Omega(k^{-d/2-\alpha-1/2}),\\ &\leqslant C(d,\alpha)(1-r)^{-d+1-2\alpha} \text{ (see Appendix C.3).}\end{aligned}$$

The function $\hat{p}$ thus defines a function $\hat{g}\in\mathcal{G}_1$ by $\hat{g}_k=\lambda_k p_k$, for which $\gamma_2(g)\leqslant C(d,\alpha)(1-r)^{(-d+1)/2-\alpha}$.

Approximation properties. We now show that g and $\hat{g}$ are close to each other. Because of the parity of g, we have $\hat{g}_k=r^k g_k$. We have, using Theorem 4.28 from Frye and Efthimiou (2012):

$$\hat{g}(x) = \sum_{k\geqslant 0}r^k = \sum_{k\geqslant 0}r^k N(d,k)\int_{\mathbb{S}^d} g(y)P_k(x^\top y)d\tau_d(y)$$

$$= \int_{\mathbb{S}^d} g(y)\Big(\sum_{k\geqslant 0} r^k N(d,k) P_k(x^\top y)\Big) d\tau_d(y)$$
$$= \int_{\mathbb{S}^d} g(y) \frac{1-r^2}{(1+r^2-2r(x^\top y))^{(d+1)/2}} d\tau_d(y).$$

Moreover, following Bourgain and Lindenstrauss (1988), we have:

$$g(x)-\hat{g}(x) = \int_{\mathbb{S}^d} [g(x)-g(y)] \frac{1-r^2}{(1+r^2-2r(x^\top w))^{(d+1)/2}} d\tau_d(y)$$
$$= 2\int_{\mathbb{S}^d,\ y^\top x\geqslant 0} [g(x)-g(y)] \frac{1-r^2}{(1+r^2-2r(x^\top w))^{(d+1)/2}} d\tau_d(y) \text{ by parity of } g,$$
$$|g(x)-\hat{g}(x)| \leqslant \int_{\mathbb{S}^d,\ y^\top x\geqslant 0} \sqrt{2}\sqrt{1-x^\top y} \frac{1-r^2}{(1+r^2-2r(x^\top y))^{(d+1)/2}} d\tau_d(y).$$

As shown by Bourgain and Lindenstrauss (1988, Eq. (2.13)), this is less than a constant that depends on d times $(1-r)\log\frac{1}{1-r}$. We thus get for δ large enough, by taking $1-r = (C/\delta)^{1/(\alpha+(d-1)/2)} \in (0,1)$, an error of

$$(C/\delta)^{1/(\alpha+(d-1)/2)} \log(C/\delta)^{-1/(\alpha+(d-1)/2)}] = O(\delta^{1/(\alpha+(d-1)/2)} \log\delta),$$

which leads to the desired result.

D.5 Finding Differentiable Functions which are not in $\mathcal{G}_2$

In this section, we consider functions on the sphere which have the proper parity with respect to α, which are s-times differentiable with bounded derivatives, but which are not in $\mathcal{G}_2$. We then provide optimal approximation rates for these functions.

We assume that $s-\alpha$ is even, we consider $g(x) = (w^\top x)_+^s$ for a certain arbitrary $w \in \mathbb{S}^d$. As computed at the end of Appendix D.1, we have $\|g_k\|^2_{L_2(\mathbb{S}^d)} = \Big(\frac{\omega_{d-1}}{\omega_d} P_k(x^\top v) \int_{-1}^1 \varphi(t) P_k(t)(1-t^2)^{(d-2)/2} dt\Big)^2 N(d,k)$. Given the computations from Appendix D.2, the squared norm equal to $\Big(\frac{\omega_{d-1}}{\omega_d} P_k(x^\top v) \int_{-1}^1 \varphi(t) P_k(t)(1-t^2)^{(d-2)/2} dt\Big)^2$ goes down to zero as $k^{-d-2s-1}$, while $N(d,k)$ grows as k^{d-1}. In order to use the computation of the RKHS norm derived in Appendix D.2, we need to make sure that g has the proper parity. This can de done by removing all harmonics with $k \leqslant s$ (note that these harmonics are also functions of $w^\top x$, and thus the function that we obtain is also a function of $w^\top x$). That function then has a squared RKHS norm equal to

$$\sum_{k\geqslant s, \lambda_k\neq 0} \|g_k\|^2_{L_2(\mathbb{S}^d)} \lambda_k^{-2}.$$

The summand has an asymptotic equivalent proportional to $k^{-d-2s-1} k^{d-1} k^{d+2\alpha+1}$ which is equal to $k^{d+2\alpha-2s-1}$. Thus if $d+2\alpha-2s \geqslant 0$, the series is divergent (the function is not in the RKHS), i.e., if $s \leqslant \alpha + \frac{d}{2}$.

Best approximation by a function in $\mathcal{G}_2$. The squared norm of the k-th harmonic $\|g_k\|^2_{L_2(\mathbb{S}^d)}$ goes down to zero as k^{-2s-2} and the squared RKHS norm of a h is equivalent to $\sum_{k\geqslant 0} \|h_k\|^2_{L_2(\mathbb{S}^d)} k^{d+2\alpha+1}$. Given δ, we may then find the function h such that $\gamma_2(h)^2 = \sum_{k\geqslant 0} \|h_k\|^2_{L_2(\mathbb{S}^d)} k^{d+2\alpha+1} \leqslant \delta^2$ with smallest $L_2(\mathbb{S}^d)$ norm distance to g, that is, $\sum_{k\geqslant 0} \|g_k - h_k\|^2_{L_2(\mathbb{S}^d)}$. The optimal approximation is $h_k = \alpha_k g_k$ for some $\alpha_k \in \mathbb{R}_+$, with error $\sum_{k\geqslant 0}(1-\alpha_k)^2 \|g_k\|^2_{L_2(\mathbb{S}^d)} \sim \sum_{k\geqslant 0}(1-\alpha_k)^2 k^{-2s-2}$ and squared γ_2-norm $\sum_{k\geqslant 0} \alpha_k^2 k^{d+2\alpha+1} k^{-2s-2} = \sum_{k\geqslant 0} \alpha_k^2 k^{d+2\alpha-2s-1}$. The optimal α_k is obtained by considering a Lagrange multiplier λ such that $(\alpha_k - 1)k^{-2s-2} + \lambda\alpha_k k^{d+2\alpha-2s-1} = 0$, that is, $\alpha_k = (k^{-2s-2} + \lambda k^{d+2\alpha-2s-1})^{-1} k^{-2s-2} = (1+\lambda k^{d+2\alpha+1})^{-1}$. We then have

$$\sum_{k\geqslant 0} \alpha_k^2 k^{d+2\alpha-2s-1} = \sum_{k\geqslant 0} (1+\lambda k^{d+2\alpha+1})^{-2} k^{d+2\alpha-2s-1}$$
$$\approx \int_0^\infty (1+\lambda t^{d+2\alpha+1})^{-2} t^{d+2\alpha-2s-1} dt \text{ by approximation by an integral,}$$
$$\propto \int_0^\infty (1+u)^{-2} d(t^{d+2\alpha-2s}) \text{ with the change of variable } u = \lambda t^{d+2\alpha+1}$$
$$\propto \lambda^{-(d+2\alpha-2s)/(d+2\alpha+1)} \text{ up to constants,}$$

which should be of order δ^2 (this gives the scaling of λ as a function of δ). Then the squared error is

$$\sum_{k\geqslant 0} (1-\alpha_k)^2 k^{-2s-2} = \sum_{k\geqslant 0} \frac{\lambda^2 t^{2d+4\alpha+2}}{(1+\lambda k^{d+2\alpha+1})^2} k^{-2s-2}$$
$$\approx \int_0^\infty \frac{\lambda^2 t^{2d+4\alpha-2s}}{(1+\lambda t^{d+2\alpha+1})^2} dt$$
$$\approx \lambda^2 \lambda^{-(2d+4\alpha-2s+1)/(d+2\alpha+1)} = \lambda^{-(2d+4\alpha-2s+1-2d-4\alpha-2)/(d+2\alpha+1)}$$
$$= \lambda^{(2s+1)/(d+2\alpha+1)} \approx \delta^{-2(2s+1)/(d+2\alpha-2s)},$$

and thus the (non-squared) approximation error scales as $\delta^{-(2s+1)/(d+2\alpha-2s)}$. For $s=1$, this leads to a scaling as $\delta^{-3/(d+2\alpha-2)}$.

D.6 Proof of Prop. 4

For $\alpha = 1$, by writing $v^\top x = (v^\top x)_+ - (-v^\top x)_+$ we obtain the upperbound $\gamma_1(g) \leqslant 2$. For all other situations, we may compute

$$\gamma_2(g)^2 = \sum_{k\geqslant 0} \frac{\|g_k\|^2_{L_2(\mathbb{S}^d)}}{\lambda_k^2}.$$

For g a linear function $g_k = 0$ except for $k=1$, for which, we have $g_1(x) = v^\top x$, and thus $\|g_k\|^2_{L_2(\mathbb{S}^d)} = \int_{\mathbb{S}^d} (v^\top x)^2 d\tau_d(x) = v^\top (\int_{\mathbb{S}^d} x x^\top \tau_d(x)) v = 1$. This implies that $\gamma_2(g) = \lambda_1^{-1}$. Given the expression (from Appendix D.2) $\lambda_1 = \frac{d-1}{d}\frac{\alpha}{4\pi}\frac{\Gamma(\alpha/2)\Gamma(d/2+1)}{\Gamma(\alpha/2+d/2+1)}$ for $\alpha > 1$ and $\lambda_1 = \frac{d-1}{2d\pi}$.

Appendix E. Computing ℓ_2-Haussdorff Distance between Ellipsoids

We assume that we are given two ellipsoids defined as $(x-a)^\top A^{-1}(x-a) \leqslant 1$ and $(x-b)^\top B^{-1}(x-b) \leqslant 1$ and we want to compute their Hausdorff distance. This leads to the two equivalent problems

$$\max_{\|w\|_2 \leqslant 1} w^\top (a-b) - \|B^{1/2} w\|_2 + \|A^{1/2} w\|_2,$$

$$\max_{\|u\|_2 \leqslant 1} \min_{\|v\|_2 \leqslant 1} \|a + A^{1/2} u - b - B^{1/2} v\|_2,$$

which are related by $w = a + A^{1/2}u - b - B^{1/2}v$. We first review classical methods for optimization of quadratic functions over the ℓ_2-unit ball.

Minimizing convex quadratic forms over the sphere. We consider the following convex optimization problem, with $Q \succcurlyeq 0$; we have by Lagrangian duality:

$$\begin{aligned}
&\min_{\|x\|_2 \leqslant 1} \frac{1}{2} x^\top Q x - q^\top x \\
&\max_{\lambda \geqslant 0} \min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top Q x - q^\top x + \frac{\lambda}{2}(\|x\|_2^2 - 1) \\
&\max_{\lambda \geqslant 0} -\frac{1}{2} q^\top (Q + \lambda I)^{-1} q - \frac{\lambda}{2} \text{ with } x = (Q+\lambda I)^{-1} q.
\end{aligned}$$

If $\|Q^{-1}q\|_2 \leqslant 1$, then $\lambda = 0$ and $x = Q^{-1}q$. Otherwise, at the optimum, $\lambda > 0$ and $\|x\|_2^2 = q^\top (Q+\lambda I)^{-2} q = 1$, which implies $1 \leqslant \frac{1}{\lambda + \lambda_{\min}(Q)} q^\top Q^{-1} q$, which leads to $\lambda \leqslant q^\top Q^{-1} q - \lambda_{\min}(Q)$, which is important to reduce the interval of possible λ. The optimal λ may then be obtained by binary search (from a single SVD of Q).

Minimizing concave quadratic forms over the sphere. We consider the following non-convex optimization problem, with $Q \succcurlyeq 0$, for which strong Lagrangian duality is known to hold (Boyd and Vandenberghe, 2004):

$$\begin{aligned}
&\min_{\|x\|_2 \leqslant 1} -\frac{1}{2} x^\top Q x + q^\top x = \min_{\|x\|_2 = 1} -\frac{1}{2} x^\top Q x + q^\top x \\
&\max_{\lambda \geqslant 0} \min_{x \in \mathbb{R}^d} -\frac{1}{2} x^\top Q x + q^\top x + \frac{\lambda}{2}(\|x\|_2^2 - 1) \\
&\max_{\lambda \geqslant \lambda_{\max}(Q)} -\frac{1}{2} q^\top (\lambda I - Q)^{-1} q - \frac{\lambda}{2} \text{ with } x = (Q - \lambda I)^{-1} q.
\end{aligned}$$

At the optimum, we have $q^\top (\lambda I - Q)^{-2} q = 1$, which implies $1 \leqslant \frac{1}{[\lambda - \lambda_{\max}(Q)]^2} \|q\|_2^2$, which leads to $0 \leqslant \lambda - \lambda_{\max}(Q) \leqslant \|q\|_2$. We may perform binary search on λ from a single SVD of Q.

Computing the Haussdorff distance. We need to compute:

$$\max_{\|u\|_2 \leqslant 1} \min_{\|v\|_2 \leqslant 1} \frac{1}{2} \|a + A^{1/2}u - b - B^{1/2}v\|_2^2$$

$$\begin{aligned}
&= \max_{\|u\|_2 \leqslant 1} \max_{\lambda \geqslant 0} \min_{\|v\|_2 \leqslant 1} \frac{1}{2} \|a + A^{1/2}u - b - B^{1/2}v\|_2^2 + \frac{\lambda}{2}(\|v\|_2^2 - 1) \\
&= \max_{\|u\|_2 \leqslant 1} \max_{\lambda \geqslant 0} -\frac{\lambda}{2} + \frac{1}{2}\|a - b + A^{1/2}u\|^2 - \frac{1}{2}(a - b + A^{1/2}u)^\top B (B + \lambda I)^{-1} (a - b + A^{1/2}u) \\
&= \max_{\|u\|_2 \leqslant 1} \max_{\lambda \geqslant 0} -\frac{\lambda}{2} + \frac{\lambda}{2}(a - b + A^{1/2}u)^\top (B + \lambda I)^{-1} (a - b + A^{1/2}u)
\end{aligned}$$

with $v = (B + \lambda I)^{-1} B^{1/2} (a - b + A^{1/2}u)$. The interval in λ which is sufficient to explore is

$$\lambda \in [0, -\lambda_{\min}(B) + (\|a-b\|_2^2 + \lambda_{\max}(A^{1/2}))^2],$$

which are bounds that are independent of u.

Given $\lambda \geqslant 0$, we have the problem of

$$\begin{aligned}
&\min_{\mu \geqslant 0} \max_{u \in \mathbb{R}^d} \frac{\lambda}{2}(a - b + A^{1/2}u)^\top (B + \lambda I)^{-1} (a - b + A^{1/2}u) - \frac{\mu}{2}(\|u\|_2^2 - 1) - \frac{\lambda}{2} \\
&= \min_{\mu \geqslant 0} \max_{u \in \mathbb{R}^d} \frac{\lambda}{2}(a-b)^\top (B + \lambda I)^{-1}(a-b) + \frac{\mu - \lambda}{2} + \lambda u^\top A^{1/2} (B + \lambda I)^{-1} (a - b) \\
&\qquad\qquad - \frac{1}{2} u^\top (\mu I - \lambda A^{1/2} (B + \lambda I)^{-1} A^{1/2}) u \\
&= \min_{\mu \geqslant 0} \frac{\lambda}{2}(a-b)^\top (B + \lambda I)^{-1}(a-b) + \frac{\mu - \lambda}{2} \\
&\qquad + \lambda^2 (a-b)^\top (B + \lambda I)^{-1} A^{1/2} (\mu I - \lambda A^{1/2} (B + \lambda I)^{-1} A^{1/2})^{-1} A^{1/2} (B + \lambda I)^{-1} (a - b)
\end{aligned}$$

We have $u = (\frac{\mu}{\lambda} I - A^{1/2}(B + \lambda I)^{-1} A^{1/2})^{-1} A^{1/2} (B + \lambda I)^{-1} (a - b)$, leading to $w \propto (\lambda^{-1} B - \mu^{-1} A + I)(a - b)$. We need $\frac{\mu}{\lambda} \geqslant \lambda_{\max}(A^{1/2}(B + \lambda I)^{-1} A^{1/2})$. Moreover

$$0 \leqslant \frac{\mu}{\lambda} - \lambda_{\max}(A^{1/2}(B + \lambda I)^{-1} A^{1/2}) \leqslant \|A^{1/2}(B + \lambda I)^{-1}(a - b)\|.$$

This means that the ℓ_2-Haussdorff distance may be computed by solving in λ and μ, by exhaustive search with respect to λ and by binary search (or Newton's method) for μ. The complexity of each iteration is that of a singular value decomposition, that is $O(d^3)$. For more details on optimization of quadratic functions on the unit-sphere, see Forsythe and Golub (1965).

References

P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2009.

R. A. Adams and J. F. Fournier. *Sobolev Spaces*, volume 140. Academic Press, 2003.

K. Atkinson and W. Han. *Spherical Harmonics and Approximations on the Unit Sphere: an Introduction*, volume 2044. Springer, 2012.

F. Bach. Consistency of the group Lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008a.

F. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2008b.

F. Bach. Convex relaxations of structured matrix factorizations. Technical Report 00861118, HAL, 2013.

F. Bach. Duality between subgradient and conditional gradient methods. *SIAM Journal on Optimization*, 25(1):115–129, 2015.

F. Bach. On the equivalence between kernel quadrature rules and random feature expansions. *Journal of Machine Learning Research*, 18:1–38, 2017.

A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2003.

P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

A. Barvinok. *A Course in Convexity*, volume 54. American Mathematical Society, 2002.

Y. Bengio, N. Le Roux, P. Vincent, O. Delalleau, and P. Marcotte. Convex neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2006.

A. Berlinet and C. Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*, volume 3. Springer, 2004.

E. D. Bolker. A class of convex bodies. *Transactions of the American Mathematical Society*, 145:323–345, 1969.

L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

J. Bourgain and J. Lindenstrauss. Projection bodies. In *Geometric Aspects of Functional Analysis*, pages 250–270. Springer, 1988.

J. Bourgain, J. Lindenstrauss, and V. Milman. Approximation of zonoids by zonotopes. *Acta Mathematica*, 162(1):73–141, 1989.

S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

L. Breiman. Hinging hyperplanes for regression, classification, and function approximation. *IEEE Transactions on Information Theory*, 39(3):999–1013, 1993.

P. Bühlmann and S. Van De Geer. *Statistics for high-dimensional data: methods, theory and applications*. Springer, 2011.

M. Burger and A. Neubauer. Error bounds for approximation with neural networks. *Journal of Approximation Theory*, 112(2):235–250, 2001.

Y. Cho and L. K. Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

A. S. Dalalyan, A. Juditsky, and V. Spokoiny. A new algorithm for estimating the effective dimension-reduction subspace. *Journal of Machine Learning Research*, 9:1647–1678, 2008.

V. F. Dem'yanov and A. M. Rubinov. The minimization of a smooth convex functional on a convex set. *SIAM Journal on Control*, 5(2):280–294, 1967.

R. A. DeVore, R. Howard, and C. Micchelli. Optimal nonlinear approximation. *Manuscripta Mathematica*, 63(4):469–478, 1989.

M. Dudik, Z. Harchaoui, and J. Malick. Lifted coordinate descent for learning with trace-norm regularization. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.

J. C. Dunn and S. Harshbarger. Conditional gradient algorithms with open loop step size rules. *Journal of Mathematical Analysis and Applications*, 62(2):432–444, 1978.

H. Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10. Springer, 1987.

L. C. Evans and R. F. Gariepy. *Measure Theory and Fine Properties of Functions*, volume 5. CRC Press, 1991.

G. E. Forsythe and G. H. Golub. On the stationary values of a second-degree polynomial on the unit sphere. *Journal of the Society for Industrial & Applied Mathematics*, 13(4): 1050–1068, 1965.

M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956.

J. H. Friedman and W. Stuetzle. Projection pursuit regression. *Journal of the American Statistical Association*, 76(376):817–823, 1981.

C. Frye and C. J. Efthimiou. Spherical Harmonics in p Dimensions. Technical Report 1205.3548, ArXiv, 2012.

K. Fukumizu, F. Bach, and M. I. Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

C. Gu. *Smoothing Spline ANOVA Models*, volume 297. Springer, 2013.

L. J. Guibas, A. Nguyen, and L. Zhang. Zonotopes as bounding volumes. In *Proceedings of the ACM-SIAM symposium on Discrete Algorithms*, 2003.

V. Guruswami and P. Raghavendra. Hardness of learning halfspaces with noise. *SIAM Journal on Computing*, 39(2):742–765, 2009.

L. Györfi and A. Krzyzak. *A Distribution-free Theory of Nonparametric Regression*. Springer, 2002.

Z. Harchaoui, A. Juditsky, and A. Nemirovski. Conditional gradient algorithms for norm-regularized smooth convex optimization. *Mathematical Programming*, pages 1–38, 2013.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2009. 2nd edition.

T. J. Hastie and R. J. Tibshirani. *Generalized Additive Models*. Chapman & Hall, 1990.

S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1994.

G. E. Hinton and Z. Ghahramani. Generative models for discovering sparse distributed representations. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 352(1358):1177–1190, 1997.

M. Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013.

S. M. Kakade, K. Sridharan, and A. Tewari. On the complexity of linear prediction: Risk bounds, margin bounds, and regularization. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

A. R. Klivans and A. A. Sherstov. Cryptographic hardness for learning intersections of halfspaces. In *Annual Symposium on Foundations of Computer Science (FOCS)*, 2006.

V. Koltchinskii. Rademacher penalties and structural risk minimization. *IEEE Transactions on Information Theory*, 47(5):1902–1914, 2001.

S. König. Computational aspects of the Hausdorff distance in unbounded dimension. Technical Report 1401.1434, ArXiv, 2014.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

V. Kurkova and M. Sanguineti. Bounds on rates of variable-basis and neural-network approximation. *IEEE Transactions on Information Theory*, 47(6):2659–2665, Sep 2001.

G. Lan. The complexity of large-scale convex programming under a linear optimization oracle. Technical Report 1309.5550, arXiv, 2013.

N. Le Roux and Y. Bengio. Continuous neural networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2007.

W. S. Lee, P. L. Bartlett, and R. C. Williamson. Efficient agnostic learning of neural networks with bounded fan-in. *IEEE Transactions on Information Theory*, 42(6):2118–2132, 1996.

M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6 (6):861–867, 1993.

K.-C. Li. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86(414):316–327, 1991.

Y. Lin and H. H. Zhang. Component selection and smoothing in multivariate nonparametric regression. *Annals of Statistics*, 34(5):2272–2297, 2006.

Roi Livni, Shai Shalev-Shwartz, and Ohad Shamir. On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*, 2014.

V. Maiorov. Approximation by neural networks and learning theory. *Journal of Complexity*, 22(1):102–117, 2006.

V. E. Maiorov and R. Meir. On the near optimality of the stochastic approximation of smooth functions by neural networks. *Advances in Computational Mathematics*, 13(1): 79–103, 2000.

Y. Makovoz. Uniform approximation by neural networks. *Journal of Approximation Theory*, 95(2):215–228, 1998.

J. Matoušek. Improved upper bounds for approximation by zonotopes. *Acta Mathematica*, 177(1):55–73, 1996.

H. N. Mhaskar. On the tractability of multivariate integration and approximation by neural networks. *Journal of Complexity*, 20(4):561–590, 2004.

H. N. Mhaskar. Weighted quadrature formulas and approximation by zonal function networks on the sphere. *Journal of Complexity*, 22(3):348–370, 2006.

G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio. On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, 2014.

V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of International Conference on Machine Learning (ICML)*, 2010.

R. M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.

Y. Nesterov. Semidefinite relaxation and nonconvex quadratic optimization. *Optimization Methods and Software*, 9(1-3):141–160, 1998.

Y. Nesterov. *Introductory lectures on convex optimization: a basic course*. Kluwer Academic Publishers, 2004.

Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming*, 103(1):127–152, 2005.

P. P. Petrushev. Approximation by ridge functions and neural networks. *SIAM Journal on Mathematical Analysis*, 30(1):155–189, 1998.

A. Pinkus. Approximation theory of the MLP model in neural networks. *Acta Numerica*, 8:143–195, 1999.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

P. Ravikumar, H. Liu, J. Lafferty, and L. Wasserman. SpAM: Sparse additive models. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1997.

F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 1958.

S. Rosset, J. Zhu, and T. Hastie. Boosting as a regularized path to a maximum margin classifier. *Journal of Machine Learning Research*, 5:941–973, 2004.

S. Rosset, G. Swirszcz, N. Srebro, and J. Zhu. ℓ_1-regularization in infinite dimensional feature spaces. In *Proceedings of the Conference on Learning Theory (COLT)*, 2007.

W. Rudin. *Real and Complex Analysis*. Tata McGraw-Hill Education, 1987.

D. E. Rumelhart, G. E Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

R. Schneider. Zu einem problem von shephard über die projektionen konvexer körper. *Mathematische Zeitschrift*, 101(1):71–82, 1967.

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

A. J. Smola, Z. L. Ovari, and R. C. Williamson. Regularization with dot-product kernels. *Advances in Neural Information Processing Systems (NIPS)*, 2001.

K. Sridharan. *Learning from an Optimization Viewpoint*. PhD thesis, Toyota Technological Institute at Chicago, 2012.

U. von Luxburg and O. Bousquet. Distance–based classification with Lipschitz functions. *Journal of Machine Learning Research*, 5:669–695, 2004.

H. Whitney. Analytic extensions of differentiable functions defined in closed sets. *Transactions of the American Mathematical Society*, 36(1):63–89, 1934.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

X. Zhang, D. Schuurmans, and Y. Yu. Accelerated training for matrix-norm regularization: A boosting approach. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

Journal of Machine Learning Research 18 (2017) 1-32 Submitted 1/15; Revised 2/17; Published 3/17

Memory Efficient Kernel Approximation

Si Si* SISIDAISY@GOOGLE.COM
Google Research
Mountain View, CA 94043, USA
Cho-Jui Hsieh CHOHSIEH@UCDAVIS.EDU
Departments of Computer Science and Statistics
University of California, Davis
Davis, CA 95616, USA
Inderjit S. Dhillon INDERJIT@CS.UTEXAS.EDU
Department of Computer Science
University of Texas at Austin
Austin, TX 78701, USA

Editor: Le Song

Abstract

Scaling kernel machines to massive data sets is a major challenge due to storage and computation issues in handling large kernel matrices, that are usually dense. Recently, many papers have suggested tackling this problem by using a low-rank approximation of the kernel matrix. In this paper, we first make the observation that the structure of shift-invariant kernels changes from low-rank to block-diagonal (without any low-rank structure) when varying the scale parameter. Based on this observation, we propose a new kernel approximation framework – Memory Efficient Kernel Approximation (MEKA), which considers both low-rank and clustering structure of the kernel matrix. We show that the resulting algorithm outperforms state-of-the-art low-rank kernel approximation methods in terms of speed, approximation error, and memory usage. As an example, on the covtype dataset with half a million samples, MEKA takes around 70 seconds and uses less than 80 MB memory on a single machine to achieve 10% relative approximation error, while standard Nyström approximation is about 6 times slower and uses more than 400MB memory to achieve similar approximation. We also present extensive experiments on applying MEKA to speed up kernel ridge regression.

Keywords: kernel approximation, Nyström method, kernel methods

1. Introduction

Kernel methods (Schölkopf and Smola, 2002) are a class of machine learning algorithms that first map samples from input space to a high-dimensional feature space. In the high-dimensional feature space, various methods can be applied depending on the machine learning task, for example, kernel support vector machine (SVM) (Cortes and Vapnik, 1995) (Hsieh, Si, and Dhillon, 2014a) and kernel ridge regression (Saunders, Gammerman, and Vovk, 1998). A key issue in scaling up kernel machines is the storage and computation of the kernel matrix, which is usually dense. Storing the dense matrix takes $O(n^2)$ space, while computing it requires $O(n^2 d)$ operations, where n is the number of data points and d is the dimension. A common approach to achieve scalability is to approximate the kernel matrix using limited memory storage. This approach not only resolves the memory issue, but also speeds up kernel machine solvers, because the time complexity for using the kernel is usually proportional to the amount of memory used to represent the kernel. Most kernel approximation methods aim to form a low-rank approximation $G \approx CC^T$ for the kernel matrix G, with $C \in \mathbb{R}^{n\times k}$ and rank $k \ll n$. Although it is well known that Singular Value Decomposition (SVD) yields the best rank-k approximation, it often cannot be applied as it requires the entire kernel matrix to be computed and stored. To overcome this issue, many methods have been proposed to approximate the best rank-k approximation of a kernel matrix, including Greedy basis selection techniques (Smola and Schölkopf, 2000), incomplete Cholesky decomposition (Fine and Scheinberg, 2001), and Nyström methods (Williams and Seeger, 2001).

However, it is unclear whether low-rank approximation is the most memory efficient way to approximate a kernel matrix. In this paper, we first make the observation that for practically used shift-invariant kernels, the kernel structure varies from low-rank to block-diagonal as the scaling parameter γ varies from 0 to ∞. This observation suggests that even the best rank-k approximation can have extremely large approximation error when γ is large, so it is worth exploiting the block structure of the kernel matrix. Based on this idea, we propose a Memory Efficient Kernel Approximation (MEKA) framework to approximate the kernel matrix. Our proposed framework considers and analyzes the use of clustering in the input space to efficiently exploit the block structure of shift-invariant kernels. We show that the individual blocks generated by kmeans clustering have low-rank structure, which motivates us to apply Nyström low-rank approximation to each block separately. Between-cluster blocks are then approximated in a memory-efficient manner. Our approach only needs $O(nk + (ck)^2)$ memory to store a rank-ck approximation(where $c \ll n$ is the number of clusters), while traditional low-rank methods need $O(nk)$ space to store a rank-k approximation. Therefore, using the same amount of storage, our method can achieve lower approximation error than the commonly used low-rank methods. Moreover, our proposed method takes less computation time than other low-rank methods to achieve a given approximation error.

Theoretically, we show that under the same amount of storage, the error bound of our approach can be better than standard Nyström if the gap between the $k+1$-st and $ck+1$-st singular values of G is larger than $\|\Delta\|_2$ where Δ consists of all between-cluster blocks. On real datasets, our proposed algorithm consumes less memory and computation time to achieve comparable reconstruction error. For example, on the covtype dataset with half million samples, MEKA takes around 70 seconds and uses less than 80 MB memory on a single machine to achieve 10% relative approximation error, while standard Nyström approximation takes more than 400 seconds and uses more than 400MB memory to achieve similar approximation. Also, MEKA is faster for kernel ridge regression compared with other state-of-the-art kernel approximation methods. As an example, on the mnist2m dataset with 2 million samples, our method takes 550 seconds on a single machine using less than 500 MBytes memory to achieve accuracy comparable with standard Nyström approximation,

∗. This work was done before joining Google.

©2017 Si Si, Cho-Jui Hsieh, and Inderjit S. Dhillon.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-025.html.

which takes more than 2700 seconds and uses more than 2 GBytes memory on the same problem.

Parts of this paper have appeared previously in (Si, Hsieh, and Dhillon, 2014a). In this paper, we provide: (1) a more detailed survey of state-of-the-art methods; (2) much more comprehensive experimental comparisons; (3) thorough investigation of the influence of the parameters in our method; (4) the application of applying the block structure of kernel matrix to speed up kernel SVM; and (5) more discussion including how to achieve stable results and solve non-psd issues in MEKA.

The rest of the paper is outlined as follows. We first present related work in Section 2. We then explain the popular Nyström approximation method and present motivation for our framework in Section 3. We then show the block structure of kernel matrix and its application to speed up kernel SVM in Section 4. Our main kernel approximation algorithm MEKA is proposed and analyzed in Section 5. Experimental results are given in Section 6, and conclusion and discussion are provided in Section 7.

2. Related Research

To approximate the kernel matrix using limited memory, one common way is to use a low-rank approximation. The best rank-k approximation can be obtained by the SVD, but it is computationally prohibitive when n grows to tens of thousands. To address the scalability issue of SVD, approximate SVD solvers such as randomized SVD (Halko, Martinsson, and Tropp, 2011) have been widely used for large-scale data. To exploit the sparse structure of large-scale network data, an alternative is to apply CUR matrix decomposition (Mahoney and Drineas, 2009) that explicitly expresses the low-rank decomposition in terms of a small number of rows and columns of the original data matrix. Another way is building a hierarchical tree to initialize a block Lanczos algorithm to efficiently compute the spectral decomposition of large-scale graphs (Si, Shin, Dhillon, and Parlett, 2014b). Unfortunately, to approximate kernel matrices, all the above approaches need to compute the entire kernel matrix, so the time complexity is at least $O(dn^2)$.

Many algorithms have been proposed to overcome the prohibitive time and space complexity of SVD for approximating kernel matrices. They can be categorized into two classes: methods that explicitly approximate kernel matrices, and methods that approximate the kernel function.

Approximating the kernel matrix. The first class of approaches approximate the kernel matrix based on a subset of sampled elements; as a result, all of them are data dependent. The Nyström method (Williams and Seeger, 2001) is the most widely used technique to approximate the kernel matrix given a sampled subset of columns. To approximate a rank-k approximation, Nyström approximation requires $O(nk^2 + k^3)$ time to form the rank-k approximation. Many strategies have been proposed to improve over the basic Nyström approximation, including ensemble Nyström (Kumar, Mohri, and Talwalkar, 2009), Nyström with k-means to obtain benchmark points (Zhang, Tsang, and Kwok, 2008; Zhang and Kwok, 2010), randomized Nyström (Li, Kwok, and Lu, 2010), Nyström approximation with shift (Wang et al., 2014), adding "pseudo landmark points" to the Nyström approximation (Hsieh, Si, and Dhillon, 2014b), and fast-Nys that uses fast transform to generate structured landmark points to speed up Nyström approximation (Si, Hsieh, and Dhillon, 2016).

Different Nyström sampling strategies are analyzed and compared in (Kumar, Mohri, and Talwalkar, 2012; Gittens and Mahoney, 2013). Besides Nyström approximation, Fine and Scheinberg (2001) use the incomplete Cholesky decomposition with pivoting for approximating kernel matrices, which requires $O(nk^2 + nkd)$ time for computing a rank-k approximation. Bach and Jordan (2005) incorporate side information (labels) into the incomplete Cholesky decomposition, and show that the resulting problem can be solved with the same $O(nk^2 + nkd)$ time complexity. Finally, Achlioptas, McSherry, and Schölkopf (2001) propose a sampling and reweighted approach to obtain an unbiased estimator of the kernel-vector product, and use subspace iteration to approximate the top k eigenvectors.

Approximating the kernel function. The second class of methods is to directly approximate the kernel function without computing elements of the kernel matrix, so the approximation does not depend on the data. To approximate the kernel function, a typical approach is to find a feature mapping $Z : \mathbb{R}^d \to \mathbb{R}^k$ where the kernel function $K(\boldsymbol{x}, \boldsymbol{y})$ can be approximated by $Z(\boldsymbol{x})^T Z(\boldsymbol{y})$. Rahimi and Recht (2007, 2008) define the random feature map for shift invariant kernel functions based on the Fourier transform. In the resulting Random Kitchen Sinks (RKS) algorithm, the main computation turns out to be the matrix vector multiplication $W\boldsymbol{x}_i$ for each instance $\boldsymbol{x}_i$, where W is a Gaussian random matrix. To improve efficiency, Le, Sarlos, and Smola (2013) show that the computation of $W\boldsymbol{x}_i$ can be sped up by the fast Hadamard transform. On the other hand, Yang et al. (2014) propose to use a quasi Monte Carlo approach to improve the approximation performance of RKS. In addition to shift invariant kernels, Kar and Karnick (2012) construct the random feature map for polynomial kernels, and Hamid et al. (2014) propose the condensed random feature map to improve performance.

Besides the above approaches based on random feature maps, there are other methods that directly approximate the kernel function. Cotter, Keshet, and Srebro (2011) approximate the Gaussian kernel by the t-th order Taylor expansion, but it requires $O(d^t)$ features, which is computationally burdensome for large d or t. Chang et al. (2010) propose to use the kernel expansion for low-degree polynomial kernels. Recently, Yang et al. (2012) showed that the Nyström method has a better generalization error bound than the RKS approach if the gap in the eigen-spectrum of the kernel matrix is large.

Most of the above methods can be viewed as faster ways to find a rank-k approximation of the kernel matrix, so the approximation error is always worse than the top-k SVD when using $O(nk)$ memory. As we show in Section 3, the kernel matrix typically changes from low-rank to block structure as the scaling parameter γ increases. However, the block structure of the kernel matrix has *never* been considered in dense kernel approximation, although it has been studied for approximation of other types of matrices. For example, Savas and Dhillon (2011) applied Clustered Low Rank Approximation (CLRA) to approximate large and sparse social networks. CLRA applies spectral or graph clustering to the adjacency matrix of a social network, runs SVD on each diagonal block, and uses matrix projection to capture off-diagonal information. All these steps require storage of the entire matrix, thus they are infeasible for large-scale dense kernel matrices. For example, computing the entire kernel matrix of the mnist2m dataset would require about 8 TBytes of memory; moreover the time for computing the SVD and projection steps is prohibitive. On the same dataset

our proposed approach can obtain accurate results in 10 minutes with only 500 MBytes memory (as shown in Section 6). To achieve this, we need totally different algorithms than the ones in CLRA for clustering and approximating blocks to yield our memory-efficient scheme.

More specifically, CLRA was designed to approximate sparse adjacency matrices, but cannot be directly applied to large kernel matrices as that would require computation and storage of the entire kernel matrix at a cost of $O(n^2d)$ time and $O(n^2)$ space. To overcome this problem, we propose the following innovations: (1) We perform clustering, and then apply Nyström approximation to within-cluster blocks to avoid computing all within-block entries; (2) We theoretically justify the use of kmeans clustering to explore the block structure of the kernel; (3) We propose a sampling approach to capture between-block information; (4) We theoretically show the error bound of our method and compare it with the traditional Nyström approach.

3. Preliminaries and Motivation

Let $K(\cdot,\cdot)$ denote the kernel function, and $G \in \mathbb{R}^{n\times n}$ be the corresponding kernel matrix where $G_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, and $\boldsymbol{x}_i$, $\boldsymbol{x}_j \in \mathbb{R}^d$ are data points. Computing and storing the kernel matrix G usually takes $O(n^2d)$ time and $O(n^2)$ space, which is prohibitive when there are millions of samples. One way to deal with these challenges is to approximate the dense kernel matrix G by a low-rank approximation $\tilde{G}$. By doing this, kernel machines are transformed to linear problems which can be solved efficiently. The best rank-k approximation of G is given by its singular value decomposition(SVD), i.e., $G \approx U_k\Sigma_k{U_k}^T$, where Σ_k is the diagonal matrix of largest k singular values and U_k contains the corresponding singular vectors.

However, computing the SVD of G is computationally prohibitive and memory intensive. Many fast kernel approximation algorithms have thus been proposed and studied. Nyström kernel approximation is a widely used approximation approach, which uses a sample of m data points and does not need to form the entire G explicitly to generate its low-rank approximation. In standard Nyström approximation (proposed in Williams and Seeger (2001)), we first uniformly at random sample m data points and assemble the corresponding m columns of G as the $n \times m$ matrix C. Let M be the $m \times m$ kernel matrix between the m sampled points, then the standard Nyström method generates a rank-k approximation to G as

$$G \approx \tilde{G} = CM_k^+C^T, \tag{1}$$

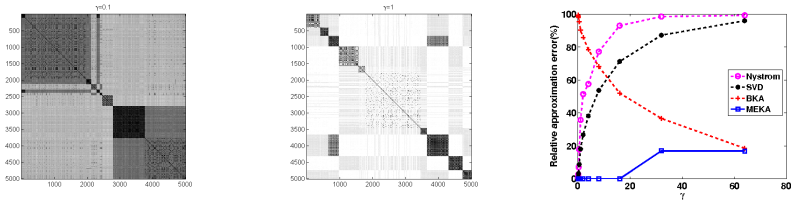
where M_k is the best rank-k approximation of M (by SVD) and M_k^+ is its pseudo-inverse. Various extensions to this Nyström based kernel approximation have been proposed. For example, k-means Nyström(Zhang, Tsang, and Kwok, 2008; Zhang and Kwok, 2010) uses clusters centroids as the landmark points to form C; ensemble Nyström(Kumar, Mohri, and Talwalkar, 2009) combines a collection of standard Nyström approximations. We will compare state-of-the-art Nyström based methods in Section 6.

We use the Gaussian kernel as an example to discuss the structure of the kernel matrix under different scale parameters. Given two samples $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, the Gaussian kernel is given by $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\gamma\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_2^2}$, where γ is a scale or width parameter; the corresponding kernel matrix entries are $G_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Low-rank approximation has been widely used

to obtain an approximation for kernel matrices. However, under different scale parameters, the kernel matrix has quite different structures, suggesting that different approximation strategies should be used for different γ.

Let us examine two extreme cases of the Gaussian kernel: when $\gamma \to 0$, $G \to \boldsymbol{e}\boldsymbol{e}^T$ where $\boldsymbol{e} = [1,\ldots,1]^T$. As a consequence, G is close to low-rank when γ is small. However, at the other extreme as $\gamma \to \infty$, G changes to the identity matrix, which has full rank with all eigenvalues equal to 1. In this case, G does not have a low-rank structure, but has a block/clustering structure. This observation motivates us to consider both low rank and clustering structure of the kernel matrix. Figures 1a and 1b give an example of the structure of a Gaussian kernel with different γ on a real dataset by randomly sampling 5000 samples from the covtype dataset.

Before discussing further details, we first contrast the use of block and low-rank approximations on the same dataset. We compare approximation errors for different methods when they use the same amount of memory in Figure 1c. Clearly, low-rank approximation methods work well only for very small γ values. Block Kernel Approximation (BKA), as proposed in Section 4.1, is a simple way to use clustering structure of G that is effective for large γ. Our proposed algorithm, MEKA, considers both block and low-rank structure of the kernel, and thus performs better than others under different γ values as seen in Figure 1c.



(a) The Gaussian kernel matrix with $\gamma = 0.1$ on covtype dataset

(b) The Gaussian kernel matrix with $\gamma = 1$ on covtype dataset

(c) Comparison of different kernel approximation methods for various γ.

Figure 1: (a) and (b) show that the structure of the Gaussian kernel matrix $K(\boldsymbol{x}, \boldsymbol{y}) = e^{-\gamma\|\boldsymbol{x}-\boldsymbol{y}\|^2}$ for the covtype data tends to become more block diagonal as γ increases(dark regions correspond to large values, while lighter regions correspond to smaller values). Plot (c) shows that low-rank approximations work only for small γ, and Block Kernel Approximation (BKA) works for large γ, while our proposed method MEKA works for small as well as large γ.

4. Block Kernel Approximation

In this section, we first introduce Block Kernel Approximation (BKA), a simple way to exploit the clustering structure of kernel matrices, and then show its application for speeding up kernel SVM.

4.1 Clustering Structure of Shift-invariant Kernel Matrices

There has been substantial research on approximating shift-invariant kernels (Rahimi and Recht, 2007). A kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is shift-invariant if the kernel value depends only on $\boldsymbol{x}_i - \boldsymbol{x}_j$, that is, $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = f(\eta(\boldsymbol{x}_i - \boldsymbol{x}_j))$ where $f(\cdot)$ is a function that maps $\mathbb{R}^d$ to $\mathbb{R}$, and $\eta > 0$ is a constant to determine the "scale" of the data. η is very crucial to the performance of kernel machines and is usually chosen by cross-validation. We further define $g_{\boldsymbol{u}}(t) = f(\eta t \boldsymbol{u})$ to be a one variable function along $\boldsymbol{u}$'s direction where $\boldsymbol{u}$ is an unit vector. We assume the kernel function satisfies the following property:

Assumption 1 *$g_{\boldsymbol{u}}(t)$ is differentiable for all $t \neq 0$.*

Most of the practically used shift-invariant kernels satisfy the above assumption, for example, the Gaussian kernel ($K(\boldsymbol{x}, \boldsymbol{y}) = e^{-\gamma\|\boldsymbol{x}-\boldsymbol{y}\|_2^2}$), and the Laplacian kernel ($K(\boldsymbol{x}, \boldsymbol{y}) = e^{-\gamma\|\boldsymbol{x}-\boldsymbol{y}\|_1}$). It is clear that η^2 is equivalent to γ for the Gaussian kernel if written in the form of $K(\boldsymbol{x}, \boldsymbol{y}) = f(\eta(\boldsymbol{x} - \boldsymbol{y}))$. When η is large, off-diagonal blocks of shift-invariant kernel matrices will become small, and most of the information is concentrated in the diagonal blocks. To approximate the kernel matrix by exploiting this clustering structure, we first present a simple Block Kernel Approximation (BKA) as follows. Given a good partition $\mathcal{V}_1, \ldots, \mathcal{V}_c$ of the data points, where each $\mathcal{V}_s$ is a subset of $\{1, \ldots, n\}$, BKA approximates the kernel matrix as:

$$G \approx \tilde{G} \equiv \begin{bmatrix} G^{(1,1)} & 0 & \ldots & 0 \\ 0 & G^{(2,2)} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & G^{(c,c)} \end{bmatrix}. \tag{2}$$

Here, $G^{(s,s)}$ denotes the kernel matrix for block $\mathcal{V}_s$ – note that this implies that diagonal blocks $\tilde{G}^{(s,s)} = G^{(s,s)}$ and all the off-diagonal blocks, $\tilde{G}^{(s,t)} = 0$ with $s \neq t$.

BKA is useful when η is large. By analyzing its approximation error, we now show that k-means in the input space can be used to capture the clustering structure for shift-invariant kernels. The approximation error equals $\|\tilde{G}-G\|_F^2 = \sum_{i,j} K(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 - \sum_{s=1}^c \sum_{i,j\in\mathcal{V}_s} K(\boldsymbol{x}_i, \boldsymbol{x}_j)^2$. Since the first term is fixed, minimizing the error $\|\tilde{G} - G\|_F^2$ is the same with maximizing the second term, the sum of squared within-cluster entries $D = \sum_{s=1}^c \sum_{i,j\in\mathcal{V}_s} K(\boldsymbol{x}_i, \boldsymbol{x}_j)^2$.

However, directly maximizing D will not give a useful partition – the maximizer will assign all the data into one cluster. The same problem occurs in graph clustering (Shi and Malik, 2000; von Luxburg, 2007). A common approach is to normalize D by each cluster's size $|\mathcal{V}_s|$. The resulting spectral clustering objective (also called ratio association) is:

$$D^{\text{kernel}}(\{\mathcal{V}_s\}_{s=1}^c) = \sum_{s=1}^{c} \frac{1}{|\mathcal{V}_s|} \sum_{i,j\in\mathcal{V}_s} K(\boldsymbol{x}_i, \boldsymbol{x}_j)^2. \tag{3}$$

Maximizing (3) usually yields a balanced partition, but the computation is expensive because we have to compute all the entries in G. In the following theorem, we derive a lower bound for $D^{\text{kernel}}(\{\mathcal{V}_s\}_{s=1}^c)$:

Theorem 1 *For any shift-invariant kernel that satisfies Assumption 1,*

$$D^{kernel}(\{\mathcal{V}_s\}_{s=1}^c) \geq \bar{C} - \eta^2 R^2 D^{kmeans}(\{\mathcal{V}_s\}_{s=1}^c) \tag{4}$$

where $\bar{C} = \frac{nf(0)^2}{2}$, R is a constant depending on the kernel function, and $D^{kmeans} \equiv \sum_{s=1}^c \sum_{i\in\mathcal{V}_s} \|\boldsymbol{x}_i - \boldsymbol{m}_s\|_2^2$ is the k-means objective function, where $\boldsymbol{m}_s = (\sum_{i\in\mathcal{V}_s} \boldsymbol{x}_i)/|\mathcal{V}_s|$, $s = 1, \cdots, c$, are the cluster centers.

Proof We use $\boldsymbol{u}$ to denote the unit vector in the direction of $\boldsymbol{x}_i - \boldsymbol{x}_j$ ($\boldsymbol{x}_i \neq \boldsymbol{x}_j$). By the mean value theorem, we have

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = g_{\boldsymbol{u}}(\eta\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2) = g_{\boldsymbol{u}}(0) + \eta g'_{\boldsymbol{u}}(s)\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2$$

for some $s \in (0, \eta\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2)$. By definition, $f(\boldsymbol{0}) = g_{\boldsymbol{u}}(0)$, so

$$f(\boldsymbol{0}) \leq K(\boldsymbol{x}_i, \boldsymbol{x}_j) + \eta R\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2, \tag{5}$$

$$\text{where } R := \sup_{\theta\in\mathbb{R}, \|\boldsymbol{v}\|=1} |g'_{\boldsymbol{v}}(\theta)|. \tag{6}$$

Squaring both sides of (5) we have

$$f(\boldsymbol{0})^2 \leq K(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 + \eta^2 R^2 \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 + 2K(\boldsymbol{x}_i, \boldsymbol{x}_j)(\eta R\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2).$$

From the classical arithmetic and geometric mean inequality, we can upper bound the last term by

$$2K(\boldsymbol{x}_i, \boldsymbol{x}_j)(\eta R\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2) \leq K(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 + \eta^2 R^2 \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2,$$

therefore

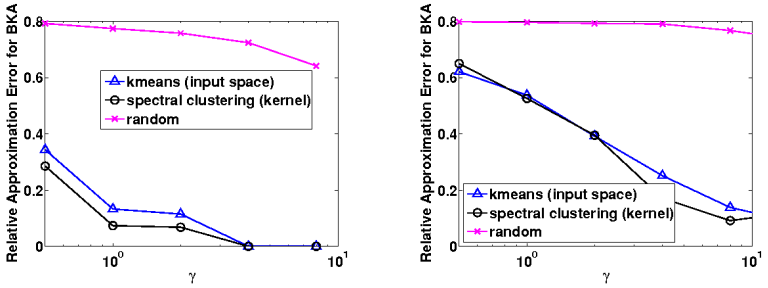
$$\frac{f(\boldsymbol{0})^2}{2} \leq K(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 + \eta^2 R^2 \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2. \tag{7}$$

Plugging (7) into (3), we have

$$\begin{aligned} \mathcal{D}^{\text{kernel}}(\{\mathcal{V}_s\}_{s=1}^c) &\geq \sum_{s=1}^{c} \frac{1}{|\mathcal{V}_s|} \sum_{i,j\in\mathcal{V}_s} \left(\frac{f(\boldsymbol{0})^2}{2} - \eta^2 R^2 \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 \right) \\ &\geq \frac{nf(\boldsymbol{0})^2}{2} - \eta^2 R^2 \sum_{s=1}^{c} \frac{1}{|\mathcal{V}_s|} \sum_{i,j\in\mathcal{V}_s} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2, \end{aligned}$$

which can be manipulated to prove the desired bound (4). ■

Interestingly, the right hand side of (4) can be maximized when the k-means objective function D^{kmeans} is minimized. Therefore, although optimal solutions for k-means and ratio association might be different (consider the two circles data, when each circle forms a cluster), Theorem 1 shows that conducting k-means in the input space will provide a reasonably good way to exploit the clustering structure of shift-invariant kernels, especially when it is infeasible to perform spectral clustering on G which might need precomputation of the entire kernel matrix. Figure 2 shows that the partition from k-means can often work as well as spectral clustering on G, which directly optimizes D^{kernel}, and both of them are much better than random partitions. One advantage of conducting k-means is that the time complexity of each iteration is $O(ndc)$, which is much less than computing the kernel when the dimensionality $d \ll n$.



(a) covtype (b) cadata

Figure 2: The Gaussian kernel approximation error of BKA using different ways to generate five partitions on 500 samples from covtype and cadata; on these data sets k-means in the input space performs similarly to spectral clustering on the kernel matrix, but is more efficient.

4.2 Speeding up Kernel SVM with BKA

In this section we will show how to use block kernel approximation(BKA) to divide kernel SVM problem into subproblems and significantly speed up the computation. Given a set of instance-label pairs $(\boldsymbol{x}_i, y_i), i = 1, \ldots, n, \boldsymbol{x}_i \in \mathbb{R}^d$ and $y_i \in \{1, -1\}$, the main task in training the kernel SVM is to solve the following quadratic optimization problem:

$$\min_{\boldsymbol{\alpha}} f(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T Q \boldsymbol{\alpha} - \boldsymbol{e}^T \boldsymbol{\alpha}, \text{ s.t. } 0 \le \boldsymbol{\alpha} \le C, \tag{8}$$

where $\boldsymbol{e}$ is the vector of all ones; C is the balancing parameter between loss and regularization in the SVM primal problem; $\boldsymbol{\alpha} \in \mathbb{R}^n$ is the vector of dual variables; and Q is an $n \times n$ matrix with $Q_{ij} = y_i y_j G_{ij}$, where $G_{ij} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the kernel value between i-th and j-th sample. Letting $\boldsymbol{\alpha}^*$ denote the optimal solution of (8), the decision value for a test data $\boldsymbol{x}$ can be computed by

$$\sum_{i=1}^{n} \alpha_i^* y_i K(\boldsymbol{x}, \boldsymbol{x}_i). \tag{9}$$

Due to high computation cost of directly solving kernel SVM, we can approximate the kernel matrix G by BKA to divide whole kernel SVM problem into subproblems, where each subproblem can be handled efficiently and independently.

To do this, we first partition the dual variables into k subsets $\{\mathcal{V}_1, \ldots, \mathcal{V}_k\}$, where $\{\mathcal{V}_1, \ldots, \mathcal{V}_k\}$ are partitions generated by performing kmeans on the data points, and then solve the respective subproblems independently

$$\min_{\boldsymbol{\alpha}_{(c)}} \frac{1}{2}(\boldsymbol{\alpha}_{(c)})^T Q_{(c,c)} \boldsymbol{\alpha}_{(c)} - \boldsymbol{e}^T \boldsymbol{\alpha}_{(c)}, \text{ s.t. } 0 \le \boldsymbol{\alpha}_{(c)} \le C, \tag{10}$$

where $c = 1, \ldots, k$, $\boldsymbol{\alpha}_{(c)}$ denotes the subvector $\{\alpha_i \mid i \in \mathcal{V}_c\}$ and $Q_{(c,c)}$ is the submatrix of Q with row and column indexes $\mathcal{V}_c$.

dataset	Number of training samples	Number of testing samples	d
ijcnn1	49,990	91,701	22
census	159,619	39,904	409
covtype	464,810	116,202	54

Table 1: Dataset statistics

The quadratic programming problem (8) has n variables, and takes at least $O(n^2)$ time to solve in practice. By dividing it into k subproblems (10) with equal sizes, the time complexity for solving the subproblems can be reduced to $O(k \cdot (\frac{n}{k})^2) = O(n^2/k)$. Moreover, the space requirement is also reduced from $O(n^2)$ to $O(n^2/k^2)$.

After computing all the subproblem solutions, we concatenate them to form an approximate solution for the whole problem $\bar{\boldsymbol{\alpha}} = [\bar{\boldsymbol{\alpha}}_{(1)}, \ldots, \bar{\boldsymbol{\alpha}}_{(k)}]$, where $\bar{\boldsymbol{\alpha}}_{(c)}$ is the optimal solution for the c-th subproblem.

4.3 Comparing BKA-SVM with Low-rank Kernel SVM Solvers(BKA-SVM)

We now compare block structure based kernel SVM solver–BKA-SVM with low-rank structure based kernel SVM solvers. All the experiments are conducted on an Intel 2.66GHz CPU with 8G RAM. We use 3 benchmark datasets as shown in Table 1. The thee datasets can be downloaded from `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets` or the UCI data repository. We use a random 80%-20% split for covtype, and the original training/testing split for other datasets.

4.3.1 Competing Methods

We include the following exact kernel SVM solvers (LIBSVM), approximate low-rank SVM solvers (LLSVM, FastFood) in our comparison:

1. LIBSVM: the implementation in the LIBSVM library (Chang and Lin, 2011) with a small modification to handle SVM without the bias term – we observe that LIBSVM has similar test accuracy with/without bias. We also include the results for using LIBSVM with random 1/5 subsamples on each dataset in Table 2.

2. LLSVM: improved Nyström method for nonlinear SVM by (Wang et al., 2011). We solve the resulting linear SVM problem by the dual coordinate descent solver in LIBLINEAR (Hsieh et al., 2008).

3. FastFood: use random Fourier features to approximate the kernel function (Le et al., 2013). We solve the resulting linear SVM problem by the dual coordinate descent solver in LIBLINEAR.

Both LLSVM and FastFood use low-rank representation to approximate kernel matrix so that to speed up solving kernel SVM.

	ijcnn1		census		covtype	
	$C = 32, \gamma = 2$		$C = 512, \gamma = 2^{-9}$		$c = 32, \gamma = 32$	
	time(s)	acc(%)	time(s)	acc(%)	time(s)	acc(%)
BKA-SVM	**5**	98.49	838	**94.72**	**436**	96.05
LIBSVM	115	**98.69**	2920	94.2	83631	**96.15**
LIBSVM (subsample)	6	98.24	641	93.2	5330	92.46
LLSVM	38	98.23	1212	92.8	4451	84.21
FastFood	87	95.95	851	91.6	8550	80.1

Table 2: Comparison on real datasets using the RBF kernel.

4.3.2 Parameter Setting

We consider the RBF kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\gamma\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2)$. We use same kernel function for both training and test phases. We chose the balancing parameter C and kernel parameter γ by 5-fold cross validation on a grid of points: $C = [2^{-10}, 2^{-9}, \ldots, 2^{10}]$ and $\gamma = [2^{-10}, \ldots, 2^{10}]$ for ijcnn1, census, and covtype. For BKA-SVM, we set the number of clusters to be 64 for these three datasets. There is a tradeoff between the number of clusters and prediction accuracy. If we increase the number of clusters, BKA-SVM will become faster, but the prediction accuracy will mostly decrease. On the other hand, if the number of clusters is set smaller, BKA-SVM can achieve higher accuracy (in most cases), while takes more time to train. The following are parameter settings for other methods in Table 2: the rank is set to be 3000 in LLSVM; number of Fourier features is 3000 in Fastfood[1]; the tolerance in the stopping condition for LIBSVM is set to 10^{-3} (the default setting of LIBSVM).

Tables 2 present time taken and test accuracies. Experimental results show that the BKA-SVM achieves near-optimal test performance. Also we observe that BKA-SVM performs better than low-rank approximation based methods for kernel SVM problem showing the benefit of using block structure of kernel matrix.

We can see that BKA exploits the block structure of kernel matrix, and can speed up the training of kernel SVM. As shown in Tandon et al. (2016), BKA can also be used for speeding up kernel ridge regression problem. About the memory requirement, which is the main theme of this paper, BKA takes $O(\frac{n^2}{k})$ memory to approximate the kernel matrix, while popular low-rank based kernel approximation methods are more memory efficient, and only need linear memory to represent the kernel matrix.

5. Memory Efficient Kernel Approximation

There are two main drawbacks of the BKA approach: (i) it ignores all off-diagonal blocks, which results in large error when η is small (as seen in Figure 1(c)); (ii) for large-scale kernel approximation, it is too expensive to compute and store all the diagonal block entries. To

1. In Fastfood we control the number of blocks so that number of Fourier features is close to 3000 for each dataset.

overcome these two drawbacks, we propose to use low-rank representation for each block in the kernel matrix.

5.1 Low Rank Structure of Each Block

To motivate the use of low-rank representation in our proposed method, we first present the following bound:

Theorem 2 *Given data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$, and a partition $\mathcal{V}_1, \ldots, \mathcal{V}_c$, and assume f is Lipschitz continuous, then for any s, t ($s = t$ or $s \neq t$)*

$$\|G^{(s,t)} - G_k^{(s,t)}\|_F \leq 4Ck^{-1/d}\sqrt{|\mathcal{V}_s||\mathcal{V}_t|}\min(r_s, r_t),$$

where $G_k^{(s,t)}$ is the best rank-k approximation to $G^{(s,t)}$; C is the Lipschitz constant of the shift-invariant function f; r_s is the radius of the s-th cluster.

Proof To prove this theorem, we use the ϵ-net theorem in (Cucker and Smale, 2001), which states that when all the data $\boldsymbol{x}_i \in \mathbb{R}^d$, $i = 1, \cdots, n$ are in a ball with radius r, there exist $T = (\frac{4r}{\bar{r}})^d$ balls of radius $\bar{r}$ that cover all the data points. If we set T to be k, then $\bar{r} = k^{-1/d}4r$.

Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{n_s}$ be the data points in the s-th cluster, and let $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{n_t}$ be the data points in the t-th cluster, and $n_s = |\mathcal{V}_s|, n_t = |\mathcal{V}_t|$. Our goal is to show that $G^{(s,t)}$ is low-rank, where $G_{i,j}^{(s,t)} = K(\boldsymbol{x}_i, \boldsymbol{y}_j)$. Assume r_t is the radius of the t-th cluster, therefore we can find k balls with radius $\bar{r} = k^{-1/d}4r_t$ to cover $\{\boldsymbol{y}_j\}_{j=1}^{n_t}$.

Assume centers of the balls for t-th cluster are $\boldsymbol{m}_1, \boldsymbol{m}_2, \ldots, \boldsymbol{m}_k$, then we can form a low-rank matrix $\bar{G}^{(s,t)} = \bar{U}\bar{V}^T$, where for all $i = 1, \ldots, n_s, j = 1, \ldots, n_t$, and $q = 1, \ldots, k$,

$$\bar{U}_{i,q} = K(\boldsymbol{x}_i, \boldsymbol{m}_q) \text{ and } \bar{V}_{j,q} = \begin{cases} 1 & \text{if } \boldsymbol{y}_j \in Ball(\boldsymbol{m}_q), \\ 0 & \text{otherwise.} \end{cases}$$

Assume $\boldsymbol{y}_j$ is in ball q, then

$$\begin{aligned}(G_{ij}^{(s,t)} - \bar{G}_{ij}^{(s,t)})^2 &= (f(\boldsymbol{x}_i - \boldsymbol{y}_j) - f(\boldsymbol{x}_i - \boldsymbol{m}_q))^2 \\ &\leq C^2\|(\boldsymbol{x}_i - \boldsymbol{y}_j) - (\boldsymbol{x}_i - \boldsymbol{m}_q)\|^2 \\ &= C^2\|\boldsymbol{y}_j - \boldsymbol{m}_q\|_2^2 \\ &\leq C^2\bar{r}^2.\end{aligned}$$

Therefore, if $(G^{(s,t)})^*$ is the best rank-k approximation for $G^{(s,t)}$, then

$$\|G^{(s,t)} - (G^{(s,t)})^*\|_F \leq \|G^{(s,t)} - \bar{G}^{(s,t)}\|_F \leq Ck^{-1/d}4r_t\sqrt{n_s n_t}. \tag{11}$$

Similarly, by covering s-th cluster $\{\boldsymbol{x}_i\}_{i=1}^{n_s}$ with k balls we can get the following inequality:

$$\|G^{(s,t)} - (G^{(s,t)})^*\|_F \leq Ck^{-1/d}4r_s\sqrt{n_s n_t}. \tag{12}$$

Combining (11) and (12) gives the results. ■

16	14	13	7	7
14	29	13	9	9
13	13	20	10	10
7	9	10	29	11
7	9	10	11	28

(a) k-means clustering.

139	99	101	44	45
99	116	86	43	44
101	86	131	46	47
44	43	46	47	45
45	44	47	45	49

(b) a random partition.

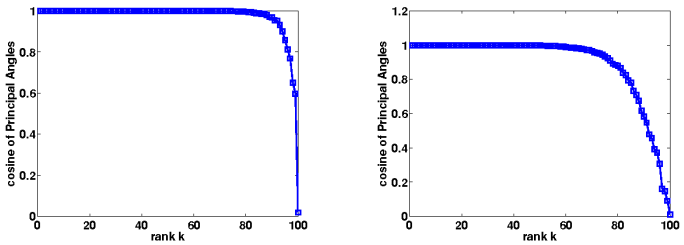
Table 3: Rank of each of the 5 blocks (from a subsampled ijcnn1 data set) using different partition strategies: (a) by k-means clustering; (b) by a random partition.

Theorem 2 suggests that each block(diagonal or off-diagonal block) of the kernel matrix will be low-rank if we find the partition by k-means in the input space and the radius of the cluster is small. In the following we present empirical confirmation of this result. In Table 3, we present the numerical rank of each block, where numerical rank for a m by n matrix A is defined as the number of singular values with magnitude larger than $\max(n,m)\|A\|_2\delta$ where δ is a small tolerance 10^{-6}. We sample 4000 data points from the ijcnn1 dataset and generate 5 clusters by k-means and random partition. Table 3a shows the numerical rank for each block using k-means, while Table 3b shows the numerical rank for each block when the partitions are random. We observe that by using k-means, the rank of each block is fairly small.

5.2 Memory Efficient Kernel Approximation (MEKA)

Based on the above observation, we propose a fast and memory efficient scheme to approximate shift-invariant kernel matrices. As suggested by Theorem 2, each block tends to be low-rank after k-means clustering; thus we can form a rank-k approximation for each of the c^2 blocks separately to achieve low error; however, this approach would require $O(cnk)$ memory, which can be prohibitive. Therefore, our proposed method first performs k-means clustering, and after rearranging the matrix according to clusters, it computes the low-rank basis only for diagonal blocks (which are more dominant than off-diagonal blocks) and uses them to approximate off-diagonal blocks. Empirically, we observe that the principal angles between the dominant singular subspaces of diagonal block and off-diagonal block are small (as shown in Figure 3). In Figure 3, we randomly sampled 1000 data points from the covtype and ijcnn1 datasets and generated 5 clusters by k-means for each dataset. The blue line shows the cosines of the principal angles between the dominant singular subspace of a diagonal block $G^{(s,s)}$ and that of an off-diagonal block $G^{(s,t)}$ for different ranks k, where s and t are randomly chosen. We can observe that most of the cosines are close to 1, showing that there is substantial overlap between the dominant singular subspaces of the diagonal and off-diagonal block.

By using our proposed approach, we focus on diagonal blocks, and spend less effort on the off-diagonal blocks. Assume the rank-k_s approximation of the s^{th} diagonal block is



(a) covtype (b) ijcnn1

Figure 3: The cosines of the principal angles between the dominant singular subspaces of diagonal block and off-diagonal block of a Gaussian kernel ((a): $\gamma = 0.1$ and 1000 random samples from covtype; (b): $\gamma = 1$ and 1000 random samples from ijcnn1) with respect to different ranks. The cosines of the principal angles are close to 1 showing that two subspaces are similar.

$W^{(s)}L^{(s,s)}(W^{(s)})^T$, we form the following memory-efficient kernel approximation:

$$\tilde{G} = WLW^T, \tag{13}$$

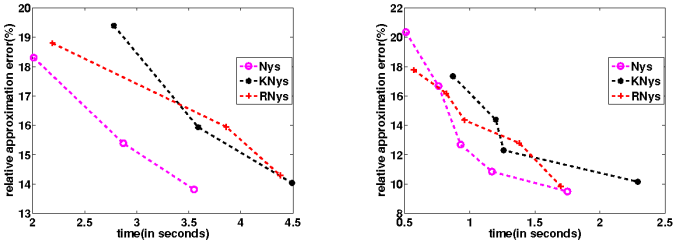
where W is a diagonal matrix as

$$W \equiv \begin{bmatrix} W^{(1)} & 0 & \dots & 0 \\ 0 & W^{(2)} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & W^{(c)} \end{bmatrix}; \tag{14}$$

and L is a "link" matrix consisting of c^2 blocks, where each $k_s \times k_t$ block $L^{(s,t)}$ captures the interaction between the s^{th} and t^{th} clusters. For now, let us first assume $k_s = k\ \forall s$ (we will discuss different strategies to choose k_s later). Note that if we were to restrict L to be a block diagonal matrix, $\tilde{G}$ would still be a block diagonal approximation of G. However, we consider the more general case that L is dense. In this case, each off-diagonal block $G^{(s,t)}$ is approximated as $W^{(s)}L^{(s,t)}(W^{(t)})^T$, and this approximation is memory efficient as only $O(k^2)$ additional memory is required to represent the (s,t) off-diagonal block. If a rank-k approximation is used within each cluster, then the generated approximation has rank ck, but only needs a total of $O(nk + (ck)^2)$ storage.

Computing $W^{(s)}$. Since we aim to deal with dense kernel matrices of huge size, we use the standard Nyström approximation to compute low-rank "basis" for each diagonal block. When applying the standard Nyström method to a $n_s \times n_s$ block $G^{(s,s)}$, we sample m columns from $G^{(s,s)}$, evaluate their kernel values, compute the rank-k pseudo-inverse of an $m \times m$ matrix, and form $G^{(s,s)} \approx W^{(s)}L^{(s,s)}(W^{(s)})^T$. The time required per block is $O(n_s m(k+d) + m^3)$, and thus our method requires a total of $O(nm(k+d) + cm^3)$ time to form W. We can replace Nyström by any other low-rank approximation method discussed

in Section 2. In Figure 4 we compare using standard Nyström(Nys)(Williams and Seeger, 2001), k-means Nyström(KNys)(Zhang and Kwok, 2010), and Nyström with randomized SVD(RNys)(Li, Kwok, and Lu, 2010) to generate $W^{(s)}$ on both ijcnn1 and cadata datasets. As shown in Figure 4, we observe that the standard Nyström method combined with MEKA gives excellent performance.



(a) ijcnn1 (b) cadata

Figure 4: Comparison of using Nys, KNys, and RNys to obtain the basis $W^{(s)}$ for diagonal blocks in MEKA on ijcnn1 and cadata datasets. The x-axis shows the computation time and y-axis shows the relative kernel approximation error(%).

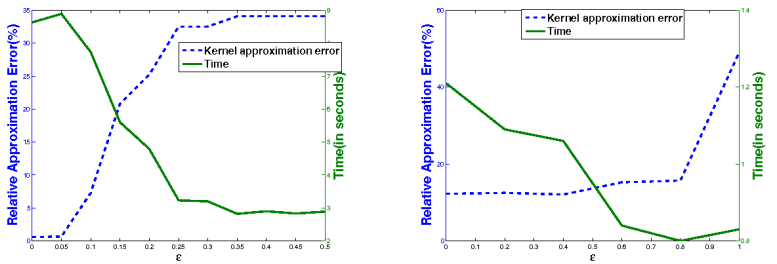
Computing $L^{(s,t)}$. The optimal least squares solution for $L^{(s,t)}(s \neq t)$ is the minimizer of the local approximation error $\|G^{(s,t)} - W^{(s)}L^{(s,t)}(W^{(t)})^T\|_F$. However, forming the entire $G^{(s,t)}$ block can be time consuming. For example, computing the whole kernel matrix for mnist2m with 2 million data points takes more than a week. Therefore, to compute $L^{(s,t)}$, we propose to randomly sample a $(1+\rho)k \times (1+\rho)k$ submatrix $\hat{G}^{(s,t)}$ from $G^{(s,t)}$, and then find $L^{(s,t)}$ that minimizes the error on this submatrix. If the row/column index set for the subsampled submatrix $\hat{G}^{(s,t)}$ in $G^{(s,t)}$ is v_s/v_t, then $L^{(s,t)}$ can be computed in closed form:

$$L^{(s,t)} = ((W^{(s)}_{v_s})^T W^{(s)}_{v_s})^{-1}(W^{(s)}_{v_s})^T \hat{G}^{(s,t)} W^{(t)}_{v_t}((W^{(t)}_{v_t})^T W^{(t)}_{v_t})^{-1},$$

where $W^{(s)}_{v_s}$ and $W^{(t)}_{v_t}$ are formed by the rows in $W^{(s)}$ and $W^{(t)}$ with row index sets v_s and v_t respectively.

Since there are only k^2 variables in $L^{(s,t)}$, we do not need too many samples for each block, and the time to compute $L^{(s,t)}$ is $O((1+\rho)^3k^3)$. In practice, we observe that setting ρ to be 2 or 3 is enough for a good approximation, so the time complexity is $O(k^3)$. Empirically, many values in the off-diagonal blocks are close to zero, and only a few of them have large values as shown in Figure 1. Based on this observation, we further propose a thresholding technique to reduce the time for storing and computing $L^{(s,t)}$. Since the distance between cluster centers is a good indicator for the values in an off-diagonal block, we can set the whole block $L^{(s,t)}$ to 0 if $K(\boldsymbol{m}_s, \boldsymbol{m}_t) \leq \epsilon$ for some thresholding parameter $\epsilon > 0$, where $\boldsymbol{m}_s$ and $\boldsymbol{m}_t$ are the k-means centroid for the s-th and t-th cluster respectively. Obviously, to choose ϵ, we need to achieve a balance between speed and accuracy. When ϵ is small, we will approximate more off-diagonal blocks; while when ϵ is large, we will set more off-diagonal blocks to be 0, but increase the approximation error. We test the influence

of thresholding parameter ϵ on the ijcnn1 and cadata data in Figure 5. When ϵ is large, although MEKA yields higher approximation error(because it omits more off-diagonal information), it is faster. On the other hand, for small ϵ, when more off-diagonal information is considered, we notice an increase in time and smaller in approximation error. In practice, we need to use cross-validation to select ϵ.

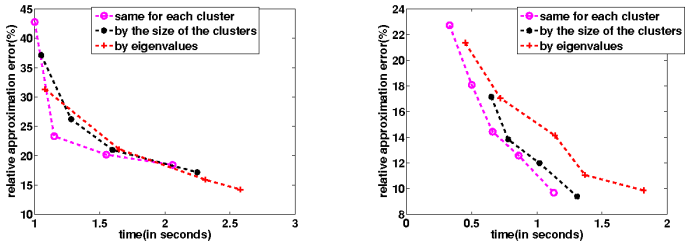


(a) ijcnn1 (b) cadata

Figure 5: Time (in seconds) and kernel approximation quality of MEKA when varying the thresholding parameter ϵ for setting off-diagonal blocks in L to be zero.

Choosing the rank k_s for each cluster. We need to decide the rank for the $s^{th}(s = 1, \cdots, c)$ cluster, k_s, which can be done in different ways: (i) the same k for all the clusters; (ii) k_s is proportional to the size of s^{th} cluster; (iii) eigenvalues based approach. For (iii), suppose $M^{(s)}$ is the $m_s \times m_s$ matrix consisting of the intersection of m_s sampled columns in $G^{(s,s)}$, and M is the $cm \times cm(\sum_{s=1}^{c} m_s = cm)$ block-diagonal matrix with $M^{(s)}$ as diagonal block. We can choose k_s such that the set of top-ck eigenvalues of M is the union of the eigenvalues of $M^{(s)}$ in each cluster, that is, $[\sigma_1(M), \ldots, \sigma_{ck}(M)] = \cup_{s=1}^{c}[\sigma_1(M^{(s)}), \ldots, \sigma_{k_s}(M^{(s)})]$. To use (iii), we can oversample points in each cluster, e.g., sample $2k$ points from each cluster, perform eigendecompostion of a $2k \times 2k$ kernel matrix, sort the eigenvalues from c clusters, and finally select the top-ck eigenvalues and their corresponding eigenvectors. Comparing these three strategies, (i) achieves lower memory usage and is fastest, and (ii) is more accurate than (i) with more memory usage, while (iii) is slowest but achieves lower error for diagonal blocks. In Figure 6, we compare these three sampling strategies to choose k_s on ijcnn1 and cadata datasets. It is shown that these three methods perform similarly well and choosing k_s to be same for each cluster performs sightly better than by the size of each cluster and singular values based approach. In the experiment, we set all the clusters to have the same rank k. We show that this simple choice of k_s already outperforms state-of-the-art kernel approximation methods.

We are now ready to present our main algorithm in Algorithm 1. In Table 4, we compare the time and space complexity for our method with SVD, standard Nyström, and RKS. We can see that MEKA is more memory efficient. For the time complexity, both T_L (time for computing off-diagonal L) and T_C (time for clustering) are small as (1) we use thresholding to force some $L^{(s,t)}$ blocks to be zero, and perform least squares on small blocks, which



(a) ijcnn1 (b) cadata

Figure 6: Comparison of different strategies to choose the rank k_s of each cluster in MEKA on ijcnn1 and cadata datasets.

Method	Storage	Rank	Time Complexity
RKS (Rahimi and Recht, 2008)	$O(cnk)$	ck	$O(cnkd)$
Nyström(Williams and Seeger, 2001)	$O(cnk)$	ck	$O(cnm(ck+d)+(cm)^3)$
SVD	$O(cnk)$	ck	$O(n^3+n^2d)$
MEKA	$O(nk+(ck)^2)$	ck	$O(nm(k+d)+cm^3+T_L+T_C)$

Table 4: Memory and time analysis of various kernel approximation methods, where T_L is the time to compute the matrix L and T_C is the time for clustering in MEKA.

means T_L can at most be $O(\frac{1}{2}c^2k^3)$; (2)T_C is proportional to the number of samples. For a large dataset, we sample 20000 points for k-means, and thus the clustering is more efficient than working on the entire data set.

Algorithm 1 Memory Efficient Kernel Approximation (MEKA)

Input : Data points $\{(\boldsymbol{x}_i)\}_{i=1}^n$, scaling parameter γ, rank k, and no. of clusters c.

Output: The rank-ck approximation $\tilde{G} = WLW^T$ using $O(nk+(ck)^2)$ space

Generate the partition $\mathcal{V}_1, \ldots, \mathcal{V}_c$ by k-means;

for $s = 1, \ldots, c$ **do**

  Perform the rank-k approximation $G^{(s,s)} \approx W^{(s)}L^{(s,s)}(W^{(s)})^T$ by standard Nyström;

end

forall $(s,t)(s \neq t)$ **do**

  Sample a submatrix $\bar{G}^{(s,t)}$ from $G^{(s,t)}$ with row index set v_s and column index set v_t;

  Form $W_{v_s}^{(s)}$ by selecting the rows in $W^{(s)}$ according to index set v_s;

  Form $W_{v_t}^{(t)}$ by selecting the rows in $W^{(t)}$ according to index set v_t;

  Solve the least squares problem: $\bar{G}^{(s,t)} \approx W_{v_s}^{(s)}L^{(s,t)}(W_{v_t}^{(t)})^T$ to obtain $L^{(s,t)}$;

end

Dataset	T_C	T_W	T_L
pendigit	0.05	0.69	0.35
ijcnn1	0.15	1.27	0.84
covtype	1.83	9.82	12.23

Table 5: Time (in seconds) for each step of MEKA, where T_W is the time to compute low-rank approximation for the diagonal block matrices; T_L is the time to form the "link" matrix L; T_C is the time for performing k-means clustering.

In Table 5, we show the time cost for each step of MEKA on pendigit, ijcnn1, and covtype datasets. The execution time of our proposed algorithm mainly consists of three parts:(1) time for performing k-means clustering (T_C), (2) time for forming the "basis" W from the diagonal blocks (T_W), (3) time to compute the link matrix L from off-diagonal blocks (T_L). From Table 5, we observe that compared with T_W and T_L, T_C is fairly small and T_W dominates the whole process in most cases. For covtype data set, since we choose c to be large, T_L is sightly larger than T_W. We will analyze T_W, T_L, and T_C for different c in the experiment part.

5.3 Analysis

We now bound the approximation error for our proposed method. We show that when $\sigma_{k+1} - \sigma_{ck+1}$ is large, where σ_{k+1} and σ_{ck+1} are the $k+1^{st}$ and $ck+1^{st}$ singular values of G respectively, and entries in off-diagonal blocks are small, MEKA has a better approximation error bound compared to standard Nyström that uses similar storage.

Theorem 3 *Let Δ denote a matrix consisting of all off-diagonal blocks of G, so $\Delta^{(s,t)} = G^{(s,t)}$ for $s \neq t$ and all zeros when $s = t$. We sample cm points from the dataset uniformly at random without replacement and split them according to the partition from k-means, such that each cluster has m_s benchmark points and $\sum_{s=1}^c m_s = cm$. Let G_{ck} be the best rank-ck approximation of G, and $\tilde{G}$ be the rank-ck approximation from MEKA. Suppose we choose the rank k_s for each diagonal block using the eigenvalue based approach as mentioned in Section 5.1, then with probability at least $1-\delta$, the following inequalities hold for any sample of size cm:*

$$\|G-\tilde{G}\|_2 \leq \|G-G_{ck}\|_2 + \frac{1}{\sqrt{c}}\frac{2n}{\sqrt{m}}G_{\max}(1+\theta) + 2\|\Delta\|_2,$$

$$\|G-\tilde{G}\|_F \leq \|G-G_{ck}\|_F + \left(\frac{64k}{m}\right)^{\frac{1}{4}} nG_{\max}(1+\theta)^{\frac{1}{2}} + 2\|\Delta\|_F$$

where $\theta = \sqrt{\frac{n-m}{n-0.5}\frac{1}{\beta(m,n)}\log\frac{1}{\delta}}d_{\max}^G/G_{\max}^{\frac{1}{2}}$; $\beta(m,n) = 1 - \frac{1}{2\max\{m,n-m\}}$; $G_{\max} = \max_i G_{ii}$; *and* $d_{\max}^G$ *represents the distance* $\max_{ij}\sqrt{G_{ii}+G_{jj}-2G_{ij}}$.

Proof Let B denote the matrix formed by the diagonal block of G, that is,

$$B \equiv \begin{bmatrix} G^{(1)} & 0 & \dots & 0 \\ 0 & G^{(2)} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & G^{(c)} \end{bmatrix}. \tag{15}$$

According to the definition of Δ, $G = B + \Delta$. In MEKA, the error $\|\tilde{G} - G\|_2$ consists of two components,

$$\|\tilde{G} - G\|_2 = \|\tilde{B} - B + (\tilde{\Delta} - \Delta)\|_2 \leq \|\tilde{B} - B\|_2 + \|\tilde{\Delta} - \Delta\|_2 \tag{16}$$

where $\tilde{B}$ and $\tilde{\Delta}$ are the approximations for B and Δ in MEKA respectively.

Let us first consider the error in approximating the diagonal blocks $\|\tilde{B} - B\|_2$. Since we sample cm benchmark points from n data points uniformly at random without replacement and distribute them according to the partition coming from k-means, the s-th cluster now has m_s benchmark points with $\sum_{s=1}^{s=c} m_s = cm$. For the s-th diagonal block $G^{(s)}$, we will perform the rank-k_s approximation using standard Nyström, so we have $G^{(s)} \approx E^{(s)}(M_{k_s}^{(s)})^+ E^{(s)}$, where $E^{(s)}$ denotes the matrix formed by m_s sampled columns from $G^{(s)}$ and $M_{k_s}^{(s)}$ is a $m_s \times m_s$ matrix consisting of the intersection of sampled m_s columns.

Suppose we use the singular value based approach to choose k_s for s-th cluster as described in Section 5.1, and

$$M_{ck}^{+} equiv \begin{bmatrix} (M_{k_1}^{(1)})^+ & 0 & \dots & 0 \\ 0 & (M_{k_2}^{(2)})^+ & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & (M_{k_c}^{(c)})^+, \end{bmatrix} \tag{17}$$

where M is the $cm \times cm$ block diagonal matrix that consists of the intersection of the sampled cm columns. Then we can see that approximating the diagonal blocks B is equivalent to directly performing standard Nyström on B by sampling cm benchmark points uniformly at random without replacement to achieve the rank-ck approximation. The standard Nyström's norm-2 and Frobenius error bound are given in (Kumar et al., 2009), so $\|B - \tilde{B}\|_2$ can be bounded with probability at least $1 - \delta$ as

$$\|B - \tilde{B}\|_2 \leq \|B - B_{ck}\|_2 + \frac{2n}{\sqrt{cm}} B_{\max}[1 + \sqrt{\frac{n - cm}{n - 0.5} \frac{1}{\beta(cm, n)} \log \frac{1}{\delta}} d_{\max}^B / B_{\max}^{\frac{1}{2}}],$$

where B_{ck} denotes the best rank-ck approximation to B; $B_{\max} = \max_i B_{ii}$; $d_{\max}^B$ represents the distance $\max_{ij} \sqrt{B_{ii} + B_{jj} - 2B_{ij}}$.

To bound $\|\tilde{\Delta} - \Delta\|_2$, recall that some off-diagonal blocks in MEKA are set to 0 by thresholding and $\mathbf{0}$ is one special solution of least squares problem to compute $L^{(s,t)}$, we have $\|\tilde{\Delta} - \Delta\|_2 \leq \|\Delta\|_2$.

Furthermore, according to perturbation theory (Stewart and Ji-Guang, 1990), we have

$$\|B - B_{ck}\|_2 \leq \|G - G_{ck}\|_2 + \|\Delta\|_2. \tag{18}$$

The inequality in (16) combined with (18) gives a bound on $\|\tilde{G} - G\|_2$ as,

$$\begin{aligned} \|\tilde{G} - G\|_2 &\leq \|B - B_{ck}\|_2 + \|\Delta\|_2 + \frac{2n}{\sqrt{cm}} B_{\max}[1 + \sqrt{\frac{n - cm}{n - 0.5} \frac{1}{\beta(cm, n)} \log \frac{1}{\delta}} d_{\max}^B / B_{\max}^{\frac{1}{2}}] \\ &\leq \|G - G_{ck}\|_2 + 2\|\Delta\|_2 + \frac{2n}{\sqrt{cm}} B_{\max}[1 + \sqrt{\frac{n - cm}{n - 0.5} \frac{1}{\beta(cm, n)} \log \frac{1}{\delta}} d_{\max}^B / B_{\max}^{\frac{1}{2}}] \\ &\leq \|G - G_{ck}\|_2 + 2\|\Delta\|_2 + \frac{2n}{\sqrt{cm}} G_{\max}[1 + \sqrt{\frac{n - cm}{n - 0.5} \frac{1}{\beta(cm, n)} \log \frac{1}{\delta}} d_{\max}^G / G_{\max}^{\frac{1}{2}}] \\ &\leq \|G - G_{ck}\|_2 + 2\|\Delta\|_2 + \frac{1}{\sqrt{c}} \frac{2n}{\sqrt{m}} G_{\max}[1 + \sqrt{\frac{n - m}{n - 0.5} \frac{1}{\beta(m, n)} \log \frac{1}{\delta}} d_{\max}^G / G_{\max}^{\frac{1}{2}}], \end{aligned}$$

where G_{ck} denotes the best rank-ck approximation to G; $G_{\max} = \max_i G_{ii}$; $d_{\max}^G$ represents the distance $\max_{ij} \sqrt{G_{ii} + G_{jj} - 2G_{ij}}$. The third inequality is because $G = B + \Delta$, $B_{\max} \leq G_{\max}$ and $d_{\max}^B \leq d_{\max}^G$. The last inequality is because $n \gg m$ and $n \gg cm$.

Similarly by using perturbation theory and upper bounds for the Frobenius error of standard Nyström, the result follows. ■

When $k_s (s = 1, \cdots, c)$ is balanced (meaning k_s is approximately the same for each cluster) and n is large, MEKA provides a rank-ck approximation using roughly the same amount of storage as rank-k approximation by standard Nyström. Interestingly, from Theorem 3, if $\|G - G_k\|_2 - \|G - G_{ck}\|_2 \geq 2\|\Delta\|_2$, then

$$\|G - \tilde{G}\|_2 \leq \|G - G_k\|_2 + \frac{1}{\sqrt{c}} \frac{2n}{\sqrt{m}} G_{\max}(1 + \theta).$$

The second term in the right hand side of above inequality is only $\frac{1}{\sqrt{c}}$ of that in the spectral norm error bound for standard Nyström that uniformly samples m columns without replacement in G to obtain the rank-k approximation as shown in (Kumar, Mohri, and Talwalkar, 2009). Thus, if there is a large enough gap between σ_{k+1} and σ_{ck+1}, the error bound for our proposed method is better than standard Nyström that uses similar storage. Furthermore, when γ is large, G tends to have better clustering structure, suggesting in Theorem 3 that $\|\Delta\|$ is usually quite small. Note that when using the same rank k for all the clusters, the above bound can be worse because of some extreme cases, e.g., all the top-ck eigenvalues are in the same cluster. In practice we do not observe those extreme situations. We also want to mention that both $\|\Delta\|_F$ and $\|\Delta\|_2$ will be affected by the number of clusters c.

6. Experimental Results

In this section, we empirically demonstrate the benefits of our proposed method, MEKA on various data sets[2] that are listed in Table 6. Experiment results here are mainly based on

2. All the datasets are downloaded from `www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets` and UCI repository (Bache and Lichman, 2013). Note that the census datasets in Table 1 and Table 6 come from different sources.

Dataset	n	d	Dataset	n	d
wine	6,497	11	census	22,784	137
cpusmall	8,192	12	ijcnn1	49,990	22
pendigit	10,992	16	covtype	581,012	54
cadata	20,640	8	mnist2m	2,000,000	784

Table 6: Data set statistics (n: number of samples; d: dimension of samples).

the Gaussian kernel, but we observe similar behavior on other shift-invariant kernels (see Section 6.2). Note that the same kernel function is used in both training and test phases. We compare our method with six state-of-the-art kernel approximation methods:

1. The standard Nyström method (denoted by Nys)(Williams and Seeger, 2001). In the experiment, we uniformly sample $2k$ columns of G without replacement, and run Nyström for rank-k approximation.

2. Kmeans Nyström (denoted by KNys)(Zhang and Kwok, 2010), where the landmark points are the cluster centroids. As suggested in (Zhang et al., 2012), we sample 20000 points for clustering when the total number of data samples is larger than 20000.

3. Random Kitchen Sinks (denoted by RKS)(Rahimi and Recht, 2008), which approximates the shift-invariant kernel based on its Fourier transform.

4. Fastfood with "Hadamard features" (denoted by fastfood)(Le, Sarlos, and Smola, 2013).

5. Ensemble Nyström (denoted by ENys) (Kumar, Mohri, and Talwalkar, 2009). Due to concern for the computation cost, we set the number of "experts" in ENys 3.

6. Nyström using randomized SVD (denoted by RNys)(Li, Kwok, and Lu, 2010). We set the number of power iterations $q = 1$ and oversampling parameter $p = 10$.

We compare all the methods on two different tasks: kernel low-rank approximation and kernel ridge regression. We do not compare with BKA in this section, because (1) the approximation error of BKA is large when γ is small (as shown in Figure 1); (2) BKA is time consuming because it needs to compute all the diagonal blocks' kernel values, which needs $O(\frac{n^2 d}{c})$ time; (3) BKA is not memory efficient to approximate the kernel matrix, because it needs $O(\frac{n^2}{c})$ space to store the approximation. All the experiments are conducted on a machine with an Intel Xeon X5440 2.83GHz CPU and 32G RAM.

6.1 Kernel Approximation Quality

We now compare the kernel approximation quality for the above methods.

Main results. The kernel approximation results are shown in Figure 7 and Table 7. We use relative kernel approximation error $\|G - \tilde{G}\|_F / \|G\|_F$ to measure the quality. We randomly sampled 20000 rows of G to evaluate the relative approximation error for ijcnn1 and covtype. In Figure 7, we show the kernel approximation performance of different methods by varying k and γ. The rank (k) varies from 100 to 600 for ijcnn1 and covtype

Dataset	k	γ	c	Nys	RNys	KNys	ENys	RKS	fastfood	MEKA
pendigit	128	2	5	0.1325	0.1361	0.0828	0.2881	0.4404	0.4726	**0.0811**
ijcnn1	128	1	10	0.0423	0.0385	0.0234	0.1113	0.2972	0.2975	**0.0082**
covtype	256	10	15	0.3700	0.3738	0.2752	0.5646	0.8825	0.8920	**0.1192**

Table 7: Comparison of approximation error of our proposed method with six other state-of-the-art kernel approximation methods on real datasets, where γ is the Gaussian scaling parameter; c is the number of clusters in MEKA; k is the rank of each diagonal-block in MEKA and the rank of the approximation for six other methods. Note that for a given k, every method has roughly the same amount of memory. All results show relative kernel approximation errors for each k.

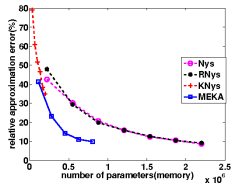
and from 20 to 200 for the pendigit data. Figure 7 shows that our proposed approximation scheme always achieves lower error with less time and memory. The main reason is that using similar amount of time and memory, our method aims to approximate the kernel matrix by a rank-ck approximation, while all other methods are only able to form a rank-k approximation.

In Table 7, we fix the rank k and γ, so that each method has the same memory usage of low-rank representation, and compare MEKA with them in terms of relative approximation error. As it can be seen, under the same amount of memory, our proposed method consistently yields lower approximation error than other methods.

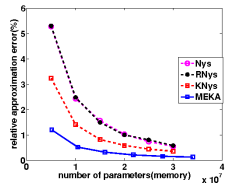
Also as we can see from Table 7 and Figure 8, Nyström based methods perform much better than random features based methods (including RKS and Fastfood here) in terms of kernel approximation quality. Therefore we do not show their performance in Figure 7, so that we could see the difference between MEKA and other Nyström based methods.

Robustness to the Gaussian scaling parameter γ. To show the robustness of our proposed algorithm with different γ as explained in Section 3, we test its performance on the ijcnn1 (Figure 9a), cadata (Figure 9b) and sampled covtype datasets (Figure 1c). The relative approximation errors for different γ values are shown in the figures using a fixed amount of memory. For large γ, the kernel matrix tends to have a block structure, so our proposed method yields lower error than other methods. The gap becomes larger as γ increases. Interestingly, Figure 1c shows that the approximation error of MEKA is superior to even the exact SVD, as it is much more memory efficient. Even for small γ where the kernel exhibits low-rank structure, our proposed method performs better than Nyström based methods, suggesting that it can get the low-rank structure of the kernel matrix.

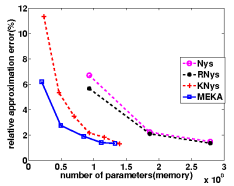
Robustness to the number of clusters c. Compared with Nyström, one main extra parameter for our method is the number of clusters c. In Figure 10, we test our method with different values of c on ijcnn1 dataset. In Figure 10a, we show how the approximation error changes when c is varied from 5 to 25 (with rank $k = 100$ in each cluster). The memory usage is $nk + (ck)^2$, so the storage increases as c increases. For a fair comparison, we increase the rank of other methods as c increases, so that all the methods use the same amount of memory. Figure 10a shows that the performance of our proposed method for varying choices of c. We can observe that under different c, MEKA always performs better than others. Furthermore, in Figure 10b we show the time cost for each step of MEKA on



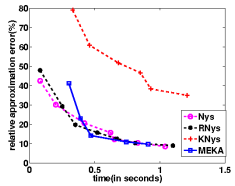
(a) pendigit ($\gamma = 2$), memory vs approx. error.



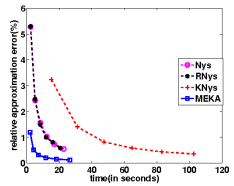
(b) ijcnn1 ($\gamma = 1$), memory vs approx. error.



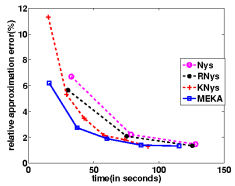
(c) covtype ($\gamma = 1$), memory vs approx. error.



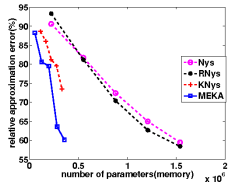
(d) pendigit ($\gamma = 2$), time vs approximation error.



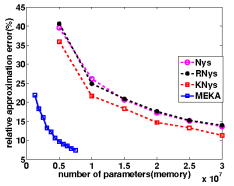
(e) ijcnn1 ($\gamma = 1$), time vs approximation error.



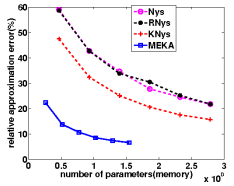
(f) covtype ($\gamma = 1$), time vs approximation error.



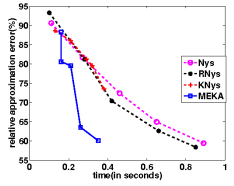
(g) pendigit ($\gamma = 10$), memory vs approximation error.



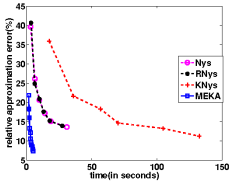
(h) ijcnn1 ($\gamma = 10$), memory vs approximation error.



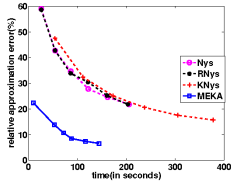
(i) covtype ($\gamma = 10$), memory vs approximation error.



(j) pendigit ($\gamma = 10$), time vs approximation error.



(k) ijcnn1 ($\gamma = 10$), time vs approximation error.



(l) covtype ($\gamma = 10$), time vs approximation error.

Figure 7: Low-rank Gaussian kernel approximation results. Methods with approximation error above the top of y-axis are not shown.

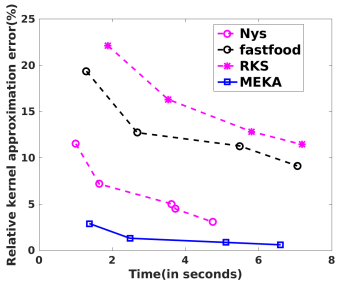
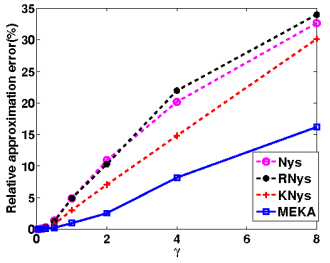
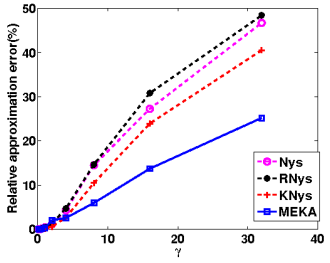


Figure 8: Comparison between Nyström based methods (MEKA and standard Nyström) and random feature based methods (RKS and fastfood). We can see that Nyström based methods perform much better than random feature based methods for kernel approximation.



(a) ijcnn1.



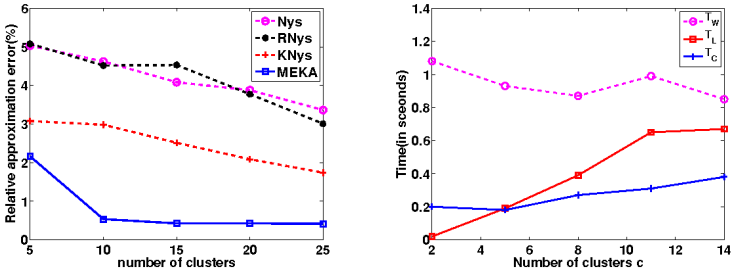
(b) cadata.

Figure 9: The kernel approximation errors for different Gaussian scaling parameter γ.

ijcnn1 dataset when varying the number of clusters c. Here the parameter γ is set to be 1. From Figure 10b, we observe that when the number of clusters c is small, T_W will dominate the whole process. As c increases, the time for computing the link matrix L, T_L, increases. This is because the number of off-diagonal blocks increases quadratically with c. Since the time complexity for k-means is $O(ncd)$, T_C increases linearly with c.

6.2 The Performance of MEKA for Approximating the Laplacian Kernel

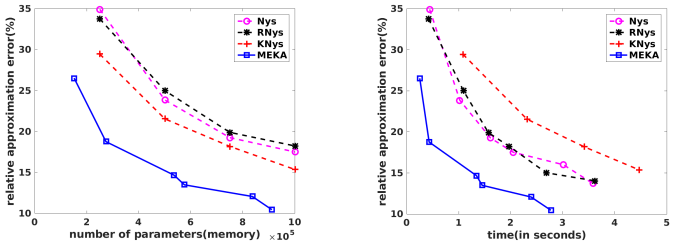
As we mentioned in Section 5, our algorithm is suitable for approximating stationary kernels. Besides Gaussian kernel, we can apply MEKA to approximate other kernels, e.g., Laplacian kernel ($K(x, y) = e^{-\gamma\|x-y\|_1}$). Figure 11 compares our proposed method with the standard Nyström(Nys), Randomized Nyström(RNys), and Kmeans Nyström(KNys) for approximating the Laplacian kernel on ijcnn1 data. Similar to Gaussian kernel, we observe that MEKA is more memory efficient and faster than other methods for approximating the Laplacian kernel.



(a) kernel approximation under different numbers of clusters c in MEKA. For standard Nyström(Nys), Randomized Nyström(RNys), and Kmeans Nyström(KNys), we use the same memory with MEKA.

(b) Time for each step of MEKA when varying the number of clusters c. T_C is the time for performing k-means clustering; T_W is the time to form the "basis" W from the diagonal blocks; and T_L is the time to compute L from off-diagonal blocks.

Figure 10: The kernel approximation errors and time cost for each step of MEKA when varying c on ijcnn1 dataset.



(a) memory vs approximation error.

(b) time vs approximation error.

Figure 11: Low-rank Laplacian kernel approximation results for ijcnn1 dataset.

6.3 Kernel Ridge Regression

Next we compare the performance of various methods on kernel ridge regression (Saunders, Gammerman, and Vovk, 1998):

$$\max_{\boldsymbol{\alpha}} \quad \lambda \boldsymbol{\alpha}^T \boldsymbol{\alpha} + \boldsymbol{\alpha}^T G \boldsymbol{\alpha} - 2\boldsymbol{\alpha}^T \boldsymbol{y}, \tag{19}$$

where G is the kernel matrix formed by training samples $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_l\}$, and $\boldsymbol{y} \in \mathbb{R}^l$ are the targets. For each kernel approximation method, we first form the approximated kernel $\tilde{G}$, and then solve (19) by conjugate gradient (CG). The main computation in CG is the matrix vector product $\tilde{G}\boldsymbol{v}$. Using low-rank approximation, this can be computed using $O(nk)$ flops. For our proposed method, we compute $WLW^T\boldsymbol{v}$, where $W^T\boldsymbol{v} = \sum_{s=1}^{c} W^{(s)}\boldsymbol{v}^{(s)}$ requires

Dataset	k	γ	c	λ	Nys	RNys	KNys	ENys	RKS	fastfood	MEKA
wine	128	2^{-10}	3	2^{-4}	0.7514	0.7555	0.7568	0.7732	0.7459	0.7509	**0.7375**
cadata	128	2^{2}	5	2^{-3}	0.1504	0.1505	0.1386	0.1462	0.1334	0.1502	**0.1209**
cpusmall	256	2^{2}	5	2^{-4}	8.8747	8.6973	6.9638	9.2831	9.6795	10.2601	**6.1130**
census	256	2^{-4}	5	2^{-5}	0.0679	0.0653	0.0578	0.0697	0.0727	0.0732	**0.0490**
covtype	256	2^{2}	10	2^{-2}	0.8197	0.8216	0.8172	0.8454	0.8011	0.8026	**0.7106**
mnist2m	256	2^{-5}	40	2^{-5}	0.2985	0.2962	0.2725	0.3018	0.3834	na	**0.2667**

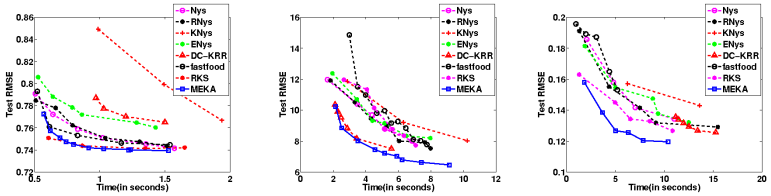
Table 8: Comparison of our proposed method with six other state-of-the-art kernel approximation methods on real datasets for kernel ridge regression, where λ is the regularization constant. All the parameters are chosen by cross validation, and every method has roughly the same amount of memory as in Table 7. All results show test RMSE for regression for each k. Note that k for fastfood needs to be larger than d, so we cannot test fastfood on mnist2m when $k = 256$.

$O(nk)$ flops, $L(W\boldsymbol{v})$ requires $O(\|L\|_0)$ flops, and $W(LW^T\boldsymbol{v})$ requires $O(nk)$ flops. Therefore, the time complexity for computing the matrix vector product for both MEKA and low-rank approximation methods is proportional to the memory for storing the approximate kernel matrices. Besides kernel approximation algorithms, we compare our method with another divide-and-conquer based kernel ridge regression method (denoted DC-KRR)(Zhang et al., 2013). The basic idea in Zhang et al. (2013) is to randomly partition the data into c parts and then train a kernel ridge regression model in each partition. To test a new data point, it will be tested on each submodel and the final prediction is the average of c predictions.

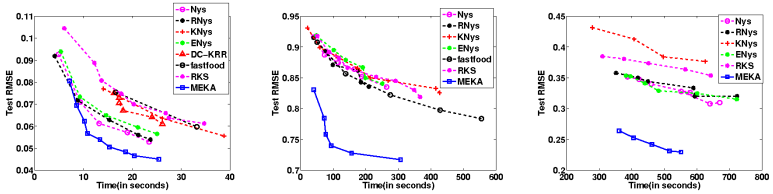
The parameters are chosen by five fold cross-validation and shown in Table 8. The rank for these algorithms is varied from 100 to 1000. The test root mean square error (test RMSE) is defined as $\|\boldsymbol{y}^{te} - G^{te}\boldsymbol{\alpha}\|$, where $\boldsymbol{y}^{te} \in \mathbb{R}^u$ is testing labels and $G^{te} \in \mathbb{R}^{u \times l}$ is the approximate kernel values between testing and training data. The covtype and mnist2m data sets are not originally designed for regression, and here we set the target variables to be 0 and 1 for mnist2m and -1 and 1 for covtype. Table 8 compares the kernel ridge regression performance of our proposed scheme with six other methods given the same amount of memory or same k in terms of test RMSE. It shows that our proposed method consistently performs better than other methods. Figure 12 shows the time usage of different methods for regression by varying the memory or rank k. As we can see that using the same amount of time, our proposed algorithm always achieves the lowest test RMSE. The total running time consists of the time for obtaining the low-rank approximation and time for regression. The former depends on the time complexity for each method, and the latter depends on the memory requirement to store the low-rank matrices. As shown in the previous experiment, MEKA is faster than the other methods while achieving lower approximation error and using less memory. As a consequence, it achieves lower test RMSE in less time compared to other kernel approximation methods.

7. Conclusions and Discussions

In this paper, we have proposed a novel framework, Memory Efficient Kernel Approximation (MEKA) for approximating shift-invariant kernel matrices. We observe that the structure of the shift-invariant kernel matrix changes from low rank to block diagonal as the scale



(a) wine, time vs regression error. (b) cpusmall, time vs regression error. (c) cadata, time vs regression error.



(d) census, time vs regression error. (e) covtype, time vs regression error. (f) mnist2m, time vs regression error.

Figure 12: Kernel ridge regression results for various data sets. Methods with regression error above the top of y-axis are not shown. All results are averaged over five independent runs.

parameter is changed. Our method exploits both low-rank and block structure present in the kernel matrix, and thus performs better than previously proposed low-rank based methods in terms of approximation and regression error, speed and memory usage. The code for MEKA is available at `www.cs.utexas.edu/~ssi/meka/`. We will discuss next about some typical problems encountered when using MEKA and how to deal with them.

7.1 Dealing with the Non-PSD Issue

If for each off-diagonal block, we sample all the entries, the resulting approximate matrix in MEKA will be positive semidefinite(PSD). The reason is as follow: if we use all the all-diagonal blocks to compute the link matrix L, the approximation will be $G \approx WLW^T$ with $L = W^T G W$. Since G is PSD, so it is with L, which proves WLW^T will be PSD.

However, due to the sampling procedure in MEKA, the resulting approximate matrix might not be PSD, which will cause some problems when PSD is required for some applications, e.g., kernel SVM with hinge loss. There are two simple and effective ways to solve this issue: (1) The first method (MEKA-PSD) is to set negative eigenvalues to 0. The procedure is first to perform eigen-decomposition on the small $ck \times ck$ "link" matrix $L = USU^T$(where U and S are the eigenvector and eigenvalue matrices for L respectively) in the MEKA representation, and then shrink its negative eigenvalues to be 0, which forms the new eigenvalue matrix $\bar{S}$ for L. After that, the new MEKA approximation will become: $\bar{G} = WU\bar{S}U^TW^T$. Since $\bar{S}$ is now PSD, so it is with $WU\bar{S}U^TW^T$. Therefore, $\bar{G}$ will be PSD after above shrinking operation. Due to eigen-decomposition on L and the following

shrinking operation, both the kernel approximation error and computation time will increase slightly. We show the comparison of Nys, MEKA and MEKA-PSD on ijcnn1 dataset in Figure 13. We can see that to achieve similar approximation, MEKA-PSD is slightly slower than MEKA, but still performs better than Nys, which also generates PSD kernel approximation. (2) The second method is to directly add a small value ϵ to the diagonal of $\bar{G}$, which is equivalent to using regularization term when applying MEKA for kernel methods. We show the kernel ridge regression results in Table 8, where we add regularization term to MEKA. Note that to make sure the resulting MEKA approximation is PSD for the second method, ϵ should be equal or larger than the absolute value of the smallest negative eigenvalue of $\bar{G}$.

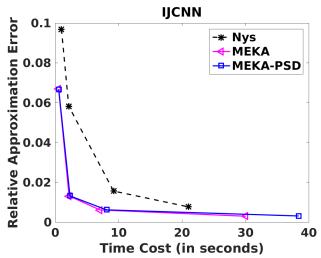


Figure 13: MEKA-PSD kernel approximation results for ijcnn1 dataset.

7.2 Dealing with the Instability Issue

There are two steps in Algorithm 1 that might cause the approximation unstable. One is due to the basis formed from each diagonal block– W_i. If the kernel matrix has strong block-diagonal structure, choosing columns from diagonal blocks to form W will perform well; on the contrary, if the kernel matrix does not have very strong block structure, other sampling strategies can be involved, for example, sampling columns from the rows corresponding to each partition to form the basis. Another step which might cause unstable result is due to insufficient entries sampled when forming the "link" matrix L. One way to solve this issue is to sample more entries from each off-diagonal block to form L when the computation time is not the main concern.

7.3 Other Applications using MEKA

In the experiment section, we apply MEKA for kernel approximation and kernel ridge regression, and besides that we could use our MEKA framework for many other machine learning applications: such as speeding up the computation of the inverse of the kernel matrix.

For many machine learning applications, e.g., Gaussian Process, we need to compute the matrix inverse $(G + \lambda I)^{-1}$, where computing the inverse of a dense matrix becomes the bottleneck for these applications. By approximating kernel matrix G in MEKA form

$G \approx WLW^T$, the inverse operation can be done in a faster fashion using Woodbury formula.

$$\begin{aligned}(\lambda I + G)^{-1} &\approx (\lambda I + WLW^T)^{-1} \\ &= \frac{1}{\lambda}(I - W(\lambda L^{-1} + W^T W)^{-1} W^T)\end{aligned}$$

We can see that after the approximation, we only need to inverse the $ck \times ck$ matrix which reduce the time complexity for inverse of the dense matrix from $O(n^3)$ to $O(n(ck)^2 + (ck)^3)$.

Besides speeding up the matrix inverse, MEKA can also be used to approximate the eigendecomposition of kernel matrix and used in various machine learning applications, for instance, manifold learning and kernelized dimensionality reduction.

Acknowledgments

This research was supported by NSF grants CCF-1320746, IIS-1546452 and CCF-1564000. Cho-Jui Hsieh also acknowledges support from XSED E startup resources.

References

D. Achlioptas, F. McSherry, and B. Schölkopf. Sampling techniques for kernel methods. In *Advances in Neural Information Processing Systems*, 2001.

F. R. Bach and M. I. Jordan. Predictive low-rank decomposition for kernel methods. In *International Conference on Machine Learning*, 2005.

K. Bache and M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

Y.-W Chang, C.-J. Hsieh, K.-W. Chang, M. Ringgaard, and C.-J Lin. Training and testing low-degree polynomial data mappings via linear SVM. *Journal of Machine Learning Research*, 11:1471–1490, 2010.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

A. Cotter, J. Keshet, and N. Srebro. Explicit approximations of the Gaussian kernel. *arXiv:1109.4760*3, 2011.

F. Cucker and S. Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39:1–49, 2001.

S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

A. Gittens and M. W. Mahoney. Revisiting the Nyström method for improved large-scale machine learning. In *International Conference on Machine Learning*, 2013.

N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, 2011.

R. Hamid, Y. Xiao, A. Gittens, and D. DeCoste. Compact random feature maps. In *International Conference on Machine Learning*, 2014.

C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear SVM. In *International Conference on Machine Learning*, 2008.

C.-J. Hsieh, S. Si, and I. S. Dhillon. A divide-and-conquer solver for kernel support vector machines. In *International Conference on Machine Learning*, 2014a.

C.-J. Hsieh, S. Si, and I. S. Dhillon. Fast prediction for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, 2014b.

P. Kar and H. Karnick. Random feature maps for dot product kernels. In *International Conference on Machine Learning*, 2012.

S. Kumar, M. Mohri, and A. Talwalkar. Ensemble Nyström methods. In *Advances in Neural Information Processing Systems*, 2009.

S. Kumar, M. Mohri, and A. Talwalkar. Sampling methods for the Nyström method. *Journal of Machine Learning Research*, 13:981–1006, 2012.

Q. V. Le, T. Sarlos, and A. J. Smola. Fastfood – approximating kernel expansions in loglinear time. In *International Conference on Machine Learning*, 2013.

M. Li, J. T. Kwok, and B.-L. Lu. Making large-scale Nyström approximation possible. In *International Conference on Machine Learning*, 2010.

M. W. Mahoney and P. Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Science*, 2009.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, 2007.

A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems*, 2008.

C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *International Conference on Machine Learning*, 1998.

B. Savas and I. S. Dhillon. Clustered low rank approximation of graphs in information science applications. In *SIAM Conference on Data Mining*, 2011.

B. Schölkopf and A. J. Smola. *Learning with kernels*. MIT Press, 2002.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

S. Si, C.-J. Hsieh, and I. S. Dhillon. Memory efficient kernel approximation. In *International Conference on Machine Learning*, 2014a.

S. Si, D. Shin, I. S. Dhillon, and Beresford N. Parlett. Multi-scale spectral decomposition of massive graphs. In *Advances in Neural Information Processing Systems*, 2014b.

S. Si, C.-J. Hsieh, and I. S. Dhillon. Computationally efficient Nyström approximation using fast transforms. In *International Conference on Machine Learning*, pages 2655–2663, 2016.

A. J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In *International Conference on Machine Learning*, 2000.

G.W. Stewart and Sun Ji-Guang. *Matrix Perturbation Theory*. Academic Press, Boston, 1990.

Rashish Tandon, Si Si, Pradeep Ravikumar, and Inderjit Dhillon. Kernel ridge regression via partitioning. *arXiv preprint arXiv:1608.01976*, 2016.

U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4), 2007.

S. Wang, C. Zhang, H. Qian, and Z. Zhang. Improving the modified Nyström method using spectral shifting. In *ACM SIGKDD Conferences on Knowledge Discovery and Data Mining*, 2014.

Z. Wang, N. Djuric, K. Crammer, and S. Vucetic. Trading representability for scalability: Adaptive multi-hyperplane machine for nonlinear classification. In *ACM SIGKDD Conferences on Knowledge Discovery and Data Mining*, 2011.

Christopher Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, 2001.

J. Yang, V. Sindhwani, H. Avron, and M. W. Mahoney. Quasi-Monte Carlo feature maps for shift-invariant kernels. In *International Conference on Machine Learning*, 2014.

T. Yang, Y.-F. Li, M. Mahdavi, R. Jin, and Z.-H. Zhou. Nyström method vs random Fourier features: A theoretical and empirical comparison. In *Advances in Neural Information Processing Systems*, 2012.

K. Zhang and J. T. Kwok. Clustered Nyström method for large scale manifold learning and dimension reduction. *IEEE Trans. Neural Networks*, 21(10):1576–1587, 2010.

K. Zhang, I. W. Tsang, and J. T. Kwok. Improved Nyström low rank approximation and error analysis. In *International Conference on Machine Learning*, 2008.

K. Zhang, L. Lan, Z. Wang, and F. Moerchen. Scaling up kernel SVM on limited resources: A low-rank linearization approach. In *International Conference on Artificial Intellignence and Statistics*, 2012.

Yuchen Zhang, John C. Duchi, and Martin J. Wainwright. Divide and conquer kernel ridge regression. In *Conference on Learning Theory*, 2013.

Journal of Machine Learning Research 18 (2017) 1-38 Submitted 5/15; Revised 10/16; Published 4/17

On the Equivalence between Kernel Quadrature Rules and Random Feature Expansions

Francis Bach FRANCIS.BACH@ENS.FR
INRIA - Sierra Project-team
Département d'Informatique de l'Ecole Normale Supérieure (UMR CNRS/ENS/INRIA)
2, rue Simone Iff
75012 Paris, France

Editor: Arthur Gretton

Abstract

We show that kernel-based quadrature rules for computing integrals can be seen as a special case of random feature expansions for positive definite kernels, for a particular decomposition that always exists for such kernels. We provide a theoretical analysis of the number of required samples for a given approximation error, leading to both upper and lower bounds that are based solely on the eigenvalues of the associated integral operator and match up to logarithmic terms. In particular, we show that the upper bound may be obtained from independent and identically distributed samples from a specific non-uniform distribution, while the lower bound if valid for any set of points. Applying our results to kernel-based quadrature, while our results are fairly general, we recover known upper and lower bounds for the special cases of Sobolev spaces. Moreover, our results extend to the more general problem of full function approximations (beyond simply computing an integral), with results in L_2- and L_∞-norm that match known results for special cases. Applying our results to random features, we show an improvement of the number of random features needed to preserve the generalization guarantees for learning with Lipshitz-continuous losses.

Keywords: Quadrature, positive-definite kernels, integral operators.

1. Introduction

The numerical computation of high-dimensional integrals is one of the core computational tasks in many areas of machine learning, signal processing and more generally applied mathematics, in particular in the context of Bayesian inference (Gelman, 2004), or the study of complex systems (Robert and Casella, 2005). In this paper, we focus on *quadrature rules*, that aim at approximating the integral of a certain function from only the (potentially noisy) knowledge of the function values at as few as possible well-chosen points. Key situations that remain active areas of research are problems where the measurable space where the function is defined on is either high-dimensional or structured (e.g., presence of discrete structures, or graphs). For these problems, techniques based on *positive definite kernels* have emerged as having the potential to efficiently deal with these situations, and to improve over plain Monte-Carlo integration (O'Hagan, 1991; Rasmussen and Ghahramani, 2003; Huszár and Duvenaud, 2012; Oates and Girolami, 2015). In particular, the quadrature problem may be cast as the one of approximating a fixed element, the mean element (Smola et al., 2007), of a Hilbert space as a linear combination of well chosen elements, the goal being to minimize the number of these factors as it corresponds to the required number of function evaluations.

A seemingly unrelated problem on positive definite kernels have recently emerged, namely the representation of the corresponding infinite-dimensional feature space from *random sets of features*. If a certain positive definite kernel between two points may be represented as the expectation of the product of two random one-dimensional (typically non-linear) features computed on these two points, the full kernel (and hence its feature space) may be approximated by sufficiently many random samples, replacing the expectation by a sample average (Neal, 1995; Rahimi and Recht, 2007; Huang et al., 2006). When using these random features, the complexity of a regular kernel method such as the support vector machine or ridge regression goes from quadratic in the number of observations to linear in the number of observations, with a constant proportional to the number of random features, which thus drives the running time complexity of these methods.

In this paper, we make the following contributions:

- After describing the functional analysis framework our analysis is based on and presenting many examples in Section 2, we show in Section 3 that these two problems are strongly related; more precisely, optimizing weights in kernel-based quadrature rules can be seen as decomposing a certain function in a special class of random features for a particular decomposition that always exists for all positive definite kernels on a measurable space.
- We provide in Section 4 a theoretical analysis of the number of required samples for a given approximation error, leading to both upper and lower bounds that are based solely on the eigenvalues of the associated integral operator and match up to logarithmic terms. In particular, we show that the upper bound may be obtained as independent and identically distributed samples from a specific non-uniform distribution, while the lower bound if valid for any set of points.
- Applying our results to kernel quadrature, while our results are fairly general, we recover known upper and lower bounds for the special cases of Sobolev spaces (Section 4.4). Moreover, our results extend to the more general problem of full function approximations (beyond simply computing an integral), with results in L_2- and L_∞-norm that match known results for special cases (Section 5).
- Applying our results to random feature expansions, we show in Section 4.5 an improvement of the number of random features needed for preserving the generalization guarantees for learning with Lipshitz-continuous losses.

2. Random Feature Expansions of Positive Definite Kernels

Throughout this paper, we consider a topological space $\mathcal{X}$ equipped with a Borel probability measure $d\rho$, which we assume to have full support. This naturally defines the space of square-integrable functions[1].

1. For simplicity and following most of the literature on kernel methods, we identify functions and their equivalence classes for the equivalence relationship of being equal except for a zero-measure (for $d\rho$) subset of $\mathcal{X}$.

©2017 Francis Bach.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-178.html.

2.1 Reproducing Kernel Hilbert Spaces and Integral Operators

We consider a continuous positive definite kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, that is a symmetric function such that for all finite families of points in $\mathcal{X}$, the matrix of pairwise kernel evaluations is positive semi-definite. This thus defines a reproducing kernel Hilbert space (RKHS) $\mathcal{F}$ of functions from $\mathcal{X}$ to $\mathbb{R}$, which we also assume separable. This RKHS has two important characteristic properties (see, e.g., Berlinet and Thomas-Agnan, 2004):

(a) *Membership of kernel evaluations*: for any $x \in \mathcal{X}$, the function $k(\cdot, x) : y \mapsto k(y, x)$ is an element of $\mathcal{F}$.

(b) *Reproducing property*: for all $f \in \mathcal{F}$ and $x \in \mathcal{X}$, $f(x) = \langle f, k(\cdot, x)\rangle_{\mathcal{F}}$. In other words, function evaluations are equal to dot-products with a specific element of $\mathcal{F}$.

Moreover, throughout the paper, we assume that the function $x \mapsto k(x, x)$ is integrable with respect to $d\rho$ (which is weaker than $\sup_{x\in\mathcal{X}} k(x, x) < \infty$). This implies that $\mathcal{F}$ is a subset of $L_2(d\rho)$; that is, functions in the RKHS $\mathcal{F}$ are all square-integrable for $d\rho$. In general, $\mathcal{F}$ is strictly included in $L_2(d\rho)$, but, in this paper, we will always assume that it is *dense* in $L_2(d\rho)$, that is, any function in $L_2(d\rho)$ may be approximated arbitrarily closely by a function in $\mathcal{F}$. Finally, for simplicity of our notation (to make sure that the sequence of eigenvalues of integral operators is infinite) we will always assume that $L_2(d\rho)$ is infinite-dimensional, which excludes finite sets for $\mathcal{X}$. Note that the last two assumptions (denseness and infinite dimensionality) can easily be relaxed.

Integral operator. Reproducing kernel Hilbert spaces are often studied through a specific integral operator which leads to an isometry with $L_2(d\rho)$ (Smale and Cucker, 2001). Let $\Sigma : L_2(d\rho) \to L_2(d\rho)$ be defined as

$$(\Sigma f)(x) = \int_{\mathcal{X}} f(y)k(x, y)d\rho(y).$$

Since $\int_{\mathcal{X}} k(x, x)d\rho(x)$ is finite, Σ is self-adjoint, positive semi-definite and trace-class (Simon, 1979). Given that Σf is a linear combination of kernel functions $k(\cdot, y)$, it belongs to $\mathcal{F}$. More precisely, since we have assumed that $\mathcal{F}$ is dense in $L_2(d\rho)$, $\Sigma^{1/2}$, which is the unique positive self-adjoint square root of Σ, is a bijection from $L_2(d\rho)$ to our RKHS $\mathcal{F}$; that is, for any $f \in \mathcal{F}$, there exists a unique $g \in L_2(d\rho)$ such that $f = \Sigma^{1/2}g$ and $\|f\|_{\mathcal{H}} = \|g\|_{L_2(d\rho)}$ (Smale and Cucker, 2001). This justifies the notation $\Sigma^{-1/2}f$ for $f \in \mathcal{F}$ and means that $\Sigma^{1/2}$ is an isometry from $L_2(d\rho)$ to $\mathcal{F}$; in other words, for any functions f and g in $\mathcal{F}$, we have:

$$\langle f, g\rangle_{\mathcal{F}} = \langle \Sigma^{-1/2}f, \Sigma^{-1/2}g\rangle_{L_2(d\rho)}.$$

This justifies the view of $\mathcal{F}$ as the subspace of functions $f \in L_2(d\rho)$ such that $\|\Sigma^{-1/2}f\|^2_{L_2(d\rho)}$. This relationship is even more transparent when considering a spectral decomposition of Σ.

Mercer decomposition. From extensions of Mercer's theorem (König, 1986), there exists an orthonormal *basis* $(e_m)_{m\geqslant 1}$ of $L_2(d\rho)$ and a summable non-increasing sequence of strictly positive eigenvalues $(\mu_m)_{m\geqslant 1}$ such that $\Sigma e_m = \mu_m e_m$. Note that since we have assumed that $\mathcal{F}$ is dense in $L_2(d\rho)$, there are no zero eigenvalues.

Since $\Sigma^{1/2}$ is an isometry from $L_2(d\rho)$ to $\mathcal{F}$, $(\mu_m^{1/2}e_m)_{m\geqslant 1}$ is an orthonormal basis of $\mathcal{F}$. Moreover, we can use the eigendecomposition to characterize elements of $\mathcal{F}$ as the functions in $L_2(d\rho)$ such that

$$\|\Sigma^{-1/2}f\|^2_{L_2(d\rho)} = \sum_{m\geqslant 1} \mu_m^{-1}\langle f, e_m\rangle^2_{L_2(d\rho)}$$

is finite. In other words, once projected in the orthonormal basis $(e_m)_{m\geqslant 1}$, elements f of $\mathcal{F}$ correspond to a certain decay of its decomposition coefficients $(\langle f, e_m\rangle_{L_2(d\rho)})_{m\geqslant 1}$.

Finally, by decomposing the function $k(\cdot, y) : x \mapsto k(x, y)$, we obtain the Mercer decomposition:

$$k(x, y) = \sum_{m\geqslant 1} \mu_m e_m(x)e_m(y).$$

Properties of the spectrum. The sequence of eigenvalues $(\mu_m)_{m\geqslant 1}$ is an important quantity that appears in most of the analyses of kernel methods (Hastie and Tibshirani, 1990; Caponnetto and De Vito, 2007; Harchaoui et al., 2008; Bach, 2013). It depends both on the kernel k and the chosen distribution $d\rho$.

Some modifications of the kernel k or the distribution $d\rho$ lead to simple behaviors for the spectrum. For example, if we have a second distribution so that $\frac{d\rho'}{d\rho}$ is upper-bounded by a constant c, then, as a consequence of the Courant-Fischer minimax theorem (Horn and Johnson, 2012), the eigenvalues for $d\rho'$ are less than than c times that the ones for $d\rho$. Similarly, if the kernel k' is such that $ck - k'$ is a positive definite kernel, then we have a similar bound between eigenvalues.

In this paper, for any strictly positive λ, we will also consider the quantity $m^*(\lambda)$ equal to the number of eigenvalues μ_m that are greater than or equal to λ. Since we have assume that the sequence m is non-increasing, we have $m^*(\lambda) = \max\{m \geqslant 1,\ \mu_m \geqslant \lambda\}$. This is a left-continuous non-increasing function, that tends to $+\infty$ when λ tends to zero (since we have assumed that there are infinitely many strictly positive eigenvalues), and characterizes the sequence $(\mu_m)_{m\geqslant 1}$, as we can recover μ_m as $\mu_m = \sup\{\lambda \geqslant 0,\ m^*(\lambda) \geqslant m\}$.

Potential confusion with covariance operator. Note that the operator Σ is a self-adjoint operator on $L_2(d\rho)$. It should not be confused with the (non-centered) covariance operator C (Baker, 1973), which is a self-adjoint operator on a different space, namely the RKHS $\mathcal{F}$, defined by $\langle g, Cf\rangle_{\mathcal{F}} = \int_{\mathcal{X}} f(x)g(x)d\rho(x)$. Given that $\Sigma^{1/2}$ is an isometry from $L_2(d\rho)$ to $\mathcal{F}$, the operator C may also be used to define an operator on $L_2(d\rho)$, which happens to be exactly Σ. Thus, the two operators have the same eigenvalues. Moreover, we have, for any $y \in \mathcal{X}$:

$$(Cf)(y) = \langle k(\cdot, y), Cf\rangle_{\mathcal{F}} = \int_{\mathcal{X}} k(x, y)f(x)d\rho(x) = (\Sigma f)(y),$$

that is, C is equal to the restriction of Σ on $\mathcal{F}$.

2.2 Kernels as Expectations

On top of the generic assumptions made above, we assume that there is another measurable set $\mathcal{V}$ equipped with a probability measure $d\tau$. We consider a function $\varphi : \mathcal{V} \times \mathcal{X} \to \mathbb{R}$ which

is square-integrable (for the measure $d\tau \otimes d\rho$), and assume that the kernel k may be written as, for all $x, y \in \mathcal{X}$:

$$k(x,y) = \int_{\mathcal{V}} \varphi(v,x)\varphi(v,y)d\tau(v) = \langle \varphi(\cdot,x), \varphi(\cdot,y)\rangle_{L_2(d\tau)}. \quad (1)$$

In other words, the kernel between x and y is simply the expectation of $\varphi(v,x)\varphi(v,y)$ for v following the probability distribution $d\tau$. In this paper, we see $x \mapsto \varphi(v,x) \in \mathbb{R}$ as a one-dimensional random feature and $\varphi(v,x)\varphi(v,y)$ is the dot-product associated with this random feature. We could consider extensions where $\varphi(v,x)$ has values in a Hilbert space (and not simply $\mathbb{R}$), but this is outside the scope of this paper.

Square-root of integral operator. Such additional structure allows to give an explicit characterization of the RKHS $\mathcal{F}$ in terms of the features φ. In terms of operators, the function φ leads to a specific square-root of the integral operator Σ defined in Section 2.1 (which is not the positive self-adjoint square-root $\Sigma^{1/2}$).

We consider the bounded linear operator $T : L_2(d\tau) \to L_2(d\rho)$ defined as

$$(Tg)(x) = \int_{\mathcal{V}} g(v)\varphi(v,x)d\tau(v) = \langle g, \varphi(\cdot,x)\rangle_{L_2(d\tau)}. \quad (2)$$

Given $T : L_2(d\tau) \to L_2(d\rho)$, the adjoint operator $T^* : L_2(d\rho) \to L_2(d\tau)$ is the unique operator such that $\langle g, T^*f\rangle_{L_2(d\tau)} = \langle Tg, f\rangle_{L_2(d\rho)}$ for all f, g. Given the definition of T in Eq. (2), we simply inverse the role of $\mathcal{V}$ and $\mathcal{X}$ and have:

$$(T^*f)(v) = \int_{\mathcal{X}} f(x)\varphi(v,x)d\rho(x).$$

This implies by Fubini's theorem that

$$\begin{aligned}(TT^*f)(y) &= \int_{\mathcal{V}} \Big(\int_{\mathcal{X}} f(x)\varphi(v,y)d\rho(x)\Big)\varphi(v,x)d\tau(v) \\ &= \int_{\mathcal{X}} f(x)\Big(\int_{\mathcal{V}} \varphi(v,y)\varphi(v,x)d\tau(v)\Big)d\rho(x) = \int_{\mathcal{X}} f(x)k(x,y)d\rho(x) = (\Sigma f)(y),\end{aligned}$$

that is we have an expression of the integral operator Σ as $\Sigma = TT^*$. Thus, the decomposition of the kernel k as an expectation corresponds to a particular *square root* T of the integral operator—there are many possible choices for such square roots, and thus many possible expansions like Eq. (1). It turns out that the positive self-adjoint square root $\Sigma^{1/2}$ will correspond to the equivalence with quadrature rules (see Section 3.2).

Decomposition of functions in $\mathcal{F}$. Since $\Sigma = TT^*$ and $\Sigma^{1/2}$ is an isometry between $L_2(d\rho)$ and $\mathcal{F}$, we can naturally expressed any elements of $\mathcal{F}$ through the operator T and thus the features φ.

As a linear operator, T defines a bijection from the orthogonal of its null space $(\text{Ker } T)^\perp \subset L_2(d\tau)$ to its image $\text{Im}(T) \subset L_2(d\rho)$, and this allows to define uniquely $T^{-1}f \in (\text{Ker } T)^\perp$ for any $f \in \text{Im}(T)$, and a dot-product on $\text{Im}(T)$ as

$$\langle f, h\rangle_{\text{Im}(T)} = \langle T^{-1}f, T^{-1}g\rangle_{L_2(d\tau)}.$$

As shown by Bach (2017, App. A), $\text{Im}(T)$ turns out to be equal to our RKHS[2]. Thus, the norm $\|f\|_{\mathcal{F}}^2$ for $f \in \mathcal{F}$ is equal to the squared L_2-norm of $T^{-1}f \in (\text{Ker } T)^\perp$, which is itself equal to the minimum of $\|g\|_{L_2(d\tau)}^2$ over all g such that $Tg = f$. The resulting g may also be defined through pseudo-inverses.

In other words, a function $f \in L_2(d\rho)$ is in $\mathcal{F}$ if and only if it may be written as

$$\forall x \in \mathcal{X},\ f(x) = \int_{\mathcal{V}} g(v)\varphi(v,x)d\tau(v) = \langle g, \varphi(\cdot,x)\rangle_{L_2(d\tau)},$$

for a certain function $g : \mathcal{V} \to \mathbb{R}$ such that $\|g\|_{L_2(d\tau)}^2$ is finite, with a norm $\|f\|_{\mathcal{F}}^2$ equal to the minimum (which is always attained) of $\|g\|_{L_2(d\tau)}^2$, over all possible decompositions of f.

Singular value decomposition. The operator T is an Hilbert-Schmidt operator, to which the singular value decopomposition can be applied (Kato, 1995). That is, there exists an orthonormal basis $(f_m)_{m\geqslant 1}$ of $(\text{Ker } T)^\perp \subset L_2(d\tau)$, together with the orthonormal basis $(e_m)_{m\geqslant 1}$ of $L_2(d\rho)$ which we have from the eigenvalue decomposition of $\Sigma = TT^*$, such that $Tf_m = \mu_m^{1/2}e_m$. Moreover, we have:

$$\varphi(v,x) = \sum_{m\geqslant 1} \mu_m^{1/2} e_m(x)f_m(v), \quad (3)$$

with a convergence in $L_2(d\tau \otimes d\rho)$. This extends the Mercer decomposition of the kernel $k(x,y)$.

Integral operator as an expectation. Given the expansion of the kernel k in Eq. (1), we may express the integral operator Σ as follows, explicitly as an expectation:

$$\begin{aligned}\Sigma f &= \int_{\mathcal{X}} f(y)k(\cdot,y)d\rho(y) = \int_{\mathcal{X}}\int_{\mathcal{V}} f(y)\varphi(v,\cdot)\varphi(v,y)d\rho(y)d\tau(v) \\ &= \int_{\mathcal{V}} \varphi(v,\cdot)\langle \varphi(v,\cdot), f\rangle_{L_2(d\rho)}d\tau(v) = \Big(\int_{\mathcal{V}} \varphi(v,\cdot) \otimes_{L_2(d\rho)} \varphi(v,\cdot)d\tau(v)\Big)f, \quad (4)\end{aligned}$$

where $a \otimes_{L_2(d\rho)} b$ is the operator $L_2(d\rho) \to L_2(d\rho)$ so that $(a \otimes_{L_2(d\rho)} b)f = \langle b, f\rangle_{L_2(d\rho)}a$. This will be useful to define empirical versions, where the integral over $d\tau$ will be replaced by a finite average.

2.3 Examples

In this section, we provide examples of kernels and usual decompositions. We first start by decompositions that always exist, then focus on specific kernels based on Fourier components.

2. The proof goes as follows: (a) for any $y \in \mathcal{X}$, $k(\cdot,y)$ can be expressed as $\int_{\mathcal{V}} \varphi(v,y)\varphi(v,\cdot)d\tau(v) = T\varphi(\cdot,y)$ and thus belongs to $\text{Im}(T)$; (b) for any $f \in \text{Im}(T)$, and $y \in \mathcal{X}$, we have $\langle f, k(\cdot,y)\rangle_{\text{Im}(T)} = \langle T^{-1}f, \varphi(\cdot,y)\rangle_{L_2(d\tau)} = (TT^{-1}f)(y) = f(y)$, that is, the reproducing property is satisfied. These two properties are characteristic of $\mathcal{F}$.

Mercer decompositions. The Mercer decomposition provides an expansion for all kernels, as follows:

$$k(x,y) = \sum_{m \geqslant 1} \frac{\mu_m}{\operatorname{tr} \Sigma} \Big[\big(\operatorname{tr} \Sigma \big)^{1/2} e_m(x) \Big] \cdot \Big[\big(\operatorname{tr} \Sigma \big)^{1/2} e_m(x) \Big],$$

which can be transformed in to an expectation with $\mathcal{V} = \mathbb{N}^*$. In Section 3.2, we provide another generic decomposition with $\mathcal{V} = \mathcal{X}$. Note that this decomposition is typically impossible to compute (except for special cases below, i.e., special pairs of kernels k and distributions $d\rho$).

Periodic kernels on $[0,1]$. We consider $\mathcal{X} = [0,1]$ and translation-invariant kernels $k(x,y)$ of the form $k(x,y) = t(x-y)$, where t is a square-integrable 1-periodic function. These kernels are positive definite if and only if the Fourier series of t is non-negative (Wahba, 1990). An orthonormal basis of $L_2([0,1])$ is composed of the constant function $c_0 : x \mapsto 1$ and the functions $c_m : x \mapsto \sqrt{2} \cos 2\pi mx$ and $s_m : x \mapsto \sqrt{2} \sin 2\pi mx$. A kernel may thus be expressed as

$$k(x,y) = \nu_0 c_0(x) + \sum_{m>0} \nu_m \big[c_m(x) c_m(y) + s_m(x) s_m(y) \big] = \nu_0 + 2 \sum_{m>0} \nu_m \cos 2\pi m(x-y).$$

This can be put trivially as an expectation with $\mathcal{V} = \mathbb{Z}$ and leads to the usual Fourier features (Rahimi and Recht, 2007). This is also exactly a Mercer decomposition for k and the uniform distribution on $[0,1]$, with eigenvalues ν_0 and ν_m, $m > 0$ (each of these with multiplicity 2). The associated RKHS norm for a function f is then equal to

$$\|f\|_{\mathcal{F}}^2 = \nu_0^{-1} \Big(\int_0^1 f(x) dx \Big)^2 + 2 \sum_{m>0} \nu_m^{-1} \left[\Big(\int_0^1 f(x) \cos 2\pi mx dx \Big)^2 + \Big(\int_0^1 f(x) \sin 2\pi mx dx \Big)^2 \right].$$

A particularly interesting example is obtained through derivatives of f. If f is differentiable and has a derivative $f' \in L_2([0,1])$, then, on the Fourier series coefficients of f, taking the derivative corresponds to multiplying the two m-th coefficients by $2\pi m$ and swapping them. Sobolev spaces for periodic functions on $[0,1]$ (i.e., such that $f(0) = f(1)$) are defined through integrability of derivatives (Adams and Fournier, 2003). In the Hilbert space set-up, a function f belongs to the Sobolev space of order s if one can define a s-th order square-integrable derivative in L_2 (for the Lebesgue measure, which happens to be equal to $d\rho$), that is, $f^{(s)} \in L_2([0,1])$. The Sobolev squared norm is then defined as any positive linear combination of the quadratic forms $\int_0^1 f^{(t)}(x)^2 dx$, $t \in \{0, \dots, s\}$, with non-zero coefficients for $t = 0$ and $t = s$ (all of these norms are then equivalent). If only using $t = 0$ and $t = s$ with non-zero coefficients, we need $\nu_0^{-1} = 1$ and $\nu_m^{-1} = 1 + m^{2s}$. An equivalent (i.e, with upper and lower bounded ratios) sequence is obtained by replacing $\nu_m = (1 + m^{2s})^{-1}$ by $\nu_m = m^{-2s}$, leading to a closed-form formula:

$$k(x,y) = 1 + \frac{(-1)^{s-1} (2\pi)^{2s}}{(2s)!} B_{2s}(\{x-y\}),$$

where $\{x-y\}$ denotes the fractional part of $x-y$, and B_{2s} is the $2s$-th Bernoulli polynomial (Wahba, 1990). The RKHS $\mathcal{F}$ is then the Sobolev space of order s on $[0,1]$, with a norm equivalent to any of the family of Sobolev norms; it will be used as a running example throughout this paper.

Extensions to $[0,1]^d$. In order to extend to $d > 1$, we may consider several extensions as described by Oates and Girolami (2015), and compute the resulting eigenvalues of the integral operators. For simplicity, we consider the Sobolev space on $[0,1]$, with $\nu_0 = 1$ and $\nu_m^{-1} = m^{2s}$ for $m > 0$. The first possibility to extend to $[0,1]^d$ is to take a kernel which is simply the pointwise product of individual kernels on $[0,1]$. That is, if $k(x,y)$ is the kernel on $[0,1]$, define $K(X,Y) = \prod_{j=1}^d k(x_j, y_j)$ between X and Y in $[0,1]^d$. As shown in Appendix A, this leads to eigenvalue decays bounded by $(\log m)^{2s(d-1)} m^{-2s}$, and thus up to logarithmic terms at the same speed m^{-2s} as $d = 1$. While this sounds attractive in terms of generalization performance, it corresponds to a space a function which is not a Sobolev space in d dimensions. That is the associated squared norm on f would be equivalent to a linear combination of squared L_2-norm of partial derivatives

$$\int_{[0,1]^d} \Big(\frac{\partial^{t_1 + \cdots + t_d} f}{\partial x_1^{t_1} \cdots \partial x_d^{t_d}} \Big)^2 dx$$

for all $t_1, \dots, t_d$ in $\{0, \dots, s\}$. This corresponds to functions which have square-integrable partial derivatives with all *individual* orders less than s. All values of $s \geqslant 1$ are allowed and lead to an RKHS.

This is thus to be contrasted with the usual multi-dimensional Sobolev space which is composed of functions which have square-integrable partial derivatives with all orders $(t_1, \dots, t_d)$ with *sum* $t_1 + \cdots + t_d$ less than s. Only $s > d/2$ is then allowed to get an RKHS (see, e.g., Berlinet and Thomas-Agnan, 2004, Theorem 132, p. 348). The Sobolev norm is then of the form

$$\sum_{t_1 + \cdots + t_d \leqslant s} \int_{[0,1]^d} \Big(\frac{\partial^{t_1 + \cdots + t_d} f}{\partial x_1^{t_1} \cdots \partial x_d^{t_d}} \Big)^2 dx.$$

In the expansion on the d-th order tensor product of the Fourier basis, the norm above is equivalent to putting a weight on the element $(m_1, \dots, m_d)$ asymptotically equivalent to $\big(\sum_{j=1}^d m_j \big)^{2s}$, which thus represent the inverse of the eigenvalues of the corresponding kernel for the uniform distribution $d\rho$ (this is simply an explicit Mercer decomposition). Thus, the number of eigenvalues which are greater than λ grows as the number of $(m_1, \dots, m_d)$ such that their sum is less than $\lambda^{-1/(2s)}$, which itself is less than a constant times $\lambda^{-d/(2s)}$ (see a proof in Appendix A). This leads to an eigenvalue decay of $m^{-2s/d}$, which is much worse because of the term in $1/d$ in the exponent.

Translation invariant kernels on $\mathbb{R}^d$. We consider $\mathcal{X} = \mathbb{R}^d$ and translation-invariant kernels $k(x,y)$ of the form $k(x,y) = t(x-y)$, where t is an integrable function from $\mathbb{R}^d$ to $\mathbb{R}$. It is known that these kernels are positive definite if and only if the Fourier transform of t is always a non-negative real number. More precisely, if $\hat{t}(\omega) = \int_{\mathbb{R}^d} t(x) e^{-i\omega^\top x} dx \in \mathbb{R}_+$, then

$$k(x,y) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \hat{t}(\omega) e^{i\omega^\top (x-y)} d\omega = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \hat{t}(\omega) \big[\cos \omega^\top x \cos \omega^\top y + \sin \omega^\top x \sin \omega^\top y \big] d\omega.$$

Following Rahimi and Recht (2007), by sampling ω from a density proportional to $\hat{t}(\omega) \in \mathbb{R}_+$ and b uniformly in $[0,1]$ (and independently of ω), then by defining $\mathcal{V} = \mathbb{R}^d \times [0,1]$ and $\varphi(\omega, b, x) = \sqrt{2} \cos(\omega^\top x + 2\pi b)$, we obtain the kernel k.

For these kernels, the decay of eigenvalues has been well-studied by Widom (1963), who relates the decay of eigenvalues to the tails of the distribution $d\rho$ and the decay of the Fourier transform of t. For example, for the Gaussian kernel where $k(x,y) = \exp(-\alpha\|x-y\|_2^2)$, on sub-Gaussian distributions, the decay of eigenvalues is geometric, and for kernels leading to Sobolev spaces of order s, such as the Matern kernel (Furrer and Nychka, 2007), the decay is of the form $m^{-2s/d}$. See also examples by Birman and Solomyak (1977); Harchaoui et al. (2008).

Finally, note that in terms of computation, there are extensions to avoid linear complexity in d (Le et al., 2013).

Kernels on hyperspheres. If $\mathcal{X} \subset \mathbb{R}^{d+1}$ is the d-dimensional hypersphere defined as $\{x \in \mathbb{R}^{d+1}, \|x\|_2^2 = 1\}$, then specific kernels may be used, of the form $k(x,y) = t(x^\top y)$, where t has to have a positive Legendre expansion (Smola et al., 2001). Alternatively, kernels based on neural networks with random weights are directly in the form of random features (Cho and Saul, 2009; Bach, 2017): for example, the kernel $k(x,y) = \mathbb{E}(v^\top x)_+^s (v^\top x)_+^s$ for v uniformly distributed in the hypersphere corresponds to sampling weights in a one-hidden layer neural network with rectified linear units (Cho and Saul, 2009). It turns out that these kernels have a known decay for their spectrum.

As shown by Smola et al. (2001); Bach (2017), the equivalent of Fourier series (which corresponds to $d = 1$) is then the basis of spherical harmonics, which is organized by integer frequencies $k \geqslant 1$; instead of having 2 basis vectors (sine and cosine) per frequency, there are $O(k^{d-1})$ of them. As shown by Bach (2017, page 44), we have an explicit expansion of $k(x,y)$ in terms of spherical harmonics, leading to a sequence of eigenvalues equal to $k^{-d-2s-1}$ on the entire subspace associated with frequency k. Thus, by taking multiplicity into account, after $\sum_{j=1}^k j^{d-1} \approx k^d$ (up to constants) eigenvalues, we have an eigenvalue of $k^{-d-2s-1}$; this leads to an eigenvalue decay (where all eigenvalues are ordered in decreasing order and we consider the m-th one) as $(m^{1/d})^{-d-2s-1} = m^{-1-1/d-2s/d}$.

2.4 Approximation from Randomly Sampled Features

Given the formulation of k as an expectation in Eq. (1), it is natural to consider sampling n elements $v_1, \dots, v_n \in \mathcal{V}$ from the distribution $d\tau$ and define the kernel approximation

$$\hat{k}(x,y) = \frac{1}{n}\sum_{i=1}^n \varphi(v_i,x)\varphi(v_i,y), \tag{5}$$

which defines a finite-dimensional RKHS $\hat{\mathcal{F}}$.

From the strong law of large numbers—which can be applied because we have the finite expectation $\mathbb{E}|\varphi(v,x)\varphi(v,y)| \leqslant \big(\mathbb{E}|\varphi(v,x)|^2\mathbb{E}|\varphi(v,y)|^2\big)^{1/2}$, when n tends to infinity, $\hat{k}(x,y)$ tends to $k(x,y)$ almost surely, and thus we get as tight as desired approximations of the kernel k, for a given pair $(x,y) \in \mathcal{X}\times\mathcal{X}$. Rahimi and Recht (2007) show that for translation-invariant kernels on a Euclidean space, then the convergence is uniform over a compact subset of $\mathcal{X}$, with the traditional rate of convergence of $\sqrt{\frac{\log n}{n}}$.

In this paper, we rather consider *approximations of functions* in $\mathcal{F}$ by functions in $\hat{\mathcal{F}}$, the RKHS associated with $\hat{k}$. A key difficulty is that in general $\hat{\mathcal{F}}$ is not even included in $\mathcal{F}$,

and therefore, we cannot use the norm in $\mathcal{F}$ to characterize approximations. In this paper, we choose the L_2-norm associated with the probability measure $d\rho$ on $\mathcal{X}$ to characterize the approximation. Given $f \in \mathcal{F}$ with norm $\|f\|_{\mathcal{F}}$ less than one, we look for a function $\hat{f} \in \hat{\mathcal{F}}$ of the smallest possible norm and so that $\|f - \hat{f}\|_{L_2(d\rho)}$ is as small as possible.

Note that the measure $d\tau$ is associated to the kernel k and the random features φ, while the measure $d\rho$ is associated to the way we want to measure errors (and leads to a specific defintion of the integral operator Σ).

Computation of error. Given the definition of the Hilbert space $\mathcal{F}$ in terms of φ in Section 2.2, given $g \in L_2(d\tau)$ with $\|g\|_{L_2(d\tau)} \leqslant 1$ and $f(x) = \int_{\mathcal{V}} g(v)\varphi(v,x)d\tau(v)$, we aim at finding an element of $\hat{\mathcal{F}}$ close to f. We can also represent $\hat{\mathcal{F}}$ through a similar decomposition, now with a finite number of features, i.e., through $\alpha \in \mathbb{R}^n$ such that $\hat{f} = \sum_{i=1}^n \alpha_i \varphi(v_i,\cdot)$ with norm[3] $\|\hat{f}\|_{\hat{\mathcal{F}}}^2 \leqslant n\|\alpha\|_2^2$ as small as possible and so that the following approximation error is also small:

$$\|\hat{f} - f\|_{L_2(d\rho)} = \Big\| \sum_{i=1}^n \alpha_i \varphi(v_i,\cdot) - \int_{\mathcal{V}} g(v)\varphi(v,\cdot)d\tau(v)\Big\|_{L_2(d\rho)}. \tag{6}$$

Note that with $\alpha_i = \frac{1}{n}g(v_i)$ and v_i sampled from $d\tau$ (independently), then, we have $\mathbb{E}(\|\alpha\|_2^2) = \sum_{i=1}^n \mathbb{E}\alpha_i^2 = \frac{1}{n}\mathbb{E}g(v)^2 \leqslant \frac{1}{n}$ and an expected error equal to $\mathbb{E}(\|f-\hat{f}\|_{L_2(d\rho)}^2) = \frac{1}{n}\mathbb{E}\|g(v)\varphi(v,\cdot)\|_{L_2(d\rho)}^2 \leqslant \frac{1}{n}\sup_{v\in\mathcal{V}}\|\varphi(v,\cdot)\|_{L_2(d\rho)}^2$; our goal is to obtain an error rate with a better scaling in n, by (a) choosing a better distribution than $d\tau$ for the points $v_1,\dots,v_n$ and (b) by finding the best possible weights $\alpha \in \mathbb{R}^n$ (that should of course depend on the function g).

Goals. We thus aim at sampling n points $v_1,\dots,v_n \in \mathcal{V}$ from a distribution with density q with respect to $d\tau$. Then the kernel approximation using *importance weights* is equal to

$$\hat{k}(x,y) = \frac{1}{n}\sum_{i=1}^n \frac{1}{q(v_i)}\varphi(v_i,x)\varphi(v_i,y)$$

(so that the law of large numbers leads to an approximation converging to k), and we thus aim at minimizing $\Big\|\sum_{i=1}^n \frac{\beta_i}{q(v_i)^{1/2}}\varphi(v_i,\cdot) - \int_{\mathcal{V}} g(v)\varphi(v,\cdot)d\tau(v)\Big\|_{L_2(d\rho)}$, with $n\|\beta\|_2^2$ (which represents the norm of the approximation in $\hat{\mathcal{F}}$ because of our importance weights are taken into account) as small as possible.

3. Quadrature in RKHSs

Given a square-integrable (with respect to $d\rho$) function $g : \mathcal{X} \to \mathbb{R}$, the quadrature problem aims at approximating, for all $h \in \mathcal{F}$, integrals

$$\int_{\mathcal{X}} h(x)g(x)d\rho(x)$$

3. Note the factor n because our finite-dimensional kernel in Eq. (5) is an average of kernels and not a sum.

by linear combinations

$$\sum_{i=1}^n \alpha_i h(x_i)$$

of evaluations $h(x_1), \dots, h(x_n)$ of the function h at well-chosen points $x_1, \dots, x_n \in \mathcal{X}$. Of course, coefficients $\alpha \in \mathbb{R}^n$ are allowed to depend on g (they will in linear fashion in the next section), but not on h, as the so-called quadrature rule has to be applied to all functions in $\mathcal{F}$.

3.1 Approximation of the Mean Element

Following Smola et al. (2007), the error may be expressed using the reproducing property as:

$$\sum_{i=1}^n \alpha_i h(x_i) - \int_{\mathcal{X}} h(x) g(x) d\rho(x) = \left\langle h, \sum_{i=1}^n \alpha_i k(\cdot, x_i) - \int_{\mathcal{X}} k(\cdot, x) g(x) d\rho(x) \right\rangle_{\mathcal{F}},$$

and by Cauchy-Schwarz inequality its supremum over $\|h\|_{\mathcal{F}} \leqslant 1$ is equal to

$$\left\| \sum_{i=1}^n \alpha_i k(\cdot, x_i) - \int_{\mathcal{X}} k(\cdot, x) g(x) d\rho(x) \right\|_{\mathcal{F}}. \tag{7}$$

The goal of quadrature rules formulated in a RKHS is thus to find points $x_1, \dots, x_n \in \mathcal{X}$ and weights $\alpha \in \mathbb{R}^n$ so that the quantity in Eq. (7) is as small as possible (Smola et al., 2007). For $g = 1$, the function $\int_{\mathcal{X}} k(\cdot, x) d\rho(x)$ is usually referred to as the mean element of the distribution $d\rho$.

The standard Monte-Carlo solution is to consider $x_1, \dots, x_n$ sampled i.i.d. from $d\rho$ and the weights $\alpha_i = g(x_i)/n$, which leads to a decrease of the error in $1/\sqrt{n}$, with $\mathbb{E}\|\alpha\|_2^2 \leqslant \frac{1}{n}$ and an expected squared error which is equal to $\frac{1}{n}\mathbb{E}\|g(v)k(\cdot, x)\|_{\mathcal{F}}^2 \leqslant \frac{1}{n}\|g\|_{L_2(d\rho)}^2 \sup_{x \in \mathcal{X}} k(x, x)$ (Smola et al., 2007). Note that when $g = 1$, Eq. (7) corresponds to a particular metric between the distribution $d\rho$ and its corresponding empirical distribution (Sriperumbudur et al., 2010).

In this paper, we explore sampling points x_i from a probability distribution on $\mathcal{X}$ with density q with respect to $d\rho$. Note that when g is a constant function, it is sometimes required that the coefficients α are non-negative and sum to a fixed constant (so that constant functions are exactly integrated). We will not pursue this here as our theoretical results do not accommodate such constraints (see, e.g., Chen et al., 2010; Bach et al., 2012, and references therein).

Tolerance to noisy function values. In practice, independent (but not necessarily identically distributed) noise ε_i may be present with variance $\sigma^2(x_i)$. Then, the worst (with respect to $\|h\|_{\mathcal{F}} \leqslant 1$) expected (with respect to the noise) squared error is

$$\begin{aligned} &\inf_{\|h\|_{\mathcal{F}} \leqslant 1} \mathbb{E} \left| \sum_{i=1}^n \alpha_i (h(x_i) + \varepsilon_i) - \int_{\mathcal{X}} h(x) g(x) d\rho(x) \right|^2 \\ = \ & \left\| \sum_{i=1}^n \alpha_i k(\cdot, x_i) - \int_{\mathcal{X}} k(\cdot, x) g(x) d\rho(x) \right\|_{\mathcal{F}}^2 + \sum_{i=1}^n \alpha_i^2 \sigma^2(x_i), \end{aligned}$$

and thus in order to be robust to noise, having a small weighted ℓ_2-norm for the coefficients $\alpha \in \mathbb{R}^n$ is important.

3.2 Reformulation as Random Features

For any $x \in \mathcal{X}$, the function $k(\cdot, x)$ is in $\mathcal{F}$, and since we have assumed that $\Sigma^{1/2}$ is an isometry from $L_2(d\rho)$ to $\mathcal{F}$, there exists a unique element, which we denote $\psi(\cdot, x)$, of $L_2(d\rho)$ such that $\Sigma^{1/2}\psi(\cdot, x) = k(\cdot, x)$. Given the Mercer decomposition $k(\cdot, x) = \sum_{m \geqslant 1} \mu_m e_m(x) e_m$, we have the expansion $\psi(\cdot, x) = \sum_{m \geqslant 1} \mu_m^{1/2} e_m(x) e_m$ (with convergence in the L_2-norm for the measure $d\rho \otimes d\rho$; note that we do not assume that $\mu_m^{1/2}$ is summable), and thus we may consider ψ as a symmetric function. Note that ψ may not be easy to compute in many practical cases (except for some periodic kernels on $[0, 1]$).

We thus have for $(x, y) \in \mathcal{X} \times \mathcal{X}$:

$$\begin{aligned} k(x, y) &= \langle k(\cdot, x), k(\cdot, y) \rangle_{\mathcal{F}} = \langle \Sigma^{1/2}\psi(\cdot, x), \Sigma^{1/2}\psi(\cdot, y) \rangle_{\mathcal{F}} = \langle \psi(\cdot, x), \psi(\cdot, y) \rangle_{L_2(d\rho)} \\ &\qquad \text{because of the isometry property of } \Sigma^{1/2}, \\ &= \int_{\mathcal{X}} \psi(v, x)\psi(v, y) d\rho(v). \end{aligned} \tag{8}$$

That is, the kernel k may always be written as an expectation. Moreover, we have the quadrature error in Eq. (7) equal to (again using the isometry $\Sigma^{1/2}$ from $L_2(d\rho)$ to $\mathcal{F}$):

$$\begin{aligned} \left\| \sum_{i=1}^n \alpha_i k(\cdot, x_i) - \int_{\mathcal{X}} k(\cdot, x) g(x) d\rho(x) \right\|_{\mathcal{F}} &= \left\| \sum_{i=1}^n \alpha_i \Sigma^{1/2}\psi(x_i, \cdot) - \int_{\mathcal{X}} \Sigma^{1/2}\psi(x, \cdot) g(x) d\rho(x) \right\|_{\mathcal{F}} \\ &= \left\| \sum_{i=1}^n \alpha_i \psi(x_i, \cdot) - \int_{\mathcal{X}} \psi(x, \cdot) g(x) d\rho(x) \right\|_{L_2(d\rho)}, \end{aligned}$$

which is exactly an instance of the approximation result in Eq. (6) with $\mathcal{V} = \mathcal{X}$ and $\varphi = \psi$, that is the random feature is indexed by the same set $\mathcal{X}$ as the kernel. Thus, the quadrature problem, that is finding points x_i and weights (α_i) to get the best possible error over all functions of the unit ball of $\mathcal{F}$, is a *subcase* of the random feature problem for a specific expansion. Note that this random decomposition in terms of ψ is always possible (although not in closed form in general).

Interpretation through square-roots of intergral operators. As shown in Section 2.2, random feature expansions correspond to square-roots of the integral operator $\Sigma : L_2(d\rho) \to L_2(d\rho)$ as $\Sigma = TT^*$. Among the many possible square roots, the quadrature case corresponds exactly to the positive self-adjoint square root $T = \Sigma^{1/2}$. In this situation, the basis $(f_m)_{m \geqslant 1}$ of the singular value decomposition of $T = \Sigma^{1/2}$ is equal to $(e_m)_{m \geqslant 1}$, recovering the expansion $\psi(x, y) = \sum_{m \geqslant 1} \mu_m^{1/2} e_m(x) e_m(y)$ which we have seen above.

Translation-invariant kernels on $[0, 1]^d$ or $\mathcal{X} = \mathbb{R}^d$. In this important situation, we have two different expansions: the one based on Fourier features, where the random variable indexing the one-dimensional feature is a *frequency*, while for the one based on the square root ψ, the random variable is a *spatial variable* in $\mathcal{X}$. As we show in Section 4, our results are independent of the chosen expansions and thus apply to both. However, (a) when the

goal is to do quadrature, we need to use ψ, and (b) in general, the decomposition based on Fourier features can be easily computed once samples are obtained, while for most kernels, $\psi(x, y)$ does not have any closed-form simple expression. In Section 6, we provide a simple example with $\mathcal{X} = [0, 1]$ where the two decompositions are considered.

Goals. In order to be able to make the parallel with random feature approximations, we consider importance-weighted coefficients $\beta_i = \alpha_i q(x_i)^{1/2}$, and we thus aim at minimizing the approximation error

$$\Big\| \sum_{i=1}^n \beta_i q(x_i)^{-1/2} k(\cdot, x_i) - \int_{\mathcal{X}} k(\cdot, x) g(x) d\rho(x) \Big\|_{\mathcal{F}}.$$

We consider potential independent noise with variance $\sigma^2(x_i) \leqslant \tau^2 q(x_i)$ for all x_i, so that the tolerance to noise is characterized by the ℓ_2-norm $\|\beta\|_2$.

3.3 Relationship with Column Sampling

The problem of quadrature is related to the problem of column sampling. Given n observations $x_1, \dots, x_n \in \mathcal{X}$, the goal of column-sampling methods is to approximate the $n \times n$ matrix of pairwise kernel evalulations, the so-called *kernel matrix*, from a subset of its columns. It has appeared under many names: Nyström method (Williams and Seeger, 2001), sparse greedy approximations (Smola and Schölkopf, 2000), incomplete Cholesky decomposition (Fine and Scheinberg, 2001), Gram-Schmidt orthonormalization (see, e.g., Shawe-Taylor and Cristianini, 2004) or CUR matrix decompositions (Mahoney and Drineas, 2009).

While column sampling has typically been analyzed for a fixed kernel matrix, it has a natural extension which is related to quadrature problems: selecting n points $x_1, \dots, x_n$ from $\mathcal{X}$ such that the projection of any element of the RKHS $\mathcal{F}$ onto the subspace spanned by $k(\cdot, x_i)$, $i = 1, \dots, n$ is as small as possible. Natural functions from $\mathcal{F}$ are $k(\cdot, x)$, $x \in \mathcal{X}$, and thus the goal is to minimize, for such $x \in \mathcal{X}$,

$$\inf_{\alpha \in \mathbb{R}^n} \Big\| \sum_{i=1}^n \alpha_i k(\cdot, x_i) - k(\cdot, x) \Big\|_{\mathcal{F}}^2.$$

In the usual sampling approach, several points are considered for testing the projection error, and it is thus natural to consider the criterion averaged through the measure $d\rho$, that is:

$$\int_{\mathcal{X}} \inf_{\alpha \in \mathbb{R}^n} \Big\| \sum_{i=1}^n \alpha_i k(\cdot, x_i) - k(\cdot, x) \Big\|_{\mathcal{F}}^2 d\rho(x).$$

In fact, when $d\rho$ is supported on a finite set, this formulation is equivalent to minimizing the nuclear norm between the kernel matrix and its low-rank approximation. There are thus several differences and similarities between recent work on column sampling (Bach, 2013; El Alaoui and Mahoney, 2015) and the present paper on quadrature rules and random features:

- **Different error measures**: The column sampling approach corresponds to a function g in Eq. (7) which is a Dirac function at the point x, and is thus not in $L_2(d\rho)$. Thus the two frameworks are not equivalent.

- **Approximation vs. prediction**: The works by Bach (2013); El Alaoui and Mahoney (2015) aim at understanding when column sampling leads to no loss in predictive performance within a supervised learning framework, while the present paper looks at approximation properties, mostly regardless of any supervised learning problem, except in Section 4.5 for random features (but not for quadrature).
- **Lower bounds**: In Section 4.3, we provide explicit lower bounds of approximations, which are not available for column sampling.
- **Similar sampling issues**: In the two frameworks, points $x_1, \dots, x_n \in \mathcal{X}$ are sampled i.i.d. with a certain distribution q, and the best choice depends on the appropriate notion of leverage scores (Mahoney, 2011), while the standard uniform distribution leads to an inferior approximation result. Moreover, the proof techniques are similar and based on concentration inequalities for operators, here in Hilbert spaces rather in finite dimensions.

3.4 Related Work on Quadrature

Many methods have been designed for the computation of integrals of a function given evaluations at certain well-chosen points, in most cases when g is constant equal to one. We review some of these below.

Uni-dimensional integrals. When the underlying set $\mathcal{X}$ is a compact interval of the real line, several methods exists, such as the trapezoidal or Simpson's rules, which are based on interpolation between the sample points, and for which the error decays as $O(1/n^2)$ and $O(1/n^4)$ for functions with uniformly bounded second or fourth derivatives (Cruz-Uribe and Neugebauer, 2002).

Gaussian quadrature is another class of methods for one-dimensional integrals: it is based on a basis of orthogonal polynomials for $L_2(d\rho)$ where $d\rho$ is a probability measure supported in an interval, and their zeros (Hildebrand, 1987, Chap. 8). This leads to quadrature rules which are exact for polynomials of degree $2n - 1$ but error bounds for non-polynomials rely on high-order derivatives, although the empirical performance on functions of a Sobolev space in our experiments is as good as optimal quadrature schemes (see Section 6); depending on the orthogonal polynomials, we get various quadrature rules, such as Gauss-Legendre quadrature for the Lebesgue measure on $[0, 1]$.

Quasi Monte-carlo methods employ a sequence of points with low discrepancy with uniform weights (Morokoff and Caflisch, 1994), leading to approximation errors of $O(1/n)$ for univariate functions with bounded variation, but typically with no adaptation to smoother functions.

Higher-dimensional integrals. All of the methods above may be generalized for products of intervals $[0, 1]^d$, typically with d small. For larger problems, Bayes-Hermite quadrature (O'Hagan, 1991) is essentially equivalent to the quadrature rules we study in this paper.

Some of the quadrature rules are constrained to have positive weights with unit sum (so that the positivity properties of integrals are preserved and constants are exactly inegrated). The quadrature rules we present do not satisfy these constraints. If these constraints are required, kernel herding (Chen et al., 2010; Bach et al., 2012) provides a novel way to select

a sequence of points based on the conditional gradient algorithm, but with currently no convergence guarantees improving over $O(1/\sqrt{n})$ for infinite-dimensional spaces.

Theoretical results. The best possible error for a quadrature rule with n points has been well-studied in several settings; see Novak (1988) for a comprehensive review. For example, for $\mathcal{X} = [0, 1]$ and the space of Sobolev functions, which are RKHSs with eigenvalues of their integral operator decreasing as m^{-2s}, Novak (1988, Prop. 2 and 3, page 38) shows that the best possible quadrature rule for the uniform distribution and $g = 1$ leads to an error rate of n^{-s}, as well as for any squared-integrable function g. The proof of these results (both upper and lower bounds) relies on detailed properties of Sobolev spaces. In this paper, we recover these results using only the decay of eigenvalues of the associated integral operator Σ, thus allowing straightforward extensions to many situations, like Sobolev spaces on manifolds such as hyperspheres (Hesse, 2006), where we also recover existing results (up to logarithmic terms).

Moreover, Novak (1988, page 17) shows that adaptive quadrature rules where points are selected sequentially with the knowledge of the function values at previous points cannot improve the worst-case guarantees. Our results do not recover this lower bound result for adaptivity.

Finally, Langberg and Schulman (2010) consider multiplicative errors in computing integrals and mainly focuses on different function spaces, such as ones used in clustering functionals. Although sampling quadrature points from a well-chosen density is common in the two approaches, the analysis tools are different. It would be interesting to see if some of these tools can be transferred to our RKHS setting.

From quadrature to function approximation and optimization. The problem of quadrature, uniformly over all functions $g \in L_2(d\rho)$ that define the integral, is in fact equivalent to the full approximation of a function h given values at n points, where the approximation error is characterized in L_2-norm. Indeed, given the observations $h(x_i)$, $i = 1, \dots, n$, we build $\sum_{i=1}^n \alpha_i h(x_i)$ as an approximation of $\int_{\mathcal{X}} g(x)h(x)d\rho(x)$. It turns out that the coefficients α_i are linear in g, that is, there exists $a_i \in L_2(d\rho)$ such that $\alpha_i = \langle a_i, g\rangle_{L_2(d\rho)}$. This implies that $\sum_{i=1}^n h(x_i)\langle a_i, g\rangle_{L_2(d\rho)}$ is an approximation of $\langle h, g\rangle_{L_2(d\rho)}$. Thus, the worst case error with respect to g in the unit ball of $L_2(d\rho)$ is $\big\| \sum_{i=1}^n h(x_i)a_i - h\big\|_{L_2(d\rho)}$, that is, we have an approximation result of h through observations of its values at certain points.

Novak (1988) considers the approximation problem in L_∞-norm and shows that for Sobolev spaces, going from L_2- to L_∞-norms incurs a loss of performance of $\sqrt{n}$. We recover partially these results in Section 5 from a more general perspective. When optimizing the points at which the function is evaluated (adaptively or not), the approximation problem is often referred to as experimental design (Cochran and Cox, 1957; Chaloner and Verdinelli, 1995).

Finally, a third problem is of interest (and outside of the scope of this paper), namely the problem of finding the minimum of a function given (potentially noisy) function evaluations. For noiseless problems, Novak (1988, page 26) shows that the approximation and optimization problems have the same worst-case guarantees (with no influence of adaptivity); this optimization problem has also been studied in the bandit setting (Srinivas et al., 2012) and in the framework of "Bayesian optimization" (see, e.g. Bull, 2011).

4. Theoretical Analysis

In this section, we provide approximation bounds for the random feature problem outlined in Section 2.4 (and thus the quadrature problem in Section 3). In Section 4.1, we provide generic upper bounds, which depend on the eigenvalues of the integral operator Σ and present matching lower bounds (up to logarithmic terms) in Section 4.3. The upper-bound depends on specific distributions of samples that we discuss in Section 4.2. We then consider consequences of these results on quadrature (Section 4.4) and random feature expansions (Section 4.5).

4.1 Upper Bound

The following proposition (see proof in Appendix B.1) determines the minimal number of samples required for a given approximation accuracy:

Proposition 1 (Approximation of the unit ball of $\mathcal{F}$) *For $\lambda > 0$ and a distribution with positive density q with respect to $d\tau$, we consider*

$$d_{\max}(q, \lambda) = \sup_{v \in \mathcal{V}} \frac{1}{q(v)} \langle \varphi(v, \cdot), (\Sigma + \lambda I)^{-1} \varphi(v, \cdot)\rangle_{L_2(d\rho)}. \tag{9}$$

Let $v_1, \dots, v_n$ be sampled i.i.d. from the density q, then for any $\delta \in (0, 1)$, if

$$n \geqslant 5 d_{\max}(q, \lambda) \log \frac{16 d_{\max}(q, \lambda)}{\delta},$$

with probability greater than $1 - \delta$, we have $\frac{1}{n}\sum_{i=1}^n q(v_i)^{-1}\|\varphi(v_i, \cdot)\|^2_{L_2(d\rho)} \leqslant \frac{2 \operatorname{tr} \Sigma}{\delta}$ and

$$\sup_{\|f\|_{\mathcal{F}} \leqslant 1} \inf_{\|\beta\|_2^2 \leqslant \frac{4}{n}} \Big\| f - \sum_{i=1}^n \beta_i q(v_i)^{-1/2} \varphi(v_i, \cdot) \Big\|^2_{L_2(d\rho)} \leqslant 4\lambda.$$

We can interpret the proposition above as follows: given any squared error $4\lambda > 0$ and a distribution with density q, the number n of samples from q needed so that the unit ball of $\mathcal{F}$ is approximated by the ball of radius 2 of $\hat{\mathcal{F}}$ is, up to logarithmic terms, at most a constant times $d_{\max}(q, \lambda)$, defined in Eq. (9). The result above is a statement for a fixed q and λ and this number of samples n depends on these.

We could also invert the relationship between λ and n, that is, answer the following question: given a fixed number n of samples, what is the approximation error λ? This requires inverting the function $\lambda \mapsto d_{\max}(q, \lambda)$. This will be done in Section 4.2 for a specific distribution q where the expression simplifies, together with specific examples from Section 2.3.

Finally, note that we also have a bound on $\frac{1}{n}\sum_{i=1}^n q(v_i)^{-1}\|\varphi(v_i, \cdot)\|^2_{L_2(d\rho)}$, which shows that our random functions are not too large on average (this constraint will be needed in the lower bound as well in Section 4.3).

Sketch of proof. The proof technique relies on computing an explicit candidate $\beta \in \mathbb{R}^n$ obtained from minimizing a regularized least-squares formulation

$$\inf_{\beta \in \mathbb{R}^n} \Big\| \sum_{i=1}^n \beta_i q(v_i)^{-1/2} \varphi(v_i, \cdot) - f \Big\|^2_{L_2(d\rho)} + n\lambda \|\beta\|_2^2.$$

It turns out that the final bound on the squared error is exactly proportional to the regularization parameter λ. As shown in Appendix B.1, this leads to an approximation $\hat{f}$ which is a linear function of f, as $\hat{f} = (\hat{\Sigma} + \lambda I)^{-1}\hat{\Sigma} f$, where $\hat{\Sigma}$ is a properly defined empirical integral operator and $\lambda > 0$ is the regularization parameter. Then, Bernstein concentration inequalities for operators (Minsker, 2011) can be used in a way similar to the work of Bach (2013); El Alaoui and Mahoney (2015) on column sampling, to provide a bound on all desired quantities.

Result in expectation. In Section 4.5, we will need a result in expectation. As shown at the end of Appendix B.1, as soons as, $\lambda \leqslant (\operatorname{tr}\Sigma)/4$ and $n \geqslant 5 d_{\max}(\lambda)\log\frac{2(\operatorname{tr}\Sigma)d_{\max}(\lambda)}{\lambda}$, then

$$\mathbb{E}\left(\sup_{\|f\|_{\mathcal{F}}\leqslant 1}\ \inf_{\|\beta\|_2^2\leqslant\frac{4}{n}}\left\|f-\sum_{i=1}^{n}\beta_i q(v_i)^{-1/2}\varphi(v_i,\cdot)\right\|_{L_2(d\rho)}^2\right)\leqslant 8\lambda.$$

4.2 Optimized Distribution

We may now consider a specific distribution that depends on the kernel and on λ, namely

$$q_\lambda^*(v)=\frac{\langle\varphi(v,\cdot),(\Sigma+\lambda I)^{-1}\varphi(v,\cdot)\rangle_{L_2(d\rho)}}{\int_{\mathcal{V}}\langle\varphi(v,\cdot),(\Sigma+\lambda I)^{-1}\varphi(v,\cdot)\rangle_{L_2(d\rho)}d\tau(v)}=\frac{\langle\varphi(v,\cdot),(\Sigma+\lambda I)^{-1}\varphi(v,\cdot)\rangle_{L_2(d\rho)}}{\operatorname{tr}\Sigma(\Sigma+\lambda I)^{-1}},\quad(10)$$

for which $d_{\max}(q_\lambda^*,\lambda)=d(\lambda)=\operatorname{tr}\Sigma(\Sigma+\lambda I)^{-1}$. With this distribution, we thus need to have $n\geqslant 5d(\lambda)\log\frac{16d(\lambda)}{\delta}$ with $d(\lambda)=\operatorname{tr}\Sigma(\Sigma+\lambda I)^{-1}$ is the *degrees of freedom*, a traditional quantity in the analysis of least-squares regression (Hastie and Tibshirani, 1990; Caponnetto and De Vito, 2007), which is always smaller than $d_{\max}(1,\lambda)$ and can be upper-bounded explicitly for many examples, as we now explain. The computation of $d_{\max}(1,\lambda)$ in the operator setting (for which we may use $q=1$), a quantity often referred to as the maximal *leverage score* (Mahoney, 2011), remains an open problem.

The quantity $d(\lambda)$ only depends on the integral operator Σ, that is, for all possible choices of square roots, i.e., all possible choices of feature expansions, the number of samples that our results guarantee is the same. This being said, some expansions may be more computationally practical than others, and when using the distribution with $q(v)=1$, the bounds will be different.

Expression in terms of singular value decomposition. Given the singular value decomposition of φ in Eq. (3), we have, for any $v\in\mathcal{V}$, $\varphi(v,\cdot)=\sum_{m\geqslant 1}\mu_m^{1/2}f_m(v)e_m$ and thus

$$q_\lambda^*(v)\propto\langle\varphi(v,\cdot),(\Sigma+\lambda I)^{-1}\varphi(v,\cdot)\rangle_{L_2(d\rho)}=\sum_{m\geqslant 1}\frac{\mu_m}{\mu_m+\lambda}f_m(v)^2,$$

which provides an explicit expression for the density q_λ^*.

For a given squared error value λ, the optimized distribution q_λ^*, while leading to the degrees of freedom that will happen to be optimal in terms of approximation, has two main drawbacks:

- **Dependence on** λ: this implies that if we want a reduced error (i.e., a smaller λ), then the samples obtained from a higher λ, may not be reused to provably obtain the desired bound; in other words, the sampling is not *anytime*. For specific examples, e.g., quadrature with periodic kernels on $[0,1]$ with the uniform distribution, then $q=1$ happens to be optimal for all λ, and thus, we may reuse samples for different values of the error.
- **Hard to compute in practice**: the optimal distribution depends on a *leverage score* $\langle\varphi(v,\cdot),(\Sigma+\lambda I)^{-1}\varphi(v,\cdot)\rangle_{L_2(d\rho)}$, which may be hard to use for several reasons; first, it requires access to the infinite-dimensional operator Σ, which may be difficult; moreover, even if it possible to invert $\Sigma+\lambda I$, the set $\mathcal{V}$ might be particularly large and impractical to sample from. At the end of Section 4.1, we propose a simple algorithm based on sampling.

Eigenvalues and degrees of freedom. In order to relate more directly to the eigenvalues of Σ, we notice that we may lower bound the degrees of freedom by a constant times the number $m^*(\lambda)$ of eigenvalues greater than λ:

$$d(\lambda)=\operatorname{tr}\Sigma(\Sigma+\lambda I)^{-1}=\sum_{m\geqslant 1}\frac{\mu_m}{\mu_m+\lambda}\geqslant\sum_{\mu_m\geqslant\lambda}\frac{\mu_m}{\mu_m+\lambda}\geqslant\frac{1}{2}\max(\{m,\ \mu_m\geqslant\lambda\})=m^*(\lambda),$$

as defined in Section 2.1.

Moreover, we have the upper-bound:

$$d(\lambda)=\sum_{\mu_m\geqslant\lambda}\frac{\mu_m}{\mu_m+\lambda}+\sum_{\mu_m<\lambda}\frac{\mu_m}{\mu_m+\lambda}\leqslant\max(\{m,\ \mu_m\geqslant\lambda\})+\frac{1}{\lambda}\sum_{\mu_m<\lambda}\mu_m.$$

We now make the assumption that there exists a $\gamma>0$ independent of j such that

$$\forall j\geqslant 1,\quad\sum_{m=j}^{\infty}\mu_m\leqslant\gamma j\mu_j.\qquad(11)$$

This assumption essentially states that the eigenvalues decay sufficiently homogeneously and is satisfied by $\mu_m\propto m^{-2\alpha}$ with $\gamma=(2\alpha-1)^{-1}$, $\mu_m\propto r^m$ with $\gamma=(1-r)^{-1}$ and similar bounds also hold for all examples in Section 2.3. It allows us to relate the degrees of freedom directly to eigenvalue decays.

Indeed, this implies that $\frac{1}{\lambda}\sum_{\mu_m<\lambda}\mu_m\leqslant\gamma\max(\{m,\ \mu_m\geqslant\lambda\})=\gamma m^*(\lambda)$ for all $\lambda\leqslant\mu_1$ (the largest eigenvalue) and thus

$$\frac{1}{2}m^*(\lambda)\leqslant d\leqslant\big[1+\gamma\big]m^*(\lambda).$$

We can now restate the approximation result of Prop. 1 from Section 4.1 with the optimized distribution (see proof in Appendix B.2):

Proposition 2 (Approximation of the unit ball of $\mathcal{F}$ for optimized distribution) *For $\lambda>0$ and the distribution with density q_λ^* defined in Eq. (10) with respect to $d\tau$, with degrees of freedom $d(\lambda)$. Let $v_1,\dots,v_n$ be sampled i.i.d. from the density q_λ^*, defining the kernel (and its associated RKHS $\hat{\mathcal{F}}$) $\hat{k}(x,y)=\frac{1}{n}\sum_{i=1}^{n}\frac{1}{q(v_i)}\varphi(v_i,x)\varphi(v_i,y)$. Then, for any $\delta\in(0,1)$, with probability $1-\delta$, we have:*

$$\sup_{\|f\|_{\mathcal{F}}\leqslant 1}\ \inf_{\|\hat{f}\|_{\hat{\mathcal{F}}}\leqslant 2}\left\|f-\hat{f}\right\|_{L_2(d\rho)}^2\leqslant 4\lambda,$$

under any of the following conditions:

(a) if $n \geqslant 5\ d(\lambda) \log\big[16 d(\lambda)/\delta\big]$,

(b) if Eq. (11) is satisfied, and, by choosing $m \leqslant \frac{n}{5(1+\gamma)\log\frac{16n}{5\delta}}$, *and* $\lambda = \mu_m$.

The statement (a) above, is a simple corollary of Prop. 1, and goes from level of error λ to minimum number n of samples. The statement (b) goes in the other direction, that is, from the number of samples n to the achieved approximation error. It depends on the eigenvalues μ_m of the integral operator taken at $m = O(n/\log(n))$. For example, for polynomial decays of eigenvalues of the form $\mu_m = O(m^{-2s})$, we get (non squared) errors proportional to $(\log n)^s n^{-s}$ for n samples, while for geometric decays, we get geometric errors as a function of the number n of samples.

Note however that for the statement (b) to hold, we need to sample the points $v_1, \dots, v_n$ from the distribution $q^*_{\mu_m}$, that is, for different numbers of samples n, the distribution is unfortunately different (except in special cases). It would be interesting to study the properties of independent but *not identically distributed* samples $v_1, \dots, v_n$ and the possibility of achieving the same rate adaptively.

Corollary for Sobolev spaces. For the sake of concreteness, we consider the special case of $\mathcal{X} = \mathbb{R}^d$ and translation-invariant kernels. We assume that the distribution $d\rho$ is sub-Gaussian. Then for Sobolev spaces of order s, the eigenvalue decay is proportional to $m^{-2s/d}$. Thus, if we can sample from the optimized distribution, after n random features, we obtain an approximation of the unit ball of $\mathcal{F}$ with error $n^{-s/d}$, independently of the chosen expansion, the spatial one used for quadrature or the spectral one used in random Fourier features. For kernels in $\mathbb{R}^d$, these distributions are not readily computed in closed form and need to computed through a dedicated algorithm such as the one we present below.

The same approximation results holds for translation-invariant kernels on $[0,1]^d$; but when $d\rho$ is the uniform distribution, as shown in Section 4.4, the optimized distribution for the quadrature case is still the uniform distribution, for all values of λ, and can thus be computed.

Algorithm to estimate the optimized distribution. We now consider a simple algorithm for estimating the optimized distribution q^*_λ. It is based on using a large number N of points $v_1, \dots, v_N$ from $d\tau$, and replacing $d\tau$ by a potentially weighted empirical distribution $d\hat{\tau}$ associated with these N points. Therefore, we may use any set of points and weights, which leads to a distribution close to $d\tau$. In full generality, only random samples from $d\tau$ are readily available (with weights $1/N$), but for special cases, such as $\mathcal{V} = [0,1]$ or $\mathcal{V} = \mathbb{N}^*$, we may use deterministic representations. See examples in Section 6.

We thus assume that we have N pairs $(v_i, \eta_i) \in \mathcal{V} \times \mathbb{R}_+$, $i = 1, \dots, N$, such that $\sum_{i=1}^n \eta_i = 1$. Since $d\hat{\tau}$ has a finite support with at most N elements, we may identify $L_2(d\hat{\tau})$ and $\mathbb{R}^N$ (with its canonical dot-product), and the operator T goes now from $\mathbb{R}^N$ to $L_2(d\rho)$, with $Tg = \sum_{i=1}^N \eta_i^{1/2} g_i \varphi(v_i, \cdot) \in L_2(d\rho)$, with $T\delta_i = \eta_i^{1/2} \varphi(v_i, \cdot) \in L_2(d\rho)$, for δ_i the i-th element of the canonical basis of $\mathbb{R}^N$. Then, we have:

$$\begin{aligned}
\langle \varphi(v_i, \cdot), (\Sigma + \lambda I)^{-1} \varphi(v_i, \cdot) \rangle_{L_2(d\rho)} &= \eta_i^{-1} \langle T\delta_i, (TT^* + \lambda I)^{-1} T\delta_i \rangle_{L_2(d\rho)} \\
&= \eta_i^{-1} \langle T\delta_i, T(T^*T + \lambda I)^{-1} \delta_i \rangle_{L_2(d\rho)} \\
&= \eta_i^{-1} \big(T^* T (T^*T + \lambda I)^{-1} \big)_{ii}.
\end{aligned}$$

This implies that the density of the optimized distribution with respect to the uniform measure on $\{v_1, \dots, v_N\}$ is proportional to $\big(T^*T(T^*T + \lambda I)^{-1}\big)_{ii}$. We can then sample any number n of points from resampling from $\{v_1, \dots, v_N\}$ from the density above. The computational complexity is $O(N^3)$. A detailed analysis of the approximation properties of this algorithm is outside the scope of this paper.

We have $(T^*T)_{ij} = \eta_i^{1/2} \eta_j^{1/2} \int_{\mathcal{X}} \varphi(v_i, x) \varphi(v_j, x) d\rho(x)$. In some cases, it can be computed in closed form—such as for quadrature where this is equal to $\eta_i^{1/2} \eta_j^{1/2} k(v_i, v_j)$. In some others, it requires i.i.d. samples $x_1, \dots, x_M$ from $d\rho$, and the following estimate: $\eta_i^{1/2} \eta_j^{1/2} M^{-1} \sum_{k=1}^M \varphi(v_i, x_k) \varphi(v_j, x_k)$.

4.3 Lower Bound

In this section, we aim at providing lower-bounds on the number of samples required for a given accuracy. We have the following result (see proof in Appendix B.3):

Proposition 3 (Lower approximation bound) *For* $\delta \in (0,1)$, *if we have a family of functions* $\psi_1, \dots, \psi_n \in L_2(d\rho)$ *such that*

$$\frac{1}{n} \sum_{i=1}^n \|\psi_i\|^2_{L_2(d\rho)} \leqslant 2 \operatorname{tr} \Sigma / \delta, \quad \text{and} \quad \sup_{\|f\|_{\mathcal{F}} \leqslant 1} \inf_{\|\beta\|_2^2 \leqslant \frac{4}{n}} \Big\| f - \sum_{i=1}^n \beta_i \psi_i \Big\|^2_{L_2(d\rho)} \leqslant 4\lambda,$$

then $n \geqslant \dfrac{\max\{m,\ \mu_m \geqslant 144\lambda\}}{4 \log \frac{10 \operatorname{tr} \Sigma}{\lambda \delta}}$.

We can make the following observations:

- The proof technique not surprisingly borrows tools from minimax estimation over ellipsoids, namely the Varshamov-Gilbert's lemma.
- We obtain matching upper and lower bounds up to logarithmic terms, using only the decay of eigenvalues $(\mu_m)_{m \geqslant 1}$ of the integral operator Σ (of course, if sampling from the optimized distribution q^*_λ is possible). Indeed in that case, as shown in Prop. 2, we have shown that we need at most $10\ d(\lambda) \log\big[2d(\lambda)\big]$, where $d(\lambda)$ is the degrees of freedom, which is upper and lower bounded by a constant times $m^*(\lambda) = \max\{m,\ \mu_m \geqslant \lambda\}$.
- In order to obtain such a bound, we need to constrain both $\|\beta\|_2$ and the norms of the vectors ψ_i, which correspond to bounded features for the random feature interpretation and tolerance to noise for the quadrature interpretation. We choose our scaling to match the constraints we have in Prop. 1, for which the parameter δ ends up entering the lower bound logarithmically.

4.4 Quadrature

We may specialize the results above to the quadrature case, namely give a formulation where the features φ do not appear (or equivalently using ψ defined in Section 3.2). This is a special case where $\mathcal{V} = \mathcal{X}$ and $\varphi = \psi$. In terms of operators T in Section 2.2, this corresponds to $T = \Sigma^{1/2}$.

Optimized distribution. Following Section 4.1, we have an expression for the optimized distribution, both in terms of operators, as follows,

$$q_\lambda^*(x) \propto \langle \psi(x,\cdot), (\Sigma+\lambda I)^{-1}\psi(x,\cdot)\rangle_{L_2(d\rho)} = \langle \Sigma^{-1/2}k(x,\cdot), (\Sigma+\lambda I)^{-1}\Sigma^{-1/2}k(x,\cdot)\rangle_{L_2(d\rho)},$$

and in terms of eigenvalues and eigenvectors of k, that is,

$$q(x) \propto \langle k(\cdot,x), \Sigma^{-1/2}(\Sigma+\lambda I)^{-1}\Sigma^{-1/2}k(\cdot,x)\rangle_{L_2(d\rho)} = \sum_{m\geqslant 1}\frac{\mu_m}{\mu_m+\lambda}e_m(x)^2. \qquad (12)$$

While this is uniform in some special cases (uniform distribution on $[0,1]$ and Sobolev kernels, as shown below), this is typically hard to compute and sample from. An algorithm for approximating it was presented at the end of Section 4.1.

A weakness of our result is that in general our optimized distribution $q_\lambda^*(x)$ depends on λ and thus on the number of samples. In some cases with symmetries (i.e., uniform distribution on $[0,1]$ or the hypersphere), q_λ^* happens to be constant for all λ. Note also that we have observed empirically that in some cases, q_λ^* converges to a certain distribution when λ tends to zero (see an example in Section 6).

Sobolev spaces. For Sobolev spaces with order s in $[0,1]^d$ or $\mathbb{R}^d$ (for which we assume $d<2s$), the decay of eigenvalues is of the form $m^{-2s/d}$ and thus the error after n samples is $n^{-s/d}$ (up to logarithmic terms), which recovers the upper and lower bounds of Novak (1988, pages 37 and 38) (also up to logarithmic terms).

For the special case of Sobolev spaces on $[0,1]^d$ with $d\rho$ the uniform distribution, the optimized distribution in Eq. (12) is also the uniform distribution. Indeed, the eigenfunctions of the integral operator Σ are d-th order tensor products of the uni-dimensional Fourier basis (the constant and all pairs of sine/cosine at a given frequency), with the *same eigenvalue* for the 2^d possibilities of sines/cosines for a given multi-dimensional frequency $(m_1,\dots,m_d)$. Therefore, when summing all squared values of the eigenfunctions corresponding to $(m_1,\dots,m_d)$, we get the sum $\sum_{a\in\{0,1\}^d}\prod_{i=1}^d \cos^{2a_i}(2\pi m_i x_i)\sin^{2(1-a_i)}(2\pi m_i x_i)$, which ends up being constant equal to one (and thus independent of x) because of the identity $\cos^2(2\pi m_i x_i)+\sin^2(2\pi m_i x_i)=1$.

Finally, we may consider Sobolev spaces on the hypersphere, with the kernels presented in Section 2.3. As shown by Bach (2017, Appendix D.3), the kernel $k(x,y) = \mathbb{E}(v^\top y)_+^s(v^\top y)_+^s$ for v uniform on the hypersphere, leads to a Sobolev space of order $t = s+\frac{d+1}{2}$, while the decay of eigenvalue of the integral operator was shown to be $m^{-1-1/d-2s/d}$ in Section 2.3. It is thus equal to $m^{-2t/d}$, and we recover the result from Hesse (2006).

Quadrature rule. We assume that points $x_1,\dots,x_n$ are sampled from the distribution with density q with respect to $d\rho$. The quadrature rule for a function $h\in\mathcal{F}$ is $\sum_{i=1}^n \frac{\beta_i h(x_i)}{q(x_i)^{1/2}}$. To compute β, we need to minimize with respect to β the error:

$$\Big\| \sum_{i=1}^n \frac{\beta_i}{q(x_i)^{1/2}}k(\cdot,x_i) - \int_{\mathcal{X}} k(\cdot,x)g(x)d\rho(x)\Big\|_{\mathcal{F}}^2 + n\lambda\|\beta\|_2^2,$$

which is the regularized worst case squared error in the estimation of the integral of h over $h\in\mathcal{F}$. The best error is obtained for $\lambda=0$, but our guarantees are valid for $\lambda>0$, with an explicit control over the norm $\|\beta\|_2^2$, which is important for robustness to noise.

Given the values of $\int_{\mathcal{X}} k(x_i,x)g(x)d\rho(x) = z_i$, for $i=1,\dots,n$, which can be computed in closed form for several triplet $(k,g,d\rho)$ (see, e.g., Smola et al., 2007; Oates and Girolami, 2015), then the problem above is equivalent to minimizing with respect to β:

$$\sum_{i=1}^n\sum_{j=1}^n \frac{\beta_i\beta_j}{q(x_i)^{1/2}q(x_j)^{1/2}}k(x_i,x_j) - \sum_{i=1}^n \frac{\beta_i}{q(x_i)^{1/2}}z_i + n\lambda\|\beta\|_2^2,$$

which leads to a $n\times n$ linear system with running time complexity $O(n^3)$. Note that when adding points sequentially (in particular for kernels for which the distribution q_λ^* is independent of λ, such as Sobolev spaces on $[0,1]$), one may update the solution so that after n steps, the overall complexity is $O(n^3)$.

Approximation of functions in $\mathcal{F}$. With the quadrature weights β estimated above and the quadrature rule $\sum_{i=1}^n \frac{\beta_i h(x_i)}{q(x_i)^{1/2}}$ for the estimation of $\int_{\mathcal{X}} g(x)f(x)d\rho(x)$, we may derive an expression which is explicitly linear in g. Following the proof of Prop. 1 in Appendix B.1, we have, when specialized to the quadrature case:

$$\hat\Sigma = \frac{1}{n}\sum_{i=1}^n \frac{1}{q(v_i)}\psi(x_i,\cdot)\otimes_{L_2(d\rho)}\psi(x_i,\cdot) = \Sigma^{-1/2}\bigg(\frac{1}{n}\sum_{i=1}^n\frac{1}{q(v_i)}k(x_i,\cdot)\otimes_{L_2(d\rho)}k(x_i,\cdot)\bigg)\Sigma^{-1/2},$$

Moreover, we have $\beta_i = \frac{1}{nq(x_i)^{1/2}}\langle k(\cdot,x_i), \Sigma^{-1/2}(\hat\Sigma+\lambda I)^{-1}\Sigma^{1/2}g\rangle_{L_2(d\rho)}$ from Eq. (15) in Appendix B.1, and the quadrature rule becomes:

$$\begin{aligned}\sum_{i=1}^n \frac{\beta_i h(x_i)}{q(x_i)^{1/2}} &= \sum_{i=1}^n \frac{\beta_i}{q(x_i)^{1/2}}\langle h, \Sigma^{-1}k(\cdot,x_i)\rangle_{L_2(d\rho)}\\ &= \bigg\langle h, \frac{1}{n}\sum_{i=1}^n \Sigma^{-1}\frac{1}{q(x_i)}\big[k(x_i,\cdot)\otimes_{L_2(d\rho)}k(x_i,\cdot)\big]\Sigma^{-1/2}(\hat\Sigma+\lambda I)^{-1}\Sigma^{1/2}g\bigg\rangle_{L_2(d\rho)}\\ &= \big\langle h, \Sigma^{-1/2}\hat\Sigma(\hat\Sigma+\lambda I)^{-1}\Sigma^{1/2}g\big\rangle_{L_2(d\rho)} = \big\langle g, \Sigma^{1/2}\hat\Sigma(\hat\Sigma+\lambda I)^{-1}\Sigma^{-1/2}h\big\rangle_{L_2(d\rho)},\end{aligned}$$

which can be put in the form $\langle \hat h, g\rangle_{L_2(d\rho)}$ with the approximation $\hat h = \Sigma^{1/2}\hat\Sigma(\hat\Sigma+\lambda I)^{-1}\Sigma^{-1/2}h$ of the function $h\in\mathcal{F}$. Having a bound for all functions g such that $\|g\|_{L_2(d\rho)}\leqslant 1$ is equivalent to having a bound on $\|h-\hat h\|_{L_2(d\rho)}$. In Section 5, we consider extensions, where we consider other norms than the L_2-norm for characterizing the approximation error $\hat h - h$. Moreover, we consider cases where h belongs to a strict subspace of $\mathcal{F}$ (with improved results).

4.5 Learning with Random Features

We consider supervised learning with m i.i.d. samples from a distribution on inputs/outputs (x,y), and a uniformly G-Lipschitz-continuous loss function $\ell(y,\cdot)$, which includes logistic regression and the support vector machine. We consider the empirical risk $\hat L(f) = \frac{1}{m}\sum_{i=1}^m \ell(y_i,f(x_i))$ and the expected risk $L(f)=\mathbb{E}\ell(y,f(x))$, with x having the marginal distribution $d\rho$ that we consider in earlier sections. We assume that $\mathbb{E}k(x,x) = \operatorname{tr}\Sigma = R^2$. We have the usual generalization bound for the minimizer $\hat f$ of $\hat L(f)$ with respect to

$\|f\|_{\mathcal{F}} \leqslant F$, based on Rademacher complexity (see, e.g., Shalev-Shwartz and Ben-David, 2014):

$$\mathbb{E}\big[L(\hat{f})\big] \leqslant \inf_{\|f\|_{\mathcal{F}} \leqslant F} L(f) + 2\mathbb{E}\Big[\sup_{\|f\|_{\mathcal{F}} \leqslant F} |L(f) - \hat{L}(f)|\Big] \leqslant \inf_{\|f\|_{\mathcal{F}} \leqslant F} L(f) + \frac{4FGR}{\sqrt{m}}. \qquad (13)$$

We now consider learning by sampling n features from the optimized distribution from Section 4.2, leading to a function parameterized by $\beta \in \mathbb{R}^n$, i.e., $\hat{g}_\beta = \sum_{i=1}^n \beta_i q(v_i)^{-1/2} \varphi(v_i, \cdot)$, an element of $L_2(d\rho)$. Applying results from Section 4.1, we assume that $\lambda \leqslant R^2/4$ and $n \geqslant 5d(\lambda) \log \frac{2(\operatorname{tr}\Sigma) d(\lambda)}{\lambda}$, where $d(\lambda)$ is equal to the degrees of freedom associated with the kernel k and distribution $d\rho$. Thus, the expected squared error for approximating the unit-ball of $\mathcal{F}$ by the ball of radius 2 of the approximation $\hat{\mathcal{F}}$ obtained from the approximated kernel is less than 8λ.

If we consider the estimator $\hat{\beta}$ obtained by minimizing the empirical risk of $\hat{g}_\beta$ subject to $\|\beta\|_2 \leqslant 2F/\sqrt{n}$. We have the following decomposition of the error for any $\gamma \in \mathbb{R}^n$ such that $\|\gamma\|_2 \leqslant 2F/\sqrt{n}$ and $f \in \mathcal{F}$ such that $\|f\|_{\mathcal{F}} \leqslant F$:

$$\begin{aligned}
& L(\hat{g}_{\hat{\beta}}) \\
= \;& L(\hat{g}_{\hat{\beta}}) - \hat{L}(\hat{g}_{\hat{\beta}}) + \hat{L}(\hat{g}_{\hat{\beta}}) - \hat{L}(\hat{g}_\gamma) + \hat{L}(\hat{g}_\gamma) - L(\hat{g}_\gamma) + L(\hat{g}_\gamma) - L(f) + L(f) \\
\leqslant \;& 2\Big[\sup_{\|\beta'\|_{\mathcal{F}} \leqslant 2F/\sqrt{n}} |L(\hat{g}_{\beta'}) - L(\hat{g}_{\beta'})|\Big] + \big[L(\hat{g}_\gamma) - L(f)\big] + L(f) \\
\leqslant \;& 2\Big[\sup_{\|\beta'\|_{\mathcal{F}} \leqslant 2F/\sqrt{n}} |L(\hat{g}_{\beta'}) - L(\hat{g}_{\beta'})|\Big] + \sup_{\|f'\|_{\mathcal{F}} \leqslant F} \inf_{\|\gamma\|_2 \leqslant 2F/\sqrt{n}} \big[L(\hat{g}_\gamma) - L(f')\big] + \inf_{\|f\|_{\mathcal{F}} \leqslant F} L(f).
\end{aligned}$$

We now take expectation with respect to the data and the random features. Following standard results for Rademacher complexities of ℓ_2-balls (Bartlett and Mendelson, 2003, Lemma 22), the first term is less than

$$\frac{4FG}{m\sqrt{n}} \mathbb{E}\Big(\sum_{i=1}^m \sum_{j=1}^n \frac{\varphi(v_i, x_j)^2}{q(v_i)}\Big)^{1/2} \leqslant \frac{4FG}{m\sqrt{n}} (nm \operatorname{tr}\Sigma)^{1/2} = \frac{4FGR}{\sqrt{m}}.$$

Because of the G-Lipschitz-continuity of the loss, we have $L(\hat{g}_\gamma) - L(f') \leqslant G\|\hat{g}_\gamma - f'\|_{L_2(d\rho)}$, and thus the second term is less than $\sqrt{8\lambda} GF \leqslant 3GF\sqrt{\lambda}$. Overall, we obtain

$$\mathbb{E}\big[L(\hat{g}_{\hat{\beta}})\big] \leqslant \inf_{\|f\|_{\mathcal{F}} \leqslant F} L(f) + 3GF\sqrt{\lambda} + \frac{4FGR}{\sqrt{m}}.$$

If we consider $\lambda = R^2/m$ in order to lose only a constant factor compared to Eq. (13), we have the constraint $n \geqslant 5d(R^2/m) \log\big[2md(R^2/m)\big]$.

We may now look at several situations. In the worst case, where the decay of eigenvalue is not fast, i.e., very close to $1/i$, then we may only use the bound $d(\lambda) = \operatorname{tr}\Sigma(\Sigma + \lambda I)^{-1} \leqslant \lambda^{-1} \operatorname{tr}\Sigma = R^2/\lambda$, and thus a sufficient condition $n \geqslant 10m \log 2m$, and we obtain the same result as Rahimi and Recht (2009).

However, when we have eigenvalue decays as $R^2 i^{-2s}$, we get (up to constants), following the same computation as Section 4.2, $d(\lambda) \leqslant (R^2/\lambda)^{1/(2s)}$, and thus $n \geqslant m^{1/(2s)} \log m$, which is a significant improvement (regardless of the value of F). Moreover, if the decay is geometric as r^i, then we get $d(\lambda) \leqslant \log(R^2/\lambda)$, and thus $n \geqslant (\log m)^2$, which is even more significant.

5. Quadrature-related Extensions

In Section 4.4, we have built an approximation $\hat{h} = \Sigma^{1/2}\hat{\Sigma}(\hat{\Sigma} + \lambda I)^{-1}\Sigma^{-1/2} h$ of a function $h \in \mathcal{F}$, which is based on n function evaluations $h(x_1), \dots, h(x_n)$. We have presented in Section 4.4 a convergence rate for the L_2-norm $\|\hat{h} - h\|_{L_2(d\rho)}$ for functions h with less than unit $\mathcal{F}$-norm $\|h\|_{\mathcal{F}} \leqslant 1$. Up to logarithmic terms, if using the optimal distribution for sampling $x_1, \dots, x_n$, then we get a squared error of μ_n where μ_n is the n-th largest eigenvalue of the integral operator Σ.

Robustness to noise. We have seen that if the noise in the function evaluations $h(x_i)$ has a variance less than $q(x_i)\tau^2$, then the error $\|h - \hat{h}\|^2_{L_2(d\rho)}$ has an additional term $\tau^2 \|\beta\|_2^2 \leqslant \frac{4\tau^2}{n}$. Hence, the amount of noise has to be less than $n\mu_n$ in order to incur no loss in performance (a bound which decreases with n).

Adaptivity to smoother functions. We assume that the function h happens to be smoother than what is sufficient to be an element of the RKHS $\mathcal{F}$, that is, if $\|\Sigma^{-s} h\|_{L_2(d\rho)} \leqslant 1$, where $s \geqslant 1/2$. The case $s = 1/2$ corresponds to being in the RKHS. In the proof of Prop. 1 in Appendix B.1, we have seen that with high-probability we have:

$$(\hat{\Sigma} + \lambda I)^{-1} \preccurlyeq 4(\Sigma + \lambda I)^{-1}. \qquad (14)$$

We now see that we can bound the error $\|\hat{h} - h\|_{L_2(d\rho)}$ as follows:

$$\begin{aligned}
\|\hat{h} - h\|_{L_2(d\rho)} &= \|\Sigma^{1/2}\hat{\Sigma}(\hat{\Sigma} + \lambda I)^{-1}\Sigma^{-1/2} h - h\|_{L_2(d\rho)} \\
&= \lambda \big\|\Sigma^{1/2}(\hat{\Sigma} + \lambda I)^{-1}\Sigma^{-1/2+s}\Sigma^{-s} h\big\|_{L_2(d\rho)} \\
&\leqslant \lambda \big\|\Sigma^{1/2}(\hat{\Sigma} + \lambda I)^{-1/2}\big\|_{\mathrm{op}} \big\|(\hat{\Sigma} + \lambda I)^{-1/2}\Sigma^{-1/2+s}\big\|_{\mathrm{op}} \|\Sigma^{-s} h\|_{L_2(d\rho)}.
\end{aligned}$$

We may now bound each term. The first one $\big\|\Sigma^{1/2}(\hat{\Sigma} + \lambda I)^{-1/2}\big\|_{\mathrm{op}}$ is less than 2, because of Eq. (14). The second one $\big\|(\hat{\Sigma} + \lambda I)^{-1/2}\Sigma^{-1/2+s}\big\|_{\mathrm{op}}$ is equal to $\big\|(\hat{\Sigma} + \lambda I)^{s-1}(\hat{\Sigma} + \lambda I)^{1/2-s}\Sigma^{-1/2+s}\big\|_{\mathrm{op}}$, and thus less than $\big\|(\hat{\Sigma}+\lambda I)^{s-1}\|_{\mathrm{op}} \cdot \big\|(\hat{\Sigma}+\lambda I)^{1/2-s}\Sigma^{-1/2+s}\big\|_{\mathrm{op}} \leqslant 2\lambda^{s-1}$. Overall we obtain

$$\|\hat{h} - h\|_{L_2(d\rho)} \leqslant 4\lambda^s.$$

The norm $h \mapsto \|\Sigma^{-s} h\|_{L_2(d\rho)}$ is an RKHS norm with kernel $\sum_{m \geqslant 0} \mu_m^{2s} e_m(x) e_m(y)$, with corresponding eigenvalues equal to $(\mu_m)^{2s}$. From Prop. 2 and 3, the optimal number of quadrature points to reach a squared error less than ε is proportional to the number $\max(\{m,\ \mu_m^{2s} \geqslant \varepsilon\})$, while using the quadrature points from $s = 1/2$, leads to a number $\max(\{m,\ \mu_m \geqslant \varepsilon^{1/(2s)}\})$, which is equal. Thus if the RKHS used to compute the quadrature weights is a bit too large (but not too large, see experiments in Section 6), then we still get the optimal rate. Note that this robustness is only shown for the regularized estimation of the quadrature coefficients (in our simulations, the non-regularized ones also exhibit the same behavior).

Approximation with stronger norms. We may consider characterizing the difference $\hat{h} - h$ with different norms than $\|\cdot\|_{L_2(d\rho)}$, in particular norms $\|\Sigma^{-r}(\hat{h} - h)\|_{L_2(d\rho)}$, with

$r \in [0, 1/2]$. For $r = 0$, this is our results in L_2-norm, while for $r = 1/2$, this is the RKHS norms. We have, using the same manipulations than above:

$$\begin{aligned}\|\Sigma^{-r}(\hat{h} - h)\|_{L_2(d\rho)} &= \lambda \big\|\Sigma^{1/2-r}(\hat{\Sigma} + \lambda I)^{-1}\Sigma^{-1/2}h\big\|_{L_2(d\rho)} \\ &\leqslant \lambda^{1/2-r}\big\|\Sigma^{1/2-r}(\hat{\Sigma} + \lambda I)^{r-1/2}\big\|_{\rm op}\|\Sigma^{-1/2}h\|_{L_2(d\rho)} \leqslant 2\lambda^{1/2-r}.\end{aligned}$$

When $r = 1/2$, we get a result in the RKHS norm, but with no decay to zero; the RKHS norm $\|\cdot\|_{\mathcal{F}}$ would allow a control in L_∞-norm, but as noticed by Steinwart et al. (2009); Mendelson and Neeman (2010), such a control may be obtained in practice with r much smaller. For example, when the eigenfunctions e_m are uniformly bounded in L_∞-norm by a constant C (as is the case for periodic kernels in $[0, 1]$ with the uniform distribution), then, for any $x \in \mathcal{X}$, we have for $t > 1$,

$$f(x)^2 = \sum_{m=1}^{\infty}(m+1)^t\langle f, e_m\rangle^2_{L_2(d\rho)}e_m(x)^2(m+1)^{-t} \leqslant \sum_{m=0}^{\infty}(m+1)^t\langle f, e_m\rangle^2_{L_2(d\rho)}\frac{C^2}{t-1}.$$

If for simplicity, we assume that $\mu_m = (m+1)^{-2s}$ (like for Sobolev spaces), we have $\|\Sigma^{-r}f\|^2_{L_2(d\rho)} = \sum_{m=1}^{\infty}\mu_m^{-2r}\langle f, e_m\rangle^2_{L_2(d\rho)} = \sum_{m=1}^{\infty}(m+1)^t\langle f, e_m\rangle^2_{L_2(d\rho)}$ with $r = t/4s$. If $\lambda \leqslant O(n^{-2s})$ (as suggested by Prop. 1), then we obtain a squared L_∞-error less than $\frac{1}{t-1}\lambda^{1-2r} = O\big(\frac{1}{t-1}n^{-2s(1-t/2s)}\big) = O\big(\frac{n^t}{t-1}n^{-2s}\big)$. With $t = 1 + \frac{1}{\log n}$, we get $O\big(\frac{n\log n}{n^{-2s}}\big)$, and thus a degradation compared to the squared L_2-loss of n (plus additional logarithmic terms), which corresponds to the (non-improvable) result of Novak (1988, page 36).

6. Simulations

In this section, we consider simple illustrative quadrature experiments[4] with $\mathcal{X} = [0, 1]$ and kernels $k(x, y) = 1 + \sum_{m=1}^{\infty}\frac{1}{m^{2s}}\cos 2\pi m(x - y)$, with various values of s and distributions $d\rho$ which are Beta random variable with the two parameters equal to $a = b$, hence symmetric around $1/2$.

Uniform distribution. For $b = 1$, we have the uniform distribution on $[0, 1]$ for which the cosine/sine basis is orthonormal, and the optimized distribution q^*_λ is also uniform. Moreover, we have $\int_0^1 k(x, y)d\rho(x) = 1$. We report results comparing different Sobolev spaces for testing functions to integrate (parameterized by s) and learning quadrature weights (parameterized by t) in Figure 1, where we compute errors averaged over 1000 draws. We did not use regularization to compute quadrature weights α. We can make the following observations:

- The exponents in the convergence rates for $s = t$ (matching RKHSs for learning quadrature weights and testing functions) are close to $2s$ as expected.
- When the functions to integrate are less smooth than the ones used for learning quadrature weights (that is $t > s$), then the quadrature performance does not necessarily decay with the number of samples.

4. Matlab code for all 5 figures may be downloaded from `http://www.di.ens.fr/~fbach/quadrature.html`.

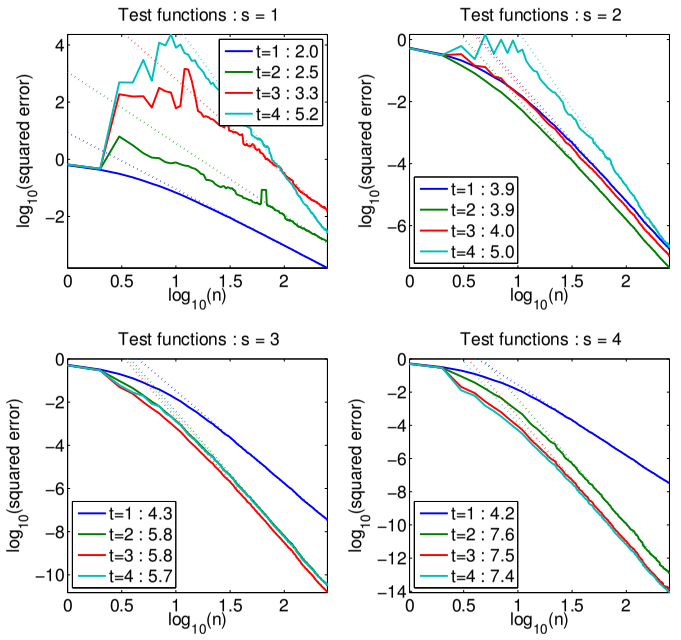


Figure 1: Quadrature for functions in a Sobolev space with parameter s (four possible values) for the uniform distribution on $[0, 1]$, with quadrature rules obtained from different Sobolev spaces with parameters t (same four possible values). We compute affine fits in log-log-space (in dotted) to estimate convergence rates of the form C/n^u and report the value of u. Best seen in color.

- On the contrary, when $s > t$, then we have convergence and the rate is potentially worse than the optimal one (attained for $s = t$), and equal when $t \geqslant s/2$, as shown in Section 5.

In Figure 2, we compare several quadrature rules on $[0, 1]$, namely Simpson's rule with uniformly spread points, Gauss-Legendre quadrature and the Sobol sequence with uniform weights. For $s = 1$, as expected, all squared errors decay as n^{-2} with a worse constant for our kernel-based rule, while for $s = 2$ (smoother test functions), the Sobol sequence is not adaptive, while all others are adaptive and get convergence rates around n^{-4}.

Non-uniform distribution. We consider the case $a = b = 1/2$, which is the distribution $d\rho$ with density $\pi^{-1}x^{-1/2}(1 - x)^{-1/2}$ with respect to the Lebesgue measure, and with cumulative distribution function $F(x) = \pi^{-1}\arccos(1 - 2x)$. We may use an approximation

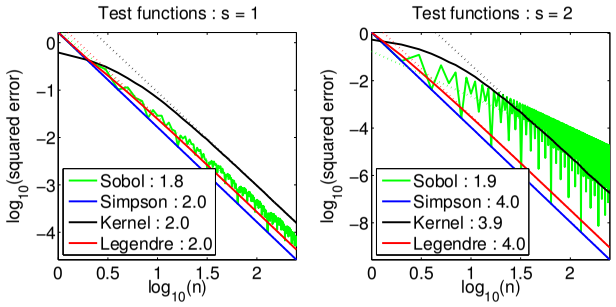


Figure 2: Quadrature for functions in a Sobolev space with parameters $s = 1$ (left) and $s = 2$ (right), for the uniform distribution on $[0, 1]$, with various quadrature rules. We compute affine fits in log-log-space (in dotted) to estimate convergence rates of the form C/n^u and report the value of u. Best seen in color.

of $d\tau$ with N unweighted points $F^{-1}(k/N) = \big(1 - \cos\frac{k\pi}{N}\big)/2$, for $k \in \{1, \dots, N\}$ and the algorithms from the end of Section 4.2. We consider the Sobolev kernel with $s = 1$.

In Figure 3, we plot all densities q_λ^* as a function of λ. When λ is large, we unsuprisingly obtain the uniform density, while, more surprisingly, when λ tends to zero, the density tends to a density, which happens here to be proportional to $x^{1/4}(1-x)^{1/4}$ (leading to a Beta distribution with parameters $a = b = .25$).

We may also consider the same kernel but with the Fourier expansion on $\mathbb{N}$. This is done by representing $d\tau \propto \delta_0 + \sum_{k \in \mathbb{Z}^*} \frac{1}{k^2}\delta_k$ by truncating to all $|k| \leqslant K$, with $K = 50$, which is a weighted representation. We plot in Figure 4 the optimal density over the set of integers, both with respect to the input density (which decays as $1/n^2$) and the counting measure. When λ is large, we recover the input density, while when λ tends to zero, q_λ^* tends to be uniform (and thus, does not converge to a finite measure).

7. Conclusion

In this paper, we have shown that kernel-based quadrature rules are a special case of random feature expansions for positive definite kernels, and derived upper and lower bounds on approximations, that match up to logarithmic terms. For quadrature, this leads to widely applicable results while for random features this allows a significantly improved guarantee within a supervised learning framework.

The present work could be extended in a variety of ways, for example towards bandit optimization rather than quadrature (Srinivas et al., 2012), the use of quasi-random sampling within our framework in the spirit of Yang et al. (2014); Oates and Girolami (2015), a similar analysis for kernel herding (Chen et al., 2010; Bach et al., 2012), an extension to fast rates for non-parametric least-squares regression (Hsu et al., 2014) but with an improved computational complexity, and a study of the consequences of our improved approximation

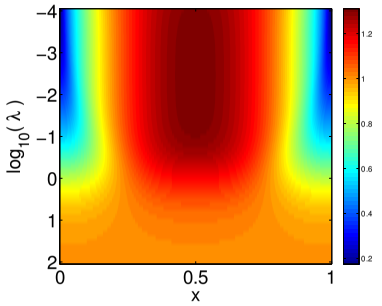


Figure 3: Optimal log-densities $q_\lambda^*(x)$ (with respect to the input distribution) for several values of λ, for the expansion used for quadrature. Best seen in color.

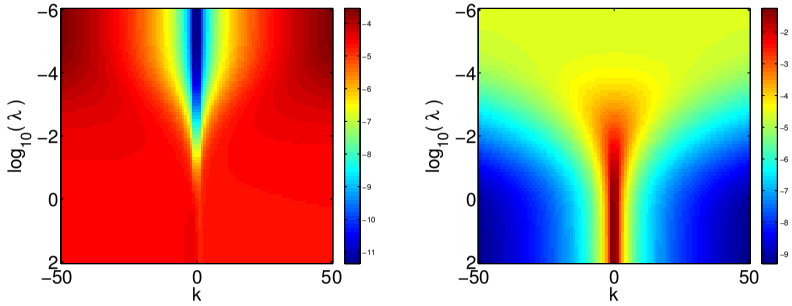


Figure 4: Optimal densities $q_\lambda^*(k)$ for several values of λ, for Fourier feature expansions. Left: with respect to the input distribution (which itself has distribution proportional to $1/k^2$ with respect to the counting measure); right: with respect to the counting measure. Best seen in color.

result for online learning and stochastic approximation, in the spirit of Dai et al. (2014); Dieuleveut and Bach (2015).

Acknowledgements

This work was partially supported by the MSR-Inria Joint Centre and a grant by the European Research Council (SIERRA project 239993). Comments of the reviewers were greatly appreciated and helped improve the presentation significantly. The author would like to thank the STVI for the opportunity of writing a single-handed paper.

Appendix A. Kernels on Product Spaces

In this appendix, we consider sets $\mathcal{X}$ which are products of several simple sets $\mathcal{X}_1, \dots, \mathcal{X}_d$, with known kernels $k_1, \dots, k_d$, each with RKHS $\mathcal{F}_1, \dots, \mathcal{F}_d$. We also assume that we have d measures $d\rho_1, \dots, d\rho_d$, leading to sequences of eigenvalues $(\mu_{jm_j})_{m_j \geqslant 1}$ and eigenfunctions $(e_{jm_j})_{m_j \geqslant 1}$.

Our aim is to define a kernel k on $\mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_d$ with the product measure $d\rho = d\rho_1 \cdots d\rho_d$. For illustration purposes, we consider decays of the form $\mu_m \propto m^{-2s}$ for the d kernels, that will be useful for Sobolev spaces. We also consider the case where $\mu_m \propto \exp(-\rho m)$. For some combinations, eigenvalue decay is the most natural, in others, the number of eigenvalues $m^*(\lambda)$ greater than a given $\lambda > 0$ is more natural.

A.1 Sum of Kernels: $k(x,y) = \sum_{j=1}^d k_j(x_j, y_j)$

In this situation, the RKHS for k is isomorphic to $\mathcal{F}_1 \times \cdots \times \mathcal{F}_d$, composed of functions g such that there exists $f_1, \dots, f_d$ in $\mathcal{F}_1, \dots, \mathcal{F}_d$ such that $g(x) = \sum_{j=1}^d f_j(x_j)$, that is we obtain separable functions, which are sometimes used in the context of generalized additive models (Hastie and Tibshirani, 1990). The corresponding integral operator is then block-diagonal with j-th block equal to the integral operator for k_j and $d\rho_j$. This implies that that its eigenvalues are the concatenation of all sequences $(\mu_{jm_j})_{m_j \geqslant 0}$. Thus the function $m^*(\lambda)$ is the sum of functions $m_1^*(\lambda) + \cdots + m_d^*(\lambda)$.

In terms of norms of functions, we have a norm equal to $\|g\|_{\mathcal{F}}^2 = \sum_{j=1}^d \|f_j\|_{\mathcal{F}_j}^2$.

In the particular case where $\mu_{jm_j} \propto m_j^{-2s}$ for all j, or equivalently, a number of eigenvalues of k_j greater than λ proportional to $\lambda^{-1/(2s)}$, we have a number of eigenvalues of k greater than λ equivalent to $d\lambda^{-1/(2s)}$, that is a decay for the eigenvalues proportional to $(m/d)^{-2s}$. Similarly, when the decay is exponential as $\exp(-\rho m)$, we get a decay of $\exp(-\rho m/d)$.

A.2 Product of Kernels: $k(x,y) = \prod_{j=1}^d k_j(x_j, y_j)$

In this situation, the RKHS for K is exactly the tensor product of $\mathcal{F}_1, \dots, \mathcal{F}_d$, i.e., the span of all functions $\prod_{j=1}^d f_j(x_j)$, for $f_1, \dots, f_d$ in $\mathcal{F}_1, \dots, \mathcal{F}_d$ (Berlinet and Thomas-Agnan, 2004). Moreover, the integral operator for k is a tensor product of the d integral operator for $k_1, \dots, k_d$. This implies that its eigenvalues are $\mu_{1m_1} \times \cdots \times \mu_{dm_d}$, $m_1, \dots, m_d \geqslant 0$. In terms of norms of functions defined on $\mathcal{X}_1 \times \cdots \times \mathcal{X}_d$, this thus corresponds to

$$\sum_{m_1, \dots, m_d \geqslant 0} \Big(\prod_{j=1}^d \mu_{jm_j} \Big)^{-1} \Big\langle f, \prod_{j=1}^d e_{jm_j}(x_j) \Big\rangle_{L_2(d\rho^{\otimes d})}^2 .$$

Special cases. In the particular case where $\mu_{jm_j} \propto m_j^{-2s}$ for all j, we have a number of eigenvalues of k greater than λ equivalent to the number of multi-indices such that $m_1 \times \cdots \times m_d$ is less than $\lambda^{-1/(2s)}$. By counting first the index m_1, this can be upper-bounded by the sum of $\frac{\lambda^{-1/(2s)}}{m_2 \cdots m_d}$ over all indices $m_2, \dots, m_d$ less than $\lambda^{-1/(2s)}$, which is less than $\lambda^{-1/(2s)} \big(\sum_{m=1}^{\lambda^{-1/(2s)}} \frac{1}{m} \big)^{d-1} = O\Big(\lambda^{-1/(2s)} \big(s \log \frac{1}{\lambda} \big)^{d-1} \Big)$. This results in a decay of eigenvalues bounded by $(\log m)^{2s(d-1)} m^{-2s}$ (this can be obtained by inverting approximately the function of λ).

When the decay is exponential as $\exp(-\rho\lambda)$, then we get that $m^*(\lambda)$ is the number of multi-indices $(m_1, \dots, m_d)$ such that their sum is less than $c = \frac{\log \frac{1}{\lambda}}{\rho}$; when c is large, this is equivalent to c^d times the volume of the d-dimensional simplex, and thus less than $\frac{c^d}{d!} = \big(\frac{\log \frac{1}{\lambda}}{\rho} \big)^d \frac{1}{d!}$. This leads to a decay of eigenvalues as $\exp(-\rho d! ^{1/d} m^{1/d})$ or, by using Stirling formula, less than $\exp(-\rho d m^{1/d})$.

Appendix B. Proofs

B.1 Proof of Prop. 1

As shown in Section 2.2, any $f \in \mathcal{F}$ with $\mathcal{F}$-norm less than one may be represented as $f = \int_{\mathcal{V}} g(v) \varphi(v, \cdot) d\tau(v)$, for a certain $g \in L_2(d\tau)$ with $L_2(d\tau)$-norm less than one. We do not solve the problem in β exactly, but use a properly chosen Lagrange multiplier λ and consider the following minimization problem:

$$\Big\| \sum_{i=1}^n \beta_i q(v_i)^{-1/2} \varphi(v_i, \cdot) - \int_{\mathcal{X}} \varphi(v, \cdot) g(v) d\tau(v) \Big\|_{L_2(d\rho)}^2 + n\lambda \|\beta\|_2^2.$$

We consider the operator $\Phi : \mathbb{R}^n \to L_2(d\rho)$ such that

$$\Phi\beta = \sum_{i=1}^n \beta_i q(v_i)^{-1/2} \varphi(v_i, \cdot).$$

We then need to minimize the familiar least-squares problem:

$$\big\| f - \Phi\beta \big\|_{L_2(d\rho)}^2 + n\lambda \|\beta\|_2^2,$$

with solution from the usual normal equations and the matrix inversion lemma for operators (Ogawa, 1988):

$$\beta = (\Phi^*\Phi + n\lambda I)^{-1} \Phi^* f = \frac{1}{n} \Phi^* (\frac{1}{n} \Phi\Phi^* + \lambda I)^{-1} f. \tag{15}$$

We consider the empirical integral operator $\hat{\Sigma} : L_2(d\rho) \to L_2(d\rho)$, defined as

$$\hat{\Sigma} = \frac{1}{n}\Phi\Phi^* = \frac{1}{n}\sum_{i=1}^n \frac{1}{q(v_i)}\varphi(v_i,\cdot) \otimes_{L_2(d\rho)} \varphi(v_i,\cdot),$$

that is, for $a, b \in L_2(d\rho)$, $\langle a, \hat{\Sigma} b\rangle_{L_2(d\rho)} = \sum_{i=1}^n \frac{\langle a, \varphi(v_i,\cdot)\rangle_{L_2(d\rho)}\langle b, \varphi(v_i,\cdot)\rangle_{L_2(d\rho)}}{q(v_i)}$. By construction, and following the end of Section 2.2, we have $\mathbb{E}\hat{\Sigma} = \Sigma$.

The value of $\|f - \Phi\beta\|^2_{L_2(d\rho)}$ is equal to

$$\begin{aligned}\|f - \Phi\beta\|^2_{L_2(d\rho)} &= \|f - \hat{\Sigma}(\hat{\Sigma}+\lambda I)^{-1}f\|^2_{L_2(d\rho)} = \|\lambda(\hat{\Sigma}+\lambda I)^{-1}f\|^2_{L_2(d\rho)} \\ &= \lambda^2\big\langle f, (\hat{\Sigma}+\lambda I)^{-2}f\big\rangle_{L_2(d\rho)} \leqslant \lambda\big\langle f, (\hat{\Sigma}+\lambda I)^{-1}f\big\rangle_{L_2(d\rho)}, \qquad (16)\end{aligned}$$

because $(\hat{\Sigma}+\lambda I)^{-2} \preccurlyeq \lambda^{-1}(\hat{\Sigma}+\lambda I)^{-1}$ (with the classical partial order between self-adjoint operators).

Finally, we have, with $\beta = \frac{1}{n}\Phi^*(\hat{\Sigma}+\lambda I)^{-1}f$:

$$n\|\beta\|_2^2 = \big\langle (\hat{\Sigma}+\lambda I)^{-1}f, \hat{\Sigma}(\hat{\Sigma}+\lambda I)^{-1}f\big\rangle_{L_2(d\rho)} \leqslant \big\langle f, (\hat{\Sigma}+\lambda I)^{-1}f\big\rangle_{L_2(d\rho)}, \qquad (17)$$

using $(\hat{\Sigma}+\lambda I)^{-2}\hat{\Sigma} \preccurlyeq (\hat{\Sigma}+\lambda I)^{-1}$.

By construction, we have $\mathbb{E}(\hat{\Sigma}) = \Sigma$. Moreover, we have, by Cauchy-Schwarz inequality:

$$\begin{aligned}\langle a, (f \otimes_{L_2(d\rho)} f)a\rangle_{L_2(d\rho)} &= \bigg(\int_{\mathcal{X}} a(x)f(x)d\rho(x)\bigg)^2 = \bigg(\int_{\mathcal{X}}\int_{\mathcal{V}} a(x)g(v)\varphi(v,x)d\tau(v)d\rho(x)\bigg)^2 \\ &\leqslant \bigg(\int_{\mathcal{V}} g(v)^2 d\tau(v)\bigg)\int_{\mathcal{V}}\bigg(\int_{\mathcal{X}} a(x)\varphi(v,x)d\rho(x)\bigg)^2 d\tau(v) \\ &= \|g\|^2_{L_2(d\rho)}\langle a, \Sigma a\rangle_{L_2(d\rho)} \leqslant \langle a, \Sigma a\rangle_{L_2(d\rho)}.\end{aligned}$$

Thus $f \otimes_{L_2(d\rho)} f \preccurlyeq \Sigma$, and we may thus define $\langle f, \Sigma^{-1}f\rangle_{L_2(d\rho)}$, which is less than one.

Overall we aim to study $\langle f, (\hat{\Sigma}+\lambda I)^{-1}f\rangle_{L_2(d\rho)}$, for $\langle f, \Sigma^{-1}f\rangle_{L_2(d\rho)} \leqslant 1$, to control both the norm $\|\beta\|_2^2$ in Eq. (17) and the approximation error $\|f - \Phi\beta\|^2_{L_2(d\rho)}$ in Eq. (16). We have, following a similar argument than the one of Bach (2013); El Alaoui and Mahoney (2015) for column sampling, i.e., by a formulation using $\Sigma - \hat{\Sigma}$ in terms of operators in an appropriate way:

$$\begin{aligned}&\langle f, (\hat{\Sigma}+\lambda I)^{-1}f\rangle_{L_2(d\rho)} \\ =\ &\langle f, (\Sigma+\lambda I+\hat{\Sigma}-\Sigma)^{-1}f\rangle_{L_2(d\rho)} \\ =\ &\big\langle (\Sigma+\lambda I)^{-1/2}f, \big[I + (\Sigma+\lambda I)^{-1/2}(\hat{\Sigma}-\Sigma)(\Sigma+\lambda I)^{-1/2}\big]^{-1}(\Sigma+\lambda I)^{-1/2}f\big\rangle_{L_2(d\rho)}.\end{aligned}$$

Thus, if $(\Sigma+\lambda I)^{-1/2}(\hat{\Sigma}-\Sigma)(\Sigma+\lambda I)^{-1/2} \succcurlyeq -tI$, with $t \in (0,1)$, we have

$$\begin{aligned}\langle f, (\hat{\Sigma}+\lambda I)^{-1}f\rangle_{L_2(d\rho)} &\leqslant \langle (\Sigma+\lambda I)^{-1/2}f, (1-t)^{-1}(\Sigma+\lambda I)^{-1/2}f\rangle_{L_2(d\rho)} \\ &= (1-t)^{-1}\langle f, (\Sigma+\lambda I)^{-1}f\rangle_{L_2(d\rho)} \\ &\leqslant (1-t)^{-1}\langle f, \Sigma^{-1}f\rangle_{L_2(d\rho)} \leqslant (1-t)^{-1}.\end{aligned}$$

Moreover, we have shown $(\hat{\Sigma}+\lambda I)^{-1} \preccurlyeq \frac{1}{1-t}(\Sigma+\lambda I)^{-1}$.

Thus, the performance depends on having $(\Sigma+\lambda I)^{-1/2}(\Sigma-\hat{\Sigma})(\Sigma+\lambda I)^{-1/2} \preccurlyeq tI$.

We consider the self-adjoint operators X_i, for $i = 1, \dots, n$, which are independent and identically distributed:

$$X_i = \frac{1}{n}(\Sigma+\lambda I)^{-1}\Sigma - \frac{1}{n}\frac{1}{q(v_i)}\big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big] \otimes_{L_2(d\rho)} \big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big],$$

so that our goal is to provide an upperbound on the probability that $\|\sum_{i=1}^n X_i\|_{\rm op} > t$, where $\|\cdot\|_{\rm op}$ is the operator norm (largest singular values). We use the notation

$$d = \operatorname{tr}\Sigma(\Sigma+\lambda I)^{-1} = \int_{\mathcal{V}} \frac{\langle \varphi(v,\cdot), (\Sigma+\lambda I)^{-1}\varphi(v,\cdot)\rangle_{L_2(d\rho)}}{q(v)} q(v)d\tau(v) \leqslant d_{\max}.$$

We have

$$\begin{aligned}\mathbb{E}X_i &= 0, \text{ by construction of } X_i, \\ X_i &\preccurlyeq \frac{1}{n}(\Sigma+\lambda I)^{-1}\Sigma \preccurlyeq \frac{1}{n}\operatorname{tr}\big[(\Sigma+\lambda I)^{-1}\Sigma\big]I \preccurlyeq \frac{d_{\max}}{n}I, \\ X_i &\succcurlyeq -\frac{1}{n}\frac{1}{q(v_i)}\big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big] \otimes_{L_2(d\rho)} \big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big] \\ &\succcurlyeq -\frac{1}{n}\frac{1}{q(v_i)}\big\|(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big\|^2_{L_2(d\rho)} I \succcurlyeq -\frac{d_{\max}}{n}I, \\ \|X_i\|_{\rm op} &\leqslant \frac{d_{\max}}{n} \text{ as a consequence of the two previous inequalities.}\end{aligned}$$

Moreover,

$$\begin{aligned}&\mathbb{E}(X_i^2) \\ =\ &\mathbb{E}\bigg[\frac{1}{n}\frac{1}{q(v_i)}\big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big] \otimes_{L_2(d\rho)} \big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big]\bigg]^2 - \big[\frac{1}{n}(\Sigma+\lambda I)^{-1}\Sigma\big]^2 \\ \preccurlyeq\ &\mathbb{E}\bigg[\frac{1}{n}\frac{1}{q(v_i)}\big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big] \otimes_{L_2(d\rho)} \big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big]\bigg]^2 \\ =\ &\mathbb{E}\frac{\langle \varphi(v_i,\cdot), (\Sigma+\lambda I)^{-1}\varphi(v_i,\cdot)\rangle_{L_2(d\rho)}}{n^2 q(v_i)^2}\big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big] \otimes_{L_2(d\rho)} \big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big] \\ \preccurlyeq\ &\frac{d_{\max}}{n^2}\mathbb{E}\bigg(\big[\frac{1}{q(v_i)}(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big] \otimes_{L_2(d\rho)} \big[(\Sigma+\lambda I)^{-1/2}\varphi(v_i,\cdot)\big]\bigg) = \frac{d_{\max}}{n^2}\Sigma(\Sigma+\lambda I)^{-1}.\end{aligned}$$

This leads to

$$\sum_{i=1}^n \mathbb{E}(X_i^2) \preccurlyeq \frac{d_{\max}}{n}(\Sigma+\lambda I)^{-1}\Sigma,$$

with a maximal eigenvalue less than $\frac{d_{\max}}{n}$ and a trace less than $\frac{d_{\max}}{n}\operatorname{tr}\Sigma(\Sigma+\lambda I)^{-1} = \frac{d\, d_{\max}}{n}$.

Following Hsu et al. (2014), we use a matrix Bernstein inequality which is independent of the underlying dimension (which is here infinite). We consider the bound of Minsker (2011, Theorem 2.1), which improves on the earlier result of Hsu et al. (2012, Theorem 4), that is:

$$\mathbb{P}\Big(\Big\|\sum_{i=1}^n X_i\Big\|_{\rm op} > t\Big) \leqslant 2d\Big(1+\frac{6}{t^2\log^2(1+nt/d_{\max})}\Big)\exp\Big(-\frac{t^2/2}{d_{\max}/n(1+t/3)}\Big)$$

We now consider $t=\frac{3}{4}$, $\delta\in(0,1)$, and $n\geqslant Bd_{\max}\log\frac{Cd_{\max}}{\delta}$, with appropriate constants $B,C>0$. This implies that

$$\exp\Big(-\frac{t^2/2}{d_{\max}/n(1+t/3)}\Big)\leqslant\exp\Big(-\frac{(3/4)^2/2}{5/4}B\log\frac{Cd_{\max}}{\delta}\Big)\leqslant\big(\frac{\delta}{Cd_{\max}}\big)^{\frac{(3/4)^2B/2}{5/4}},$$

which is less then $\big(\frac{\delta}{Cd}\big)^{\frac{(3/4)^2B/2}{5/4}}$, and, if $d_{\max}\geqslant D$, using $n\geqslant Bd_{\max}\log CD$,

$$1+\frac{6}{t^2\log^2(1+nt/d_{\max})}\leqslant 1+\frac{6\cdot 16/9}{\log^2\big(1+(3B/4)\log(CD)\big)},$$

while if $d_{\max}\leqslant D$ and $n\geqslant 1$,

$$1+\frac{6}{t^2\log^2(1+nt/d_{\max})}\leqslant 1+\frac{6\cdot 16/9}{\log^2\big(1+(3/4D)\big)}.$$

In order to get a bound, we need $\frac{(3/4)^2B/2}{5/4}\geqslant 1$, and we can take $B=5$. If we take $C=8$, then in order to have $1+\frac{6}{t^2\log^2(1+nt/d_{\max})}\leqslant 4$, we can take $D=3/8$. Thus the probability is less than δ.

Finally, in order to get the extra bound on $\frac{1}{n}\sum_{i=1}^n q(v_i)^{-1}\|\varphi(v_i,\cdot)\|^2_{L_2(d\rho)}$, we consider $\mathbb{E}\operatorname{tr}\hat\Sigma=\operatorname{tr}\Sigma=\int_{\mathcal{X}}k(x,x)d\rho(x)$, and thus, by Markov's inequality, with probability $1-\delta$,

$$\frac{1}{n}\sum_{i=1}^n q(v_i)^{-1}\|\varphi(v_i,\cdot)\|^2_{L_2(d\rho)}=\operatorname{tr}\hat\Sigma\leqslant\frac{1}{\delta}\operatorname{tr}\Sigma. \tag{18}$$

By taking $\delta/2$ instead of δ in the control of $\|\sum_{i=1}^n X_i\|_{\rm op}>t$ and in the Markov inequality above, we have a control over $\|\beta\|_2^2$, $\operatorname{tr}\hat\Sigma$ and the approximation error, which leads to the desired result in Prop 1. This will be useful for the lower bound of Prop. 3.

We can make the following extra observations regarding the proof:

- It may be possible to derive a similar result with a thresholding of eigenvalues in the spirit of Zwald et al. (2004), but this would require Bernstein-type concentration inequalities for the projections on principal subspaces.
- We have seen that with high-probability, we have $(\hat\Sigma+\lambda I)^{-1}\preccurlyeq 4(\Sigma+\lambda I)^{-1}$. Note that $A\preccurlyeq B$ does not imply in $A^2\preccurlyeq B^2$ (Bhatia, 2009, page 9) and that in general we do not have $(\hat\Sigma+\lambda I)^{-2}\preccurlyeq C(\Sigma+\lambda I)^{-2}$ for any constant C (which would allow an improvement in the error by replacing λ by λ^2, and violate the lower bound of Prop. 3).
- We may also obtain a result in expectation, by using $\delta=4\lambda/\operatorname{tr}\Sigma$ (which is assumed to be less than 1), leading to a squared error with expectation less than 8λ as soon as $n\geqslant 5d_{\max}(\lambda)\log\frac{2(\operatorname{tr}\Sigma)d_{\max}(\lambda)}{\lambda}$. Indeed, we can use the bound 4λ with probability $1-\delta$ and $\|f\|^2_{L_2(d\rho)}\leqslant\operatorname{tr}\Sigma$ with probability δ, leading to a bound of $4\lambda(1-\delta)+\delta\operatorname{tr}\Sigma\leqslant 8\lambda$. We use this result in Section 4.5.

B.2 Proof of Prop. 2

We start from the bound above, with the constraint $n\geqslant 5d(\lambda)\log\frac{16d(\lambda)}{\delta}$. Statement (a) is a simple reformulation of Prop. 1. For statement (b), if we assume $m\leqslant\frac{n}{5(1+\gamma)\log\frac{16n}{5\delta}}$, and $\lambda=\mu_m$, then we have $d(\lambda)\leqslant(1+\gamma)m$, which implies $n\geqslant 5d(\lambda)\log\frac{16d(\lambda)}{\delta}$, and (b) is a consequence of (a).

B.3 Proof of Prop. 3

We first use the Varshamov-Gilbert's lemma (see, e.g., Massart, 2003, Lemma 4.7). That is, for any integer s, there exists a family $(\theta_j)_{j\in J}$ of at least $|J|\geqslant e^{s/8}$ distinct elements of $\{0,1\}^s$, such that for $j\neq j'\in J$, $\|\theta_j-\theta_{j'}\|_2^2\geqslant\frac{s}{4}$.

For each $\theta\in\{0,1\}^s$, we define an element of $\mathcal{F}$ with norm less than one, as $f(\theta)=\frac{\sqrt{\mu_s}}{\sqrt{s}}\sum_{i=1}^s\theta_i e_i\in\mathcal{F}$, where (e_i,μ_i), $i=1,\dots,s$ are the eigenvector/eigenvalue pairs associated with the s largest eigenvalues of Σ. We have, since $\mu_i\geqslant\mu_s$ for $i\in\{1,\dots,s\}$ and $\|\theta\|_2^2\leqslant 1$:

$$\|f(\theta)\|_{\mathcal{F}}^2=\frac{\mu_s}{s}\sum_{i=1}^s\theta_i^2\mu_i^{-1}\leqslant\frac{\mu_s}{s}\sum_{i=1}^s\theta_i^2\mu_s^{-1}\leqslant\frac{1}{s}\sum_{i=1}^s\theta_i^2\leqslant 1.$$

Moreover, for any $j\neq j'\in J$, we have $\|f(\theta_j)-f(\theta_{j'})\|^2_{L_2(d\rho)}=\frac{\mu_s}{s}\|\theta_j-\theta_{j'}\|_2^2\geqslant\frac{\mu_s}{4}$.

We now assume that s is selected so that $\sqrt{4\lambda}\leqslant\sqrt{\frac{\mu_s}{4}}/3$. By applying the existence results to all functions f_j, $j\in J$, then there exists a family $(\beta_j)_{j\in J}$ of elements of $\mathbb{R}^n$, with squared ℓ_2-norm less than $\frac{4}{n}$, and for which, for all j,

$$\Big\|f_j-\sum_{i=1}^n(\beta_j)_i\psi_i\Big\|_{L_2(d\rho)}\leqslant\sqrt{4\lambda}.$$

This leads to, for any $j\neq j'\in J$,

$$\begin{aligned}\Big\|\sum_{i=1}^n(\beta_j-\beta_{j'})_i\psi_i\Big\|_{L_2(d\rho)}&\geqslant\big\|f_j-f_{j'}\big\|_{L_2(d\rho)}-\Big\|\sum_{i=1}^n(\beta_j)_i\psi_i-f_j\Big\|_{L_2(d\rho)}-\Big\|\sum_{i=1}^n(\beta_{j'})_i\psi_i-f_{j'}\Big\|_{L_2(d\rho)}\\&\geqslant\sqrt{\mu_s/4}-2\sqrt{\frac{\mu_s}{4}}/3=\sqrt{\frac{\mu_s}{4}}/3.\end{aligned}$$

Moreover, we have the bound

$$\Big\|\sum_{i=1}^n(\beta_j-\beta_{j'})_i\psi_i\Big\|^2_{L_2(d\rho)}\leqslant\Big(\sum_{i=1}^n(\beta_j-\beta_{j'})_i^2\Big)\sum_{i=1}^n\|\psi_i\|^2_{L_2(d\rho)}\leqslant\|\beta_j-\beta_{j'}\|_2^2\cdot n(2\delta^{-1}\operatorname{tr}\Sigma).$$

Combining the last two inequalities, we get $\|\beta_j - \beta_{j'}\|_2 \geqslant \sqrt{\frac{\delta\mu_s}{72n\operatorname{tr}\Sigma}} = \Delta$. Thus, $e^{s/8}$ is less than the Δ-packing number of the ball of radius $r = 2/\sqrt{n}$, which is itself less than $(r/\Delta)^n(2+\Delta/r)^n$ (see, e.g., Massart, 2003, Lemma 4.14). Since $\Delta/r = \sqrt{\frac{\delta\mu_s}{4\cdot 72\operatorname{tr}\Sigma}} \leqslant \frac{1}{12\sqrt{2}}$, we have

$$\frac{s}{8} \leqslant n\left(\frac{1}{2}\log\frac{4\cdot 72\operatorname{tr}\Sigma}{\delta\mu_s} + \log(2+\frac{1}{12\sqrt{2}})\right).$$

This implies $n \geqslant \frac{s}{4\log\frac{\operatorname{tr}\Sigma}{\delta\mu_s}+29}$. Given that we have to choose $\mu_s \geqslant 144\lambda$ for the result to hold, this implies the desired result, since $4\log(1440) \geqslant 29$.

References

R. A. Adams and J. F. Fournier. *Sobolev Spaces*, volume 140. Academic Press, 2003.

F. Bach. Sharp analysis of low-rank kernel matrix approximations. In *Proceedings of the International Conference on Learning Theory (COLT)*, 2013.

F. Bach. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research*, 18:1–53, 2017.

F. Bach, S. Lacoste-Julien, and G. Obozinski. On the equivalence between herding and conditional gradient algorithms. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2012.

C. R. Baker. Joint measures and cross-covariance operators. *Transactions of the American Mathematical Society*, 186:273–289, 1973.

P. L. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2003.

A. Berlinet and C. Thomas-Agnan. *Reproducing Kernel Hilbert Spaces in Probability and Statistics*, volume 3. Springer, 2004.

R. Bhatia. *Positive definite matrices*. Princeton University Press, 2009.

M. Sh. Birman and M. Z. Solomyak. Estimates of singular numbers of integral operators. *Russian Mathematical Surveys*, 32(1):15–89, 1977.

A. D. Bull. Convergence rates of efficient global optimization algorithms. *Journal of Machine Learning Research*, 12:2879–2904, 2011.

A. Caponnetto and E. De Vito. Optimal rates for the regularized least-squares algorithm. *Found. Comput. Math.*, 7(3):331–368, 2007.

K. Chaloner and I. Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 10(3):273–304, 1995.

Y. Chen, M. Welling, and A. Smola. Super-samples from kernel herding. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2010.

Y. Cho and L. K. Saul. Kernel methods for deep learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

W. G. Cochran and G. M. Cox. *Experimental designs*. John Wiley & Sons, 1957.

D. Cruz-Uribe and C. J. Neugebauer. Sharp error bounds for the trapezoidal rule and Simpson's rule. *Journal of Inequalities in Pure and Applied Mathematics*, 3(4), 2002.

B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M.-F. Balcan, and L. Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

A. Dieuleveut and F. Bach. Non-parametric stochastic approximation with large step sizes. *The Annals of Statistics*, 44(4):1363–1399, 2015.

A. El Alaoui and M. W. Mahoney. Fast randomized kernel methods with statistical guarantees. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.

E. M. Furrer and D. W. Nychka. A framework to understand the asymptotic properties of kriging and splines. *Journal of the Korean Statistical Society*, 36(1):57–76, 2007.

A. Gelman. *Bayesian Data Analysis*. CRC Press, 2004.

Z. Harchaoui, F. Bach, and E. Moulines. Testing for homogeneity with kernel Fisher discriminant analysis. Technical Report 00270806, HAL, April 2008.

T. J. Hastie and R. J. Tibshirani. *Generalized Additive Models*. Chapman & Hall, 1990.

K. Hesse. A lower bound for the worst-case cubature error on spheres of arbitrary dimension. *Numerische Mathematik*, 103(3):413–433, 2006.

F. B. Hildebrand. *Introduction to Numerical Analysis*. Courier Dover Publications, 1987.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 2012.

D. Hsu, S. M. Kakade, and T. Zhang. Tail inequalities for sums of random matrices that depend on the intrinsic dimension. *Electronic Communications in Probability*, 17(14): 1–13, 2012.

D. Hsu, S. M. Kakade, and T. Zhang. Random design analysis of ridge regression. *Foundations of Computational Mathematics*, 14(3):569–600, 2014.

G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1):489–501, 2006.

F. Huszár and D. Duvenaud. Optimally-weighted herding is Bayesian quadrature. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2012.

T. Kato. *Perturbation theory for linear operators*. Springer Science & Business Media, 1995.

H. König. Eigenvalues of compact operators with applications to integral operators. *Linear Algebra and its Applications*, 84:111–122, 1986.

M. Langberg and L. J. Schulman. Universal epsilon-approximators for integrals. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2010.

Q. Le, T. Sarlós, and A. Smola. Fastfood: approximating kernel expansions in log-linear time. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013.

M. W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2):123–224, 2011.

M. W. Mahoney and P. Drineas. CUR matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009.

P. Massart. *Concentration Inequalities and Model Selection: Ecole d'été de Probabilités de Saint-Flour 23*. Springer, 2003.

S. Mendelson and J. Neeman. Regularization in kernel learning. *The Annals of Statistics*, 38(1):526–565, 2010.

S. Minsker. On some extensions of Bernstein's inequality for self-adjoint operators. Technical Report 1112.5448, arXiv, 2011.

W. J. Morokoff and R. E. Caflisch. Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing*, 15(6):1251–1279, 1994.

R. M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.

E. Novak. *Deterministic and Stochastic Error Bounds in Numerical Analysis*. Springer-Verlag, 1988.

C. J. Oates and M. Girolami. Variance reduction for quasi-Monte-Carlo. Technical Report 1501.03379, arXiv, 2015.

H. Ogawa. An operator pseudo-inversion lemma. *SIAM Journal on Applied Mathematics*, 48(6):1527–1531, 1988.

A. O'Hagan. Bayes-Hermite quadrature. *Journal of statistical planning and inference*, 29 (3):245–260, 1991.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

C. E. Rasmussen and Z. Ghahramani. Bayesian Monte Carlo. In *Advances in Neural Information Processing Systems (NIPS)*, 2003.

C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer New York, 2005.

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

B. Simon. *Trace ideals and their applications*, volume 35. Cambridge University Press, 1979.

S. Smale and F. Cucker. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39(1):1–49, 2001.

A. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In *Algorithmic Learning Theory*, pages 13–31. Springer, 2007.

A. J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2000.

A. J. Smola, Z. L. Ovari, and R. C. Williamson. Regularization with dot-product kernels. *Advances in Neural Information Processing Systems (NIPS)*, 2001.

N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory*, 58(5):3250–3265, 2012.

B. K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. R. G. Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11:1517–1561, 2010.

I. Steinwart, D. R. Hush, and C. Scovel. Optimal rates for regularized least squares regression. In *Proceedings of the International Conference on Learning Theory (COLT)*, 2009.

G. Wahba. *Spline Models for observational data*. SIAM, 1990.

H. Widom. Asymptotic behavior of the eigenvalues of certain integral equations I. *Transactions of the American Mathematical Society*, 109:278–295, 1963.

C. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

J. Yang, V. Sindhwani, H. Avron, and M. Mahoney. Quasi-Monte Carlo feature maps for shift-invariant kernels. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2014.

L. Zwald, G. Blanchard, P. Massart, and R. Vert. Kernel projection machine: a new tool for pattern recognition. In *Advances in Neural Information Processing Systems (NIPS)*, 2004.

Journal of Machine Learning Research 18 (2017) 1-40 Submitted 9/15; Revised 11/16; Published 4/17

Analyzing Tensor Power Method Dynamics in Overcomplete Regime

Animashree Anandkumar A.ANANDKUMAR@UCI.EDU
Department of Electrical Engineering and Computer Science
University of California, Irvine
Engineering Hall, Room 4408
Irvine, CA 92697, USA

Rong Ge RONGGE@CS.DUKE.EDU
Department of Computer Science
Duke University
308 Research Drive (LSRC Building), Room D226
Durham, NC 27708, USA

Majid Janzamin MJANZAMI@UCI.EDU
Department of Electrical Engineering and Computer Science
University of California, Irvine
Engineering Hall, Room 4406
Irvine, CA 92697, USA

Editor: Tommi Jaakkola

Abstract

We present a novel analysis of the dynamics of tensor power iterations in the overcomplete regime where the tensor CP rank is larger than the input dimension. Finding the CP decomposition of an overcomplete tensor is NP-hard in general. We consider the case where the tensor components are randomly drawn, and show that the simple power iteration recovers the components with bounded error under mild initialization conditions. We apply our analysis to unsupervised learning of latent variable models, such as multi-view mixture models and spherical Gaussian mixtures. Given the third order moment tensor, we learn the parameters using tensor power iterations. We prove it can correctly learn the model parameters when the number of hidden components k is much larger than the data dimension d, up to $k = o(d^{1.5})$. We initialize the power iterations with data samples and prove its success under mild conditions on the signal-to-noise ratio of the samples. Our analysis significantly expands the class of latent variable models where spectral methods are applicable. Our analysis also deals with noise in the input tensor leading to sample complexity result in the application to learning latent variable models.

Keywords: tensor decomposition, tensor power iteration, overcomplete representation, unsupervised learning, latent variable models

©2017 Animashree Anandkumar, Rong Ge, and Majid Janzamin.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-486.html.

1. Introduction

CANDECOMP/PARAFAC (CP) decomposition of a symmetric tensor $T \in \mathbb{R}^{d\times d\times d}$ is the process of decomposing it into a succinct sum of rank-one tensors, given by

$$T = \sum_{j\in[k]} \lambda_j a_j \otimes a_j \otimes a_j, \quad \lambda_j \in \mathbb{R},\ a_j \in \mathbb{R}^d, \tag{1}$$

where $\otimes$ denotes the outer product. The minimum k for which the tensor can be decomposed in the above form is called the (symmetric) tensor rank. *Tensor power iteration* is a simple, popular and efficient method for recovering the tensor rank-one components a_j's. The tensor power iteration is given by

$$x \leftarrow \frac{T(I,x,x)}{\|T(I,x,x)\|}, \tag{2}$$

where

$$T(I,x,x) := \sum_{j,l\in[d]} x_j x_l T(:,j,l) \in \mathbb{R}^d$$

is a *multilinear* combination of tensor *fibers*, and $\|\cdot\|$ is the ℓ_2 norm operator. See Section 1.3 for an overview of tensor notations and preliminaries.

The tensor power iteration is a generalization of matrix power iteration: for matrix $M \in \mathbb{R}^{d\times d}$, the power iteration is given by $x \leftarrow Mx/\|Mx\|$. Dynamics and convergence properties of matrix power iterations are well understood (Horn and Johnson, 2012). On the other hand, a theoretical understanding of tensor power iterations is much more limited. Tensor power iteration can be viewed as a *gradient descent* step (with infinite step size), corresponding to the problem of finding the best rank-1 approximation of the input tensor T (Anandkumar et al., 2014c). This optimization problem is non-convex. Unlike the matrix case, where the number of isolated stationary points of power iteration is at most the dimension (given by eigenvectors corresponding to unique eigenvalues), in the tensor case, the number of stationary points is, in fact, exponential in the input dimension (Cartwright and Sturmfels, 2013). This makes the analysis of tensor power iteration far more challenging.

Despite the above challenges, many advances have been made in understanding the tensor power iterations in specific regimes. When the components a_j's are orthogonal to one another, it is known that there are no spurious local optima for tensor power iterations, and the only stable fixed points correspond to the true a_j's (Zhang and Golub, 2001; Anandkumar et al., 2014c). Any tensor with linearly independent components a_j's can be orthogonalized, via an invertible transformation (whitening) and thus, its components can be recovered efficiently. A careful perturbation analysis in this setting was carried out in Anandkumar et al. (2014c).

The framework in Anandkumar et al. (2014c) is however not applicable in the overcomplete setting, where the tensor rank k exceeds the dimension d. Such overcomplete tensors cannot be orthogonalized and finding guaranteed decomposition is a challenging open problem. It is known that finding CP tensor decomposition is NP-hard (Hillar and Lim, 2013). In this paper, we make significant headway in showing that the simple power iterations can recover the components in the overcomplete regime under a set of mild conditions on the components a_j's.

Overcomplete tensors also arise in many machine learning applications such as moments of many latent variable models, e.g., multiview mixtures, independent component Analysis (ICA), and sparse coding models, where the number of hidden variables exceeds the input dimensions (Anandkumar et al., 2015). Overcomplete models often have impressive empirical performance (Coates et al., 2011), and can provide greater flexibility in modeling, and are more robust to noise (Lewicki and Sejnowski, 2000). By studying algorithms for overcomplete tensor decomposition, we expand the class of models that can be learnt efficiently using simple spectral methods such as tensor power iterations. Note there are other algorithms for decomposing overcomplete tensors (De Lathauwer et al., 2007; Goyal et al., 2013; Bhaskara et al., 2013), but they all require tensors of at least 4-th order and require large computational complexity. Ge and Ma (2015) works for 3rd order tensor but requires quasi-polynomial time. The main contribution of this paper is an analysis for the practical power method in the overcomplete regime.

1.1 Summary of Results

We analyze the dynamics of third order tensor power iterations in the overcomplete regime. We assume that the tensor components a_j's are randomly drawn from the unit sphere. Since general tensor decomposition is challenging in the overcomplete regime, we argue that this is a natural first step to consider for tractable recovery.

We characterize the basin of attraction for the local optima near the rank-one components a_j's. We show that under mild initialization condition, there is fast convergence to these local optima in $O(\log\log d)$ iterations. This result is the core technical analysis of this paper stated in the following theorem.

Theorem 1 (Dynamics of tensor power iteration) *Consider tensor $\hat{T} = T + E$ such that exact tensor T has rank-k decomposition in (1) with rank-one components $a_j \in \mathbb{R}^d, j \in [k]$ being uniformly i.i.d. drawn from the unit d-dimensional sphere, and the ratio of maximum and minimum (in absolute value) weights λ_j's being constant. In addition, suppose the perturbation tensor E has bounded spectral norm as*

$$\|E\| \leq \epsilon \frac{\sqrt{k}}{d}, \quad \text{where} \quad \epsilon < o\left(\frac{\sqrt{k}}{d}\right). \tag{3}$$

Let tensor rank $k = o(d^{1.5})$, and the unit-norm initial vector $x^{(1)}$ satisfy the correlation bound

$$|\langle x^{(1)}, a_j\rangle| \geq d^{\beta}\frac{\sqrt{k}}{d}, \tag{4}$$

w.r.t. some true component $a_j, j \in [k]$, for some $\beta > (\log d)^{-c}$ for some universal constant $c > 0$. After $N = \Theta(\log\log d)$ iterations, the tensor power iteration in (2) outputs a vector having w.h.p. a constant correlation with the true component a_j as $|\langle x^{(N+1)}, a_j\rangle| \geq 1 - \gamma$, for any fixed constant $\gamma > 0$.

As a corollary, this result can be used for learning latent variable models such as multiview mixtures. We show that the above initialization condition is satisfied using a sample with mild signal-to-noise ratio; see Section 2 for more details on this.

The above result is a significant improvement over the recent analysis by Anandkumar et al. (2015, 2014a,b) for overcomplete tensor decomposition. In these works, it is required for the initialization vectors to have a constant amount of correlation with the true a_j's. However, obtaining such strong initializations is usually not realistic in practice. On the other hand, the initialization condition in (4) is mild, and decaying even when the rank k is significantly larger than dimension d; up to $k = o(d^{1.5})$. In learning the mixture model, such initialization vectors can be obtained as samples from the mixture model, even when there is a large amount of noise. Given this improvement, we combine our analysis in Theorem 1, and the guarantees in Anandkumar et al. (2015, 2014a), proving that the model parameters can be recovered consistently.

A detailed proof outline for Theorem 1 is provided in Section 3.1. Under the random assumption, it is not hard to show that the first iteration of tensor power update makes progress. However, after the first iteration, the input vector and the tensor components are no longer *independent* of each other. Therefore, we cannot directly repeat the same argument for the second step.

How do we analyze the second step even though the vector and tensor components are correlated? The main intuition is to characterize the dependency between the vector and the tensor components, and show that there is still enough randomness left for us to repeat the argument. This idea was inspired by the analysis of Approximate Message Passing (AMP) algorithms (Bayati and Montanari, 2010). However, our analysis here is very different in several key aspects: 1) In approximate message passing, typically the analysis works in the *large system limit*, where the number of iterations is fixed and the dimension goes to infinity. Here we can handle a superconstant number of iterations $O(\log\log d)$, even for finite d; 2) Usually k is assumed to be a constant factor times d in the AMP-like analysis, while here we allow them to be polynomially related.

1.2 Related Work

Tensor decomposition for learning latent variable models: In the introduction, some related works are mentioned which study the theoretical and practical aspects of spectral techniques for learning latent variable models. Among them, Anandkumar et al. (2014c) provide the analysis of tensor power iteration for learning several latent variable models in the undercomplete regime. Anandkumar et al. (2014a) provide the analysis in the overcomplete regime and Anandkumar et al. (2014b) provide tensor concentration bounds and apply the analysis in (Anandkumar et al., 2014a) to learning LVMs proposing tight sample complexity guarantees.

Learning mixture of Gaussians: Here, we provide a subset of related works studying learning mixture of Gaussians which are more comparable with our result. For a more detailed list of these works, see Anandkumar et al. (2014c); Hsu and Kakade (2013). The problem of learning mixture of Gaussians dates back to the work by Pearson (1895). They propose a moment-based technique that involves solving systems of multivariate polynomials which is in general challenging in both computational and statistical sense. Recently, lots of studies on learning Gaussian mixture models have been done improving both aspects which can be divided to two main classes: distance-based and spectral methods.

Distance-based methods impose separation condition on the mean vectors showing that under enough separation the parameters can be estimated. Among such approaches, we can mention Dasgupta (1999); Vempala and Wang (2002); Arora and Kannan (2005). As discussed in the summary of results, these results work even if $k > d^{1.5}$ as long as the separation condition between means is satisfied, but our work can tolerate higher level of noise in the regime of $k = o(d^{1.5})$ with polynomial computational complexity. The guarantees in (Vempala and Wang, 2002) also work in the high noise regime but need higher computational complexity as polynomial in $k^{O(k)}$ and d.

In the spectral approaches, the observed moments are constructed and the spectral decomposition of the observed moments are performed to recover the parameters (Kalai et al., 2010; Anandkumar et al., 2012, 2014b). Kalai et al. (2010) analyze the problem of learning mixture of two general Gaussians and provide algorithm with high order polynomial sample and computational complexity. Note that in general, the complexity of such methods grows exponentially with the number of components without further assumptions (Moitra and Valiant, 2010). Hsu and Kakade (2013) provide a spectral algorithm under non-degeneracy conditions on the mean vectors and propose guarantees with polynomial sample complexity depending on the condition number of the moment matrices. Anandkumar et al. (2014b) perform tensor power iteration on the third order moment tensor to recover the mean vectors in the overcomplete regime as long as $k = o(d^{1.5})$, but need very good initialization vector having constant correlation with the true mean vector. Here, we improve the correlation level required for convergence.

1.3 Notation and Tensor Preliminaries

Let $[k] := \{1, 2, \ldots, k\}$, and $\|v\|$ denote the ℓ_2 norm of vector v. We use $\tilde{O}$ and $\tilde{\Omega}$ to hide polylog factors in asymptotic notations O and Ω, respectively.

Tensor preliminaries: A real *p-th order tensor* $T \in \bigotimes^p \mathbb{R}^d$ is a member of the outer product of Euclidean spaces $\mathbb{R}^d$. The different dimensions of the tensor are referred to as *modes*. For instance, for a matrix, the first mode refers to columns and the second mode refers to rows. In addition, *fibers* are higher order analogues of matrix rows and columns. A fiber is obtained by fixing all but one of the indices of the tensor (and is arranged as a column vector). For example, for a third order tensor $T \in \mathbb{R}^{d\times d\times d}$, the mode-1 fiber is given by $T(:, j, l)$. Similarly, *slices* are obtained by fixing all but two of the indices of the tensor. For example, for the third order tensor T, the slices along 3rd mode are given by $T(:, :, l)$.

We view a tensor $T \in \mathbb{R}^{d\times d\times d}$ as a multilinear form. In particular, for vectors $u, v, w \in \mathbb{R}^d$, we have [1]

$$T(I, v, w) := \sum_{j,l\in[d]} v_j w_l T(:, j, l) \ \in \mathbb{R}^d, \tag{5}$$

which is a multilinear combination of the tensor mode-1 fibers. Similarly $T(u, v, w) \in \mathbb{R}$ is a multilinear combination of the tensor entries, and $T(I, I, w) \in \mathbb{R}^{d\times d}$ is a linear combination of the tensor slices.

A 3rd order tensor $T \in \mathbb{R}^{d\times d\times d}$ is said to be rank-1 if it can be written in the form

$$T = \lambda \cdot a \otimes b \otimes c \Leftrightarrow T(i, j, l) = \lambda \cdot a(i) \cdot b(j) \cdot c(l), \tag{6}$$

1. Compare with the matrix case where for $M \in \mathbb{R}^{d\times d}$, we have $M(I, u) = Mu := \sum_{j\in[d]} u_j M(:, j)$.

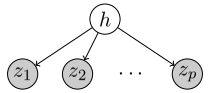


Figure 1: Multiview mixture model.

where notation $\otimes$ represents the *outer product* and $a, b, c \in \mathbb{R}^d$ are unit vectors. A tensor $T \in \mathbb{R}^{d\times d\times d}$ is said to have a CP *rank* at most k if it can be written as the sum of k rank-1 tensors as

$$T = \sum_{i\in[k]} \lambda_i a_i \otimes b_i \otimes c_i, \quad \lambda_i \in \mathbb{R}, \ a_i, b_i, c_i \in \mathbb{R}^d. \tag{7}$$

For third order tensor $T \in \mathbb{R}^{d\times d\times d}$, the spectral (operator) norm is defined as

$$\|T\| := \sup_{\|u\|=\|v\|=\|w\|=1} |T(u, v, w)|.$$

In the rest of the paper, Section 2 describes how to apply our tensor results to learning multiview mixture models. Section 3 illustrates the proof ideas, with more details in the Appendix. Finally we conclude in Section 4.

2. Learning Multiview Mixture Model through Tensor Methods

We proposed our main technical result in Section 1.1 providing convergence guarantees for the tensor power iterations given mild initialization conditions in the overcomplete regime; see Theorem 1. Along this result we provide the application to learning multiview mixtures model in Theorem 2. In this section, we briefly introduce the tensor decomposition framework as the learning algorithm and then state the learning guarantees with more details and remarks.

2.1 Multiview Mixture Model

Consider an exchangeable multiview mixture model with k components and $p \geq 3$ views; see Figure 1. Suppose that hidden variable h is a discrete categorical random variable taking one of the k states. It is convenient to represent it by basis vectors such that

$$h = e_j \in \mathbb{R}^k \quad \text{if and only if} \quad \text{it takes the } j\text{-th state.}$$

Note that $e_j \in \mathbb{R}^k$ denotes the j-the basis vector in the k-dimensional space. The prior probability for each hidden state is $\Pr[h = e_j] = \lambda_j, j \in [k]$. For simplicity, in this paper we assume all the λ_i's are the same. However, similar argument works even when the ratio of maximum and minimum prior probabilities $\lambda_{\max}/\lambda_{\min}$ is bounded by some constant.

The variables (views) $z_l \in \mathbb{R}^d$ are related to the hidden state through *factor matrix* $A \in \mathbb{R}^{d\times k}$ such that

$$z_l = Ah + \eta_l, \quad l \in [p],$$

Algorithm 1 Learning multiview mixture model via tensor power iterations

Require: 1) Third order moment tensor $T \in \mathbb{R}^{d\times d\times d}$ in (8), 2) n samples of z_1 in multiview mixture model as $z_1^{(\tau)}, \tau \in [n]$, and 3) number of iterations N.

1: **for** $\tau = 1$ **to** n **do**
2: **Initialize** unit vectors $x_\tau^{(1)} \leftarrow z_1^{(\tau)} / \left\| z_1^{(\tau)} \right\|$.
3: **for** $t = 1$ **to** N **do**
4: Tensor power updates (see (5) for the definition of the multilinear form):

$$x_\tau^{(t+1)} = \frac{T\left(I, x_\tau^{(t)}, x_\tau^{(t)}\right)}{\left\| T\left(I, x_\tau^{(t)}, x_\tau^{(t)}\right)\right\|}, \tag{9}$$

5: **end for**
6: **end for**
7: **return** the output of Procedure 2 with input $\left\{x_\tau^{(N+1)} : \tau \in [n]\right\}$ as estimates x_j.

where zero-mean noise vectors $\eta_l \in \mathbb{R}^d$ are independent of each other and the hidden state h. Given this, the variables (views) $z_l \in \mathbb{R}^d$ are conditionally independent given the latent variable h, and the conditional means are $\mathbb{E}[z_l|h = e_j] = a_j$, where $a_j \in \mathbb{R}^d$ denotes the j-th column of factor matrix $A = [a_1 \cdots a_k] \in \mathbb{R}^{d\times k}$. In addition, the above properties imply that the order of observations z_l do not matter and the model is *exchangeable*. The goal of the learning problem is to recover the parameters of the model (factor matrix) A given observations.

For this model, the third order[2] observed moment has the form (Anandkumar et al., 2014c)

$$\mathbb{E}[z_1 \otimes z_2 \otimes z_3] = \sum_{j\in[k]} \lambda_j a_j \otimes a_j \otimes a_j. \tag{8}$$

Hence, given third order observed moment, the unsupervised learning problem (recovering factor matrix A) reduces to computing a tensor decomposition as in (8).

2.2 Tensor Decomposition Algorithm

The algorithm for unsupervised learning of multiview mixture model is based on tensor decomposition techniques provided in Algorithm 1. The main step in (9) performs *tensor power iteration*;[3] see (5) for the multilinear form definition. After running the algorithm for all different initialization vectors, the clustering process from Anandkumar et al. (2015) ensures that the best converged vectors are returned as the estimation of true components a_j.

2. It is enough to form the third order moment for our learning purpose.
3. This is the generalization of matrix power iteration to 3rd order tensors.

Procedure 2 Clustering process (Anandkumar et al., 2015)

Require: Tensor $T \in \mathbb{R}^{d\times d\times d}$, set $S := \left\{x_\tau^{(N+1)} : \tau \in [n]\right\}$, parameter ν.

1: **while** S is not empty **do**
2: Choose $x \in S$ which maximizes $|T(x, x, x)|$.
3: Do N more iterations of (9) starting from x.
4: **Output** the result of iterations denoted by $\hat{x}$.
5: Remove all the $x \in S$ with $|\langle x, \hat{x}\rangle| > \nu/2$.
6: **end while**

2.3 Learning Guarantees

We assume a Gaussian prior on the mean vectors, i.e., the vectors $a_j \sim \mathcal{N}(0, I_d/d)$, $j \in [k]$ are i.i.d. drawn from a standard multivariate Gaussian distribution with unit expected square norm. Note that in the high dimension (growing d), this assumption is the same as uniformly drawing from unit sphere since the norm of vector concentrates in the high dimension and there is no need for normalization. Even though we impose a prior distribution, we do not use a MAP estimator, since the corresponding optimization is NP-hard. Instead, we learn the model parameters through decomposition of the third order moments through tensor power iterations. The assumption of a Gaussian prior is standard in machine learning applications. We impose it here for tractable analysis of power iteration dynamics. Such Gaussian assumptions have been used before for analysis of other iterative methods such as approximate message passing algorithms, and there are evidences that similar results hold for more general distributions; see (Bayati and Montanari, 2010) and references there.

As explained in the previous sections, we use tensor power method to learn the components a_j's, and the method is initialized with observed samples z_i. Intuitively, this initialization is useful since $z_i = Ah + \eta_i$ is a perturbed version of desired parameter a_j (when $h = e_j$). Thus, we present the result in terms of the signal-to-noise (SNR) ratio which is the expected norm of signal a_j (which is one here) divided by the expected norm of noise η_i, i.e., the SNR in the i-th sample $z_i = a_j + \eta_i$ (assumed $h = e_j$) is defined as $\text{SNR} := \mathbb{E}[\|a_j\|]/\mathbb{E}[\|\eta_i\|]$. This specifies how much noise the initialization vector z_i can tolerate in order to ensure the convergence of tensor power iteration to a desired local optimum. We now propose the conditions required for recovery guarantees, and state a brief explanation of them.

Conditions for Theorems 2 and 3:

- Rank condition: $k \le o(d^{1.5})$.
- The columns of A are uniformly i.i.d. drawn from unit d-dimensional sphere.
- The noise vectors $\eta_l, l \in [3]$, are independent of matrix A and each other. In addition, the signal-to-noise ratio (SNR) is w.h.p. bounded as

$$\text{SNR} \ge \Omega\left(\frac{\sqrt{\max\{k, d\}}}{d^{1-\beta}}\right),$$

for some $\beta \ge (\log d)^{-c}$ for universal constant $c > 0$.

The rank condition bounds the level of overcompleteness for which the recovery guarantees are satisfied. The random assumption on the columns of A are crucial for analyzing the dynamics of tensor power iteration. We use it to argue there exists enough randomness left in the components after conditioning on the previous iterations; see Section 3.1 for the details. The bound on the SNR is required to make sure the given sample used for initialization is close enough to the corresponding mean vector. This ensures that the initial vector is inside the basin-of-attraction of the corresponding component, and hence, the convergence to the mean vector can be guaranteed. Under these assumptions we have the following theorem.

Theorem 2 (Learning multiview mixture model: closeness to single columuns) *Consider a multiview mixture model (or a spherical Gaussian mixture) in the above settings with k components in d dimensions. If the above conditions hold, then the tensor power iteration converges to a vector close to one of the true mean vectors a_j's (having constant correlation).*

In particular, for mildly overcomplete models, where $k = \alpha d$ for some constant $\alpha > 1$, the signal-to-noise ratio (SNR) is as low as $\Omega(d^{-1/2+\epsilon})$, for any $\epsilon > 0$. Thus, we can learn mixture models with a high level of noise. In general, we establish how the required noise level scales with the number of hidden components k, as long as $k = o(d^{1.5})$.

The above theorem states convergence to desired local optima which are close to true components a_j's. In Theorem 3, we show that we can sharpen the above result, by jointly iterating over the recovered vectors, and consistently recover the components a_j's. This result also uses the analysis from Anandkumar et al. (2015).

Theorem 3 (Learning multiview mixture model: recovering the factor matrix) *Assume the above conditions hold. The initialization of power iteration is performed by samples of z_1 in multiview mixture model. Suppose the tensor power iterations are at least initialized once for each $a_j, j \in [k]$ such that $z_1 = a_j + \eta_1$.[4] Then by using the exact 3rd order moment tensor in (8) as input, the tensor decomposition algorithm outputs an estimate $\hat{A}$ (up to permutation of its columns) satisfying w.h.p. (over the randomness of the components a_j's)*

$$\left\| \hat{A} - A \right\|_F \leq \epsilon,$$

where the number of iterations of the algorithm is $N = \Theta\left(\log\left(\frac{1}{\epsilon}\right) + \log\log d\right)$.

See Section 3 for the proof.

The above theorems assume the exact third order tensor is given to the algorithm. We provide the results given empirical tensor in Section 2.3.1.

Learning spherical Gaussian mixtures: Consider a mixture of k different Gaussian vectors with spherical covariance. Let $a_j \in \mathbb{R}^d, j \in [k]$ denote the mean vectors and the covariance matrices are $\sigma^2 I$. Assuming the parameter σ is known, the modified third order observed moment

$$M_3 := \mathbb{E}[z \otimes z \otimes z] - \sigma^2 \sum_{i \in [d]} \left(\mathbb{E}[z] \otimes e_i \otimes e_i + e_i \otimes \mathbb{E}[z] \otimes e_i + e_i \otimes e_i \otimes \mathbb{E}[z] \right)$$

4. Note that this happens for component j with high probability when the number of initializations is proportional to inverse prior probability corresponding to that mixture.

has the tensor decomposition form (Hsu and Kakade, 2012)

$$M_3 = \sum_{j \in [k]} \lambda_j a_j \otimes a_j \otimes a_j,$$

where λ_j is the probability of drawing j-th Gaussian mixture. The above guarantees can be applied to learning mean vectors a_j in this model with the additional property that the noise is spherical Gaussian.

Learning multiview mixture model with distinct factor matrices: Consider the multiview mixture model with different factor matrices where the first three views are related to the hidden state as

$$z_1 = Ah + \eta_1, \quad z_2 = Bh + \eta_2, \quad z_3 = Ch + \eta_3.$$

Then, the guarantees in the above theorem can be extended to recovering the columns of all three factor matrices A, B, and C with appropriate modifications in the power iteration algorithm as follows. First the update formula (9) is changed as

$$x_{1,\tau}^{(t+1)} = \frac{T\left(I, x_{2,\tau}^{(t)}, x_{3,\tau}^{(t)}\right)}{\left\| T\left(I, x_{2,\tau}^{(t)}, x_{3,\tau}^{(t)}\right) \right\|}, \quad x_{2,\tau}^{(t+1)} = \frac{T\left(x_{1,\tau}^{(t)}, I, x_{3,\tau}^{(t)}\right)}{\left\| T\left(x_{1,\tau}^{(t)}, I, x_{3,\tau}^{(t)}\right) \right\|}, \quad x_{3,\tau}^{(t+1)} = \frac{T\left(x_{1,\tau}^{(t)}, x_{2,\tau}^{(t)}, I\right)}{\left\| T\left(x_{1,\tau}^{(t)}, x_{2,\tau}^{(t)}, I\right) \right\|},$$

which is the alternating asymmetric version of symmetric power iteration in (9). Here, we alternate among different modes of the tensor. In addition, the initialization for each mode of the tensor is appropriately performed with the samples corresponding to that mode. Note that the analysis still works in the asymmetric version since there exists even more independence relationships through the iterations of the power update because of introducing new random matrices B and C.

2.3.1 Sample Complexity Analysis

In the previous section, we assumed the exact third order tensor in (8) is given to the tensor decomposition Algorithm 1. We now estimate the tensor given n samples $z_1^{(i)}, z_2^{(i)}, z_3^{(i)}, i \in [n]$, as

$$\hat{T} = \frac{1}{n} \sum_{i \in [n]} z_1^{(i)} \otimes z_2^{(i)} \otimes z_3^{(i)}. \tag{10}$$

For the multiview mixture model introduced in Section 2.1, let the noise vector η_l be spherical, and ζ^2 denote the variance of each entry of noise vector. We now provide the following recovery guarantees.

Additional conditions for Theorem 4:

- Let $E_1 := [\eta_1^{(1)}, \eta_1^{(2)}, \ldots, \eta_1^{(n)}] \in \mathbb{R}^{d \times n}$, where $\eta_1^{(i)} \in \mathbb{R}^d$ is the i-th sample of noise vector η_1. These noise matrices satisfy the following *RIP property* which is adapted from Candes and Tao (2006). Matrix $E_1 \in \mathbb{R}^{d \times n}$ satisfies a weak RIP condition such that for any subset of $O\left(\frac{d}{\log^2 d}\right)$ number of columns, the spectral norm of E_1 restricted to those columns is bounded by 2. The same condition is satisfied for similarly defined noise matrices E_2 and E_3.

- The number of samples n satisfies lower bound such that

$$\zeta\left(\frac{\sqrt{d}}{n}+\sqrt{\lambda_{\max}\frac{d}{n}}\right)+\zeta^2\left(\frac{d}{n}+\sqrt{\lambda_{\max}\frac{d^{1.5}}{n}}\right)+\zeta^3\left(\frac{d^{1.5}}{n}+\sqrt{\frac{d}{n}}\right)\leq\min\left\{\epsilon\frac{\sqrt{k}}{d},\tilde{O}(\lambda_{\min})\right\},\qquad(11)$$

where $\epsilon<o\left(\sqrt{k}/d\right)$.

Theorem 4 (Learning multiview mixture model) *Consider the empirical tensor in* (10) *as the input to tensor decomposition Algorithm 1. Suppose the above additional conditions are also satisfied. Then, the same guarantees as in Theorem 2 hold. In addition, the same guarantees as in Theorem 3 also hold with the recovery bound (up to permutation of columns of $\hat{A}$) changed as*

$$\left\|\hat{A}-A\right\|_F\leq\tilde{O}\left(\frac{\sqrt{k}\cdot\|E\|}{\lambda_{\min}}\right),$$

where E denotes the perturbation tensor originated from empirical estimation in (10)*, and its spectral norm $\|E\|$ is bounded by the LHS of* (11)*.*

See Section 3 for the proof.

3. Proof Outline

Our main technical result is the analysis of third order tensor power iteration provided in Theorem 1 which also allows to tolerate some amount of noise in the input tensor. We analyze the noiseless and noisy settings in different ways. We basically first prove the result for the noiseless setting where the input tensor has an exact rank-k decomposition in (1). When the noise is also considered, we show that the contribution of noise in the analysis is dominated by the main signal, and thus, the same result still holds. For the rest of this section we focus on the noiseless setting, while we discuss the proof ideas for the noisy case in Section 3.2.

We first discuss the proof of Theorem 3 which involves two phases. In the first phase, we show that under certain small amount of correlation (see (13)) between the initial vector and the true component, the power iteration in (2) converges to some vector which has constant correlation with the true component. This result is the core technical analysis of this paper which is provided in Lemma 5. In the second phase, we incorporate the result of Anandkumar et al. (2015, 2014a) which guarantees the convergence of power iteration (followed by a coordinate descent iteration) given initial vectors having constant correlation with the true components. This is stated in Lemma 6.

To simplify the notation, we consider the tensor[5]

$$T=\sum_{j\in[k]}a_j\otimes a_j\otimes a_j,\quad a_j\sim\mathcal{N}(0,\frac{1}{d}I_d).\qquad(12)$$

5. In the analysis, we assume that all the weights are equal to one which can be generalized to the case when the ratio of maximum and minimum weights (in absolute value) are constant.

Notice that this is exactly proportional to the 3rd order moment tensor of the multiview mixture model in (8).

The following lemma is restatement of Theorem 1 in the noiseless setting.

Lemma 5 (Dynamics of tensor power iteration, phase 1) *Consider the rank-k tensor T of the form in* (12)*. Let tensor rank $k=o(d^{1.5})$, and the unit-norm initial vector $x^{(1)}$ satisfies the correlation bound*

$$|\langle x^{(1)},a_j\rangle|\geq d^{\beta}\frac{\sqrt{k}}{d},\qquad(13)$$

w.r.t. some true component $a_j, j\in[k]$, for some $\beta>(\log d)^{-c}$ for some universal constant $c>0$. After $N=\Theta(\log\log d)$ iterations, the tensor power iteration in (2) *outputs a vector having w.h.p. a constant correlation with the true component a_j as*

$$|\langle x^{(N+1)},a_j\rangle|\geq 1-\gamma,$$

for any fixed constant $\gamma>0$.

The proof outline of above lemma is provided in Section 3.1. Next, we provide the following lemma from Anandkumar et al. (2015) which provides the dynamics of tensor power iteration when the initialization satisfies the constant correlation bound stated below.

Lemma 6 (Dynamics of tensor power iteration, phase 2) *Consider the rank-k tensor T of the form in* (12) *with rank condition $k\leq o(d^{1.5})$. Let the initial vectors $x_j^{(1)}$ satisfy the constant correlation bound*

$$|\langle x_j^{(1)},a_j\rangle|\geq 1-\gamma_j,$$

w.r.t. true components $a_j, j\in[k]$, for some constants $\gamma_j>0$. Let the output of the tensor power updates[6] in (2) *applied to all these different initialization vectors after $N=\Theta\left(\log\frac{1}{\epsilon}\right)$ iterations be stacked as columns of matrix $\hat{A}$. Then, we have w.h.p.[7]*

$$\left\|\hat{A}-A\right\|_F\leq\epsilon,$$

where the recovery error is up to permutation of columns of $\hat{A}$.

See Anandkumar et al. (2015) for the proof of above lemma. Given the above two lemmas, the learning result in Theorem 3 is directly proved.

Proof of Theorem 3 The result is proved by combining Lemma 5 and Lemma 6. Note that the initialization condition in (4) is w.h.p. satisfied given the SNR bound assumed. ■

Proof of Theorem 4 In Theorem 3, we provided the result given exact tensor by combining Lemmas 5 and 6. The only difference here is we are given an empirical estimate of the tensor and we need to incorporate the effect of noise in the empirical input. We now use

6. This result also needs an additional step of coordinate descent iterations since the true components are not the fixed points of power iteration; see Anandkumar et al. (2015, 2014a) for the details.
7. Anandkumar et al. (2015, 2014a) recover the vector up to sign since they work in the asymmetric case. In symmetric case it is easy to resolve sign ambiguity issue.

Theorem 1 that characterizes the effect of noise in first step (adapting Lemma 5 to noisy setting), and Anandkumar et al. (2015) that provide the result of Lemma 6 in noisy setting. In addition, the tensor concentration bound for multiview mixture model is analyzed in Theorem 1 of Anandkumar et al. (2014b) (Lemma 56 in Anandkumar et al. (2015)) that shows the error between empirical and exact tensors is bounded as

$$\|\hat{T}-T\| \leq \zeta\left(\frac{\sqrt{d}}{n}+\sqrt{\lambda_{\max}\frac{d}{n}}\right)+\zeta^2\left(\frac{d}{n}+\sqrt{\lambda_{\max}\frac{d^{1.5}}{n}}\right)+\zeta^3\left(\frac{d^{1.5}}{n}+\sqrt{\frac{d}{n}}\right).$$

The sample complexity requirement in (11) is then derived by imposing the error requirements in our noisy analysis of tensor power dynamics in Theorem 1 (see Equation (3)) and the noisy analysis of Lemma 6 (see Theorem 1 of Anandkumar et al. (2015) where the perturbation tensor E needs to be bounded as $\|E\| \leq \tilde{O}(\lambda_{\min})$). The final recovery error on $\left\|\hat{A}-A\right\|_F$ is also from Theorem 1 of Anandkumar et al. (2015). ■

3.1 Proof Outline of Lemma 5 (Noiseless Case of Theorem 1)

First step: We first intuitively show the first step of the algorithm makes progress. Suppose the tensor is $T=\sum_{j\in[k]} a_j\otimes a_j\otimes a_j$, and the initial vector x has correlation $|\langle x,a_1\rangle| \geq d^{\beta}\frac{\sqrt{k}}{d}$ with the first component. The result of the first iteration is the normalized version of the following vector:

$$\tilde{x}=\sum_{j\in[k]}\langle a_j,x\rangle^2 a_j.$$

Intuitively, this vector should have roughly $\langle a_1,\tilde{x}\rangle = d^{2\beta}\frac{k}{d^2}$ correlation with a_1 (as the other terms are random they don't contribute much). On the other hand, the norm of this vector is roughly $O(\sqrt{k}/d)$: this is because $\langle a_j,x\rangle^2$ for $j\neq 1$ is roughly[8] $1/d$, and the sum of k random vectors with length $1/d$ will have length roughly $O(\sqrt{k}/d)$. These arguments can be made precise showing the normalized version $\tilde{x}/\|\tilde{x}\|$ has correlation $d^{2\beta}\frac{\sqrt{k}}{d}$ with a_1 ensuring progress in the first step.

Going forward: As we explained, the basic idea behind proving Lemma 5 is to characterize the conditional distribution of random Gaussian tensor components a_j's given previous iterations. In particular, we show that the residual independent randomness left in these conditional distributions is large enough and we can exploit it to obtain tighter concentration bounds throughout the analysis of the iterations. The Gaussian assumption on the components, and small enough number of iterations are crucial in this argument.

Notations: For two vectors $u,v\in\mathbb{R}^k$, the Hadamard product denoted by $*$ is defined as the entry-wise multiplication of vectors, i.e., $(u*v)_j := u_jv_j$ for $j\in[k]$. For a matrix A, let $P_{\perp_A}$ denote the projection operator to the subspace orthogonal to column span of A. For a subspace R, let $R^{\perp}$ denote the space orthogonal to it. Therefore, for a subspace R, the projection operator on the subspace orthogonal to R is equivalently denoted by $P_{R^{\perp}}$ or $P_{\perp_R}$. For a random matrix D, let $D|\{u=Dv\}$ denote the conditional distribution of D given linear constraints $u=Dv$. We also use equality notation $\overset{(d)}{=}$ to denote the equivalence in distribution.

Lemma 5 involves analyzing the dynamics of power iteration in (2) for 3rd order rank-k tensors. For the rank-k tensor in (12), the power iterative form $x\leftarrow\frac{T(I,x,x)}{\|T(I,x,x)\|}$ can be written as

$$x^{(t+1)}=\frac{A\left(A^\top x^{(t)}\right)^{*2}}{\left\|A\left(A^\top x^{(t)}\right)^{*2}\right\|}, \tag{14}$$

where the multilinear form in (5) is used. Here, $A=[a_1\cdots a_k]\in\mathbb{R}^{d\times k}$ denotes the factor matrix, and for vector $y\in\mathbb{R}^k$, $y^{*2}:=y*y\in\mathbb{R}^k$ represents the element-wise square of entries of y.

We consider the case where $a_i\sim\mathcal{N}(0,\frac{1}{d}I)$ are i.i.d. drawn and we analyze the evolution of the dynamics of the power update. As explained earlier, for a given initialization $x^{(1)}$, the update in the first step can be analyzed easily since A is independent of $x^{(1)}$. However, in subsequent steps, the updates $x^{(t)}$ are dependent on A, and it is no longer clear how to provide a tight bound on the evolution of $x^{(t)}$. In this work, we provide a careful analysis by controlling the amount of "correlation build-up" by exploiting the structure of Gaussian matrices under linear constraints. This enables us to provide better guarantees for matrix A with Gaussian entries compared to general matrices A.

*Intermediate update steps and variables:*Before we proceed, we need to break down power update in (2) and introduce some intermediate update steps and variables as follows. Recall that $x^{(1)}\in\mathbb{R}^d$ denotes the initialization vector. Without loss of generality, let us analyze the convergence of power update to first component of rank-k tensor T denoted by a_1. Hence, let the first entry of $x^{(1)}$ denoted by $x_1^{(1)}$ be the maximum entry (in absolute value) of $x^{(1)}$, i.e., $x_1^{(1)}=\|x^{(1)}\|_\infty$. Let $B:=[a_2\ a_3\ \cdots\ a_k]\in\mathbb{R}^{d\times(k-1)}$, and therefore $A=[a_1|B]$. We break the power update formula in (2) into a few steps by introducing intermediate variables $y^{(t)}\in\mathbb{R}^k$ and $\tilde{x}^{(t+1)}\in\mathbb{R}^d$ as

$$y^{(t)}:=A^\top x^{(t)},\quad \tilde{x}^{(t+1)}:=A(y^{(t)})^{*2}.$$

Note that $\tilde{x}^{(t+1)}$ is the unnormalized version of $x^{(t+1)}:=\tilde{x}^{(t+1)}/\|\tilde{x}^{(t+1)}\|$, i.e., $\tilde{x}^{(t+1)}:=T(I,x^{(t)},x^{(t)})$. Thus, we need to jointly analyze the dynamics of all variables $x^{(t)}$, $y^{(t)}$ and $(y^{(t)})^{*2}$. Define

$$X^{[t]}:=\left[x^{(1)}|\ldots|x^{(t)}\right],\quad Y^{[t]}:=\left[y^{(1)}|\ldots|y^{(t)}\right].$$

Matrix B is randomly drawn with i.i.d. Gaussian entries $B_{ij}\sim\mathcal{N}(0,\frac{1}{d})$. As the iterations proceed, we consider the following conditional distributions

$$B^{(t,1)}:=B|\{X^{[t]},Y^{[t]}\},\quad B^{(t,2)}:=B|\{X^{[t+1]},Y^{[t]}\}. \tag{15}$$

Thus, $B^{(t,1)}$ is the conditional distribution of B at the middle of t^{th} iteration (before update step $\tilde{x}^{(t+1)}=A(y^{(t)})^{*2}$) and $B^{(t,2)}$ is the conditional distribution at the end of t^{th} iteration (after update step $\tilde{x}^{(t+1)}=A(y^{(t)})^{*2}$). By analyzing the above conditional distributions, we can characterize the left independent randomness in B.

8. The correlation between two unit Gaussian vectors in d dimensions is roughly $1/\sqrt{d}$.

3.1.1 Conditional Distributions

In order to characterize the conditional distribution of B under evolution of $x^{(t)}$ and $y^{(t)}$ in (15), we exploit the following basic fact (see (Bayati and Montanari, 2010) for proof).

Lemma 7 (Conditional distribution of Normal matrices under linear condition) *Consider random matrix D with i.i.d. Gaussian entries $D_{ij} \sim \mathcal{N}(0,\sigma^2)$. Conditioned on $u = Dv$ with known vectors u and v, the matrix D is distributed as*

$$D|\{u = Dv\} \overset{(\mathrm{d})}{=} \frac{1}{\|v\|^2} uv^\top + \tilde{D}P_{\perp_v},$$

where random matrix $\tilde{D}$ is an independent copy of D with i.i.d. Gaussian entries $\tilde{D}_{ij} \sim \mathcal{N}(0,\sigma^2)$, and $P_{\perp_v}$ is the projection operator on to the subspace orthogonal to v.

We refer to $\tilde{D}P_{\perp_v}$ as the *residual* random matrix since it represents the remaining *randomness* left after conditioning. It is a random matrix whose rows are independent random vectors that are orthogonal to v, and the variance in each direction orthogonal to v is equal to σ^2.

The above Lemma can be exploited to characterize the conditional distribution of B introduced in (15). However, a naive direct application using the constraint $Y^{[t]} = A^\top X^{[t]}$ is not transparent for analysis. The reason is the evolution of $x^{(t)}$ and $y^{(t)}$ are themselves governed by the conditional distribution of B given previous iterations. Therefore, we need the following recursive version of Lemma 7 which can be immediately argued by induction.

Corollary 8 (Iterative conditioning) *Consider random matrix D with i.i.d. Gaussian entries $D_{ij} \sim \mathcal{N}(0,\sigma^2)$, and let $F \overset{(\mathrm{d})}{=} P_{\perp_C} D P_{\perp_R}$ be the random Gaussian matrix whose columns are orthogonal to space C and rows are orthogonal to space R. Conditioned on the linear constraint $u = Dv$, where[9] $u \in C^\perp$, the matrix F is distributed as*

$$F|\{u = Dv\} \overset{(\mathrm{d})}{=} \frac{1}{\|(P_{\perp_R} v)\|^2} u(P_{\perp_R} v)^\top + P_{\perp_C} \tilde{D} P_{\perp_{\{R,v\}}},$$

where random matrix $\tilde{D}$ is an independent copy of D with i.i.d. Gaussian entries $\tilde{D}_{ij} \sim \mathcal{N}(0,\sigma^2)$.

Thus, the *residual* random matrix $P_{\perp_C}\tilde{D}P_{\perp_{\{R,v\}}}$ is a random Gaussian matrix whose columns are orthogonal to C and rows are orthogonal to $\mathrm{span}\{R, v\}$. The variance in any remaining dimension is equal to σ^2.

3.1.2 Form of Iterative Updates

Now we exploit the conditional distribution arguments proposed in the previous section to characterize the conditional distribution of B given the update variables x and y up to the current iteration; recall (15) where $B^{(t,1)}$ is the conditional distribution of B at the middle of t^{th} iteration and $B^{(t,2)}$ at the end of t^{th} iteration. Before that, we need to introduce some more intermediate variables.

9. We need that $u \in C^\perp$, otherwise the event $u = Dv$ is impossible.

Intermediate variables: We separate the first entry of y and $(y)^{*2}$ from the rest, i.e., we have

$$y_1^{(t)} = a_1^\top x^{(t)}, \quad y_{-1}^{(t)} = B^\top x^{(t)} \sim (B^{(t-1,2)})^\top x^{(t)},$$

where $y_{-1}^{(t)} \in \mathbb{R}^{k-1}$ denotes $y^{(t)} \in \mathbb{R}^k$ with the first entry removed. The update formula for $\tilde{x}^{(t+1)}$ can be also decomposed as

$$\tilde{x}^{(t+1)} = (y_1^{(t)})^2 a_1 + Bw^{(t)} \sim (y_1^{(t)})^2 a_1 + B^{(t,1)} w^{(t)},$$

where

$$w^{(t)} := (y_{-1}^{(t)})^{*2} \in \mathbb{R}^{k-1},$$

is the new intermediate variable in the power iterations. Let $B_{\text{res.}}^{(t,1)}$ and $B_{\text{res.}}^{(t,2)}$ denote the *residual* random matrices corresponding to $B^{(t,1)}$ and $B^{(t,2)}$ respectively, and

$$u^{(t+1)} := B_{\text{res.}}^{(t,1)} w^{(t)}, \quad v^{(t)} := (B_{\text{res.}}^{(t-1,2)})^\top x^{(t)},$$

where $u^{(t)} \in \mathbb{R}^d$ and $v^{(t)} \in \mathbb{R}^{k-1}$ are respectively the part of $x^{(t)}$ and $y_{-1}^{(t)}$ representing the residual randomness after conditioning on the previous iterations. We also summarize all variables and notations in Table 1 in the Appendix which can be used as a reference throughout the paper.

Finally we make the following observations.

Lemma 9 (Form of iterative updates) *The conditional distribution of B at the middle of t^{th} iteration denoted by $B^{(t,1)}$ satisfies*

$$B^{(t,1)} \overset{(\mathrm{d})}{=} \sum_{i\in[t-1]} \frac{u^{(i+1)}(P_{\perp_{W^{[i-1]}}} w^{(i)})^\top}{\|P_{\perp_{W^{[i-1]}}} w^{(i)}\|^2} + \sum_{i\in[t]} \frac{P_{\perp_{X^{[i-1]}}} x^{(i)} (v^{(i)})^\top}{\|P_{\perp_{X^{[i-1]}}} x^{(i)}\|^2} + B_{\text{res.}}^{(t,1)}, \tag{16}$$

$$B_{\text{res.}}^{(t,1)} \overset{(\mathrm{d})}{=} P_{\perp_{X^{[t]}}} \tilde{B} P_{\perp_{W^{[t-1]}}}, \tag{17}$$

where random matrix $\tilde{B}$ is an independent copy of B with i.i.d. Gaussian entries $\tilde{B}_{ij} \sim \mathcal{N}(0, \frac{1}{d})$. Similarly, the conditional distribution of B at the end of t^{th} iteration denoted by $B^{(t,2)}$ satisfies

$$B^{(t,2)} \overset{(\mathrm{d})}{=} \sum_{i\in[t]} \left(\frac{u^{(i+1)}(P_{\perp_{W^{[i-1]}}} w^{(i)})^\top}{\|P_{\perp_{W^{[i-1]}}} w^{(i)}\|^2} + \frac{P_{\perp_{X^{[i-1]}}} x^{(i)} (v^{(i)})^\top}{\|P_{\perp_{X^{[i-1]}}} x^{(i)}\|^2} \right) + B_{\text{res.}}^{(t,2)}, \tag{18}$$

$$B_{\text{res.}}^{(t,2)} \overset{(\mathrm{d})}{=} P_{\perp_{X^{[t]}}} B' P_{\perp_{W^{[t]}}}, \tag{19}$$

where random matrix B' is an independent copy of B with i.i.d. Gaussian entries $B'_{ij} \sim \mathcal{N}(0, \frac{1}{d})$.

The lemma can be directly proved by applying the iterative conditioning argument in Corollary 8. See the detailed proof in the appendix.

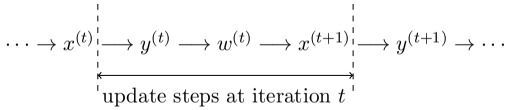


Figure 2: Flow of the power update algorithm stating intermediate steps. Iteration t for which the inductive step should be argued is also indicated.

3.1.3 Analysis of Iterative Updates

Lemma 9 characterizes the conditional distribution of B given the update variables x and y up to the current iteration; see (15) for the definition of conditional forms of B denoted by $B^{(t,1)}$ and $B^{(t,2)}$. Intuitively, when the number of iterations $t \ll d$, then the residual independent randomness left in $B^{(t,1)}$ and $B^{(t,2)}$ (respectively denoted by $B^{(t,1)}_{\text{res.}}$ and $B^{(t,2)}_{\text{res.}}$) characterized in Lemma 9 is large enough and we can exploit it to obtain tighter concentration bounds throughout the analysis of the iterations.

Note that the goal is to show that under $t \ll d$, the iterations $x^{(t)}$ converge to the true component with constant error, i.e., $|\langle x^{(t)}, a_1\rangle| \geq 1-\gamma$ for some constant $\gamma > 0$. If this already holds before iteration t we are done, and if it does not hold, next iteration is analyzed to finally achieve the goal. This analysis is done via *induction argument*. During the iterations, we maintain several invariants to analyze the dynamics of power update. The goal is to ensure progress in each iteration as in (20).

Induction hypothesis: The following are assumed at the beginning of the iteration t as induction hypothesis; see Figure 2 for the scope of inductive step.

1. *Length of Projection on x:*

$$\delta_t \leq \|P_{\perp_{X^{[t-1]}}} x^{(t)}\| \leq 1,$$

where δ_t is of order $1/\operatorname{polylog} d$, and the value of δ_t only depends on t and $\log d$.

2. *Length of Projection on w:*

$$\delta'_{t-1}\frac{\sqrt{k}}{d} \leq \|P_{\perp_{W^{[t-2]}}} w^{(t-1)}\| \leq \Delta'_{t-1}\frac{\sqrt{k}}{d},$$
$$\|P_{\perp_{W^{[t-2]}}} w^{(t-1)}\|_\infty \leq \Delta'_{t-1}\frac{1}{d},$$

where δ'_t is of order $1/\operatorname{polylog} d$ and Δ'_t is of order $\operatorname{polylog} d$. Both δ'_t and Δ'_t only depend on t and $\log d$.

3. *Progress:*[10]

$$|\langle a_1, x^{(t)}\rangle| \in [\delta^*_t, \Delta^*_t] d^{\beta 2^{t-1}}\frac{\sqrt{k}}{d}, \tag{20}$$
$$|\langle a_1, P_{\perp_{X^{[t-1]}}} x^{(t)}\rangle| \leq \Delta^*_t d^{\beta 2^{t-1}}\frac{\sqrt{k}}{d}.$$

4. *Norm of u,v:*

$$\frac{\delta_{t-1}}{2}\sqrt{\frac{k}{d}} \leq \|v^{(t-1)}\| \leq 2\sqrt{\frac{k}{d}},$$
$$\frac{\delta'_{t-1}}{2}\frac{\sqrt{k}}{d} \leq \|u^{(t)}\| \leq 2\Delta'_{t-1}\frac{\sqrt{k}}{d}.$$

The analysis for basis of induction and inductive step are provided in Appendix B.

3.2 Effect of Noise in Theorem 1

Given rank-k random tensor T in (12), and a starting point $x^{(1)}$, our analysis in the noiseless setting shows that the tensor power iteration in (2) outputs a vector which will be close to a_j if $x^{(1)}$ has a large enough correlation with a_j.

Now suppose we are given noisy tensor $\hat{T} = T + E$ where E has some small norm. In this case where the noise is also present, we get a sequence $\hat{x}^{(t)} = x^{(t)} + \xi^{(t)}$ where $x^{(t)}$ is the component not incorporating any noise (as in previous section[11]), while $\xi^{(t)}$ represents the contribution of noise tensor E in the power iteration; see (21) below. We prove that $\xi^{(t)}$ is a very small noise that does not change our calculations stated in the following lemma.

Lemma 10 (Bounding norm of error) *Suppose the spectral norm of the error tensor E is bounded as*

$$\|E\| \leq \epsilon\sqrt{k}/d, \quad \text{where} \quad \epsilon < o(\sqrt{k}/d).$$

Then the noise vector $\xi^{(t)}$ at iteration t satisfies the ℓ_2 norm bound

$$\|\xi^{(t)}\| \leq \tilde{O}(d^{\beta 2^{t-1}}\epsilon).$$

Note that when t is the first number such that $d^{\beta 2^{t-1}} \geq d/\sqrt{k}$, we have $\|\xi^{(t)}\| = o(1)$.

Notice that since when $d^{\beta 2^{t-1}} \geq d/\sqrt{k}$, the main induction is already over and we know $x^{(t)}$ is constant close to the true component, and thus, the noise is always small.

Proof idea: We now provide an overview of ideas for proving the above lemma; see Appendix D for the complete proof which is based on an induction argument. We first

10. Note that although the bounds on $y^{(t)}_{-1}$ are argued at iteration t, the bound on the first entry of $y^{(t)}$ denoted by $y^{(t)}_1 = \langle a_1, x^{(t)}\rangle$ is assumed here in the induction hypothesis at the end of iteration $t-1$.

11. Note that there is a subtle difference between notation $x^{(t)}$ in the noiseless and noisy settings. In the noiseless setting, this vector is normalized, while in the noisy setting the whole vector $\hat{x}^{(t)} = x^{(t)} + \xi^{(t)}$ is normalized.

write the following recursion expanding the contribution of main signal and noise terms in the tensor power iteration as

$$\begin{aligned} x^{(t+1)} + \xi^{(t+1)} &= \text{Norm}\left(\hat{T}(x^{(t)} + \xi^{(t)}, x^{(t)} + \xi^{(t)}, I)\right) \\ &= \text{Norm}\left(T(x^{(t)}, x^{(t)}, I) + 2T(x^{(t)}, \xi^{(t)}, I) + T(\xi^{(t)}, \xi^{(t)}, I) + E(\hat{x}^{(t)}, \hat{x}^{(t)}, I)\right), \end{aligned} \tag{21}$$

where for vector v, we have $\text{Norm}(v) := v/\|v\|$, i.e., it normalizes the vector. The first term is the desired main signal and should have the largest norm, and the rest of the terms are the noise terms. The third term is of order $\|\xi^{(t)}\|^2$, and hence, it should be fine whenever we choose $\|E\|$ to be small enough. The last term is $O(\|E\|)$ and is the same for all iterations so that is also fine. The problematic term is the second term, whose norm if we bound naively is $2\|\xi^{(t)}\|$. However the normalization factor also contributes a factor of roughly $d/\sqrt{k}$, and thus, this term grows exponentially; it is still fine if we just do a constant number of iterations, but the exponent will depend on the number of iterations.

In order to solve this problem, and make sure that the amount of noise we can tolerate is independent of the number of iterations, we need a better way to bound the noise term $\xi^{(t)}$. The main problem here is we bound the norm of $\|T(x^{(t)}, \xi^{(t)}, I)\|$ by $\|T\|\|\xi^{(t)}\| \leq O(\xi^{(t)})$, by doing this we ignored the fact that $x^{(t)}$ is uncorrelated with the components in T. In order to get a tighter bound, we introduce another norm $\|\cdot\|_*$; see Definition 21 for the exact form. Intuitively, the norm $\|\cdot\|_*$ captures the fact that x does not have a high correlation with the components (except for the first component that x will converge to), and gives a better bound. In particular we have $\|T(x^{(t)}, \xi^{(t)}, I)\| \approx \frac{\sqrt{k}}{d}\|\xi^{(t)}\|_2$. Therefore, the normalization factor is compensated by the additional term $\frac{\sqrt{k}}{d}$.

4. Conclusion

In this paper, we provide a novel analysis for the dynamics of third order tensor power iteration showing convergence guarantees to vectors having constant correlation with the tensor component. This enables us to prove unsupervised learning of latent variable models in the challenging overcomplete regime where the hidden dimensionality is larger than the observed dimension. The main technical observation is that under random Gaussian tensor components and small number of iterations, the residual randomness in the components (which are involved in the iterative steps) are sufficiently large. This enables us to show progress in the next iteration of the update step. As future work, it is very interesting to generalize this analysis to higher order tensor power iteration, and more generally to other kinds of iterative updates.

Acknowledgments

A. Anandkumar is supported in part by Microsoft Faculty Fellowship, NSF Career award CCF-1254106, NSF award CCF-1219234, ONR award N00014-14-1-0665, ARO YIP award W911NF-13-1-0084, and AFOSR YIP award FA9550-15-1-0221. M. Janzamin is supported by NSF Award CCF-1219234.

Variable	Space	Description	Recursion formula
A	$\mathbb{R}^{d\times k}$	mapping matrix in (14)	n.a.
$x^{(t)}$	$\mathbb{R}^d$	update variable in (14)	$x^{(t+1)} := \frac{A(y^{(t)})^{*2}}{\|A(y^{(t)})^{*2}\|}$
$y^{(t)}$	$\mathbb{R}^k$	intermediate variable in (14)	$y^{(t)} := A^\top x^{(t)}$
$\tilde{x}^{(t)}$	$\mathbb{R}^d$	unnormalized version of $x^{(t)}$	$\tilde{x}^{(t+1)} := A(y^{(t)})^{*2}$
$\hat{x}^{(t)}$	$\mathbb{R}^d$	noisy version of $x^{(t)}$	$\hat{x}^{(t)} = x^{(t)} + \xi^{(t)}$; see (21)
$\xi^{(t)}$	$\mathbb{R}^d$	Contribution of noise in tensor power update given noisy tensor $\hat{T} = T + E$	$\hat{x}^{(t)} = x^{(t)} + \xi^{(t)}$; see (21)
B	$\mathbb{R}^{d\times(k-1)}$	matrix $A := [a_1\ a_2\ \cdots\ a_k]$ with first column removed, i.e., $B := [a_2\ a_3\ \cdots\ a_k]$. Note that the first column a_1 is the desired one to recover.	n.a.
$B^{(t,1)}$	$\mathbb{R}^{d\times(k-1)}$	conditional distribution of B given previous iterations at the middle of t^{th} iteration (before update step $\tilde{x}^{(t+1)} = A(y^{(t)})^{*2}$)	$B^{(t,1)} \overset{(\text{d})}{=} B\|\{X^{[t]}, Y^{[t]}\}$
$B^{(t,2)}$	$\mathbb{R}^{d\times(k-1)}$	conditional distribution of B given previous iterations at the end of t^{th} iteration (after update step $\tilde{x}^{(t+1)} = A(y^{(t)})^{*2}$)	$B^{(t,2)} \overset{(\text{d})}{=} B\|\{X^{[t+1]}, Y^{[t]}\}$
$B_{\text{res.}}^{(t,1)}$	$\mathbb{R}^{d\times(k-1)}$	residual independent randomness left in $B^{(t,1)}$; see Lemma 9.	see equation (17)
$B_{\text{res.}}^{(t,2)}$	$\mathbb{R}^{d\times(k-1)}$	residual independent randomness left in $B^{(t,2)}$; see Lemma 9.	see equation (19)
$w^{(t)}$	$\mathbb{R}^{k-1}$	intermediate variable in update formula (14)	$w^{(t)} := (y_{-1}^{(t)})^{*2}$
$u^{(t)}$	$\mathbb{R}^d$	part of $x^{(t)}$ representing the left independent randomness	$u^{(t+1)} := B_{\text{res.}}^{(t,1)} w^{(t)}$
$v^{(t)}$	$\mathbb{R}^{k-1}$	part of $y_{-1}^{(t)}$ representing the left independent randomness	$v^{(t)} := (B_{\text{res.}}^{(t-1,2)})^\top x^{(t)}$

Table 1: Table of parameters and variables. Superscript (t) denotes the variable at t-th iteration.

Appendix A. Proof of Lemma 9

Proof of Lemma 9 Recall that we have updates of the form

$$\tilde{x}^{(t+1)} = A(y^{(t)})^{*2}, \quad w^{(t)} := (y_{-1}^{(t)})^{*2}, \quad y^{(t)} = A^\top x^{(t)}.$$

Let

$$X^{[t]\setminus 1} := \left[x^{(2)}|\dots|x^{(t)}\right],$$

and let the rows of $Y^{[t]}$ are partitioned as the first and the rest of rows as

$$Y^{[t]} = \left[Y_1^{[t]\top} \middle| Y_{-1}^{[t]\top}\right]^\top.$$

We now make the following simple observations

$$\begin{aligned} B^{(t,1)} &\overset{(d)}{=} B|\{Y^{[t]} = A^\top X^{[t]},\ \tilde{X}^{[t]\setminus 1} = A(Y^{[t-1]})^{*2}\} \\ &\overset{(d)}{=} B|\{Y_{-1}^{[t]} = B^\top X^{[t]},\ \tilde{X}^{[t]\setminus 1} = a_1(Y_1^{[t-1]})^{*2} + BW^{[t-1]}\} \\ &\overset{(d)}{=} B|\{v^{(1)} = B^\top x^{(1)}, \dots, v^{(t)} = (B_{\text{res.}}^{(t-1,2)})^\top x^{(t)}, \\ &\qquad u^{(2)} = B_{\text{res.}}^{(1,1)} w^{(1)}, \dots, u^{(t)} = B_{\text{res.}}^{(t-1,1)} w^{(t-1)}\}, \end{aligned}$$

where the second equivalence comes from the fact that B is matrix A with first column removed. Now applying Corollary 8, we have the result. The distribution of $B^{(t,2)}$ follow similarly. ■

Appendix B. Analysis of Induction Argument

In this section, we analyze the basis of induction and inductive step for the induction argument proposed in Section 3.1.3 for the proof of Lemma 5.

B.1 Basis of Induction

We first show that the hypothesis holds for initialization vector $x^{(1)}$ as the basis of induction.

Claim 1 (Basis of induction) *The induction hypothesis is true for $t = 1$.*

Proof Notice that induction hypothesis for $t = 1$ only involves the bounds on $\|x^{(1)}\|$ and $\langle a_1, x^{(1)}\rangle$ as in Hypotheses 1 and 3, respectively. These bounds are directly argued by the correlation assumption on the initial vector $x^{(1)}$ stated in (13) where $\delta_1 = \delta_1^* = \Delta_1^* = 1$. ■

B.2 Inductive Step

Assuming the induction hypothesis holds for all the values till the end of iteration $t-1$ (stated in Section 3.1.3), we analyze the t-th iteration of the algorithm, and prove that induction hypothesis also holds for the values at the end of iteration t. See Figure 2 where the scope of iteration t and the flow of the algorithm is shown. In the rest of this section, we pursue the flow of the algorithm at iteration t starting from computing $y^{(t)}$ and ending up with computing $x^{(t+1)}$ to prove the desired induction hypothesis at iteration t.

B.2.1 Hypothesis 4

We start by showing that the induction Hypothesis 4 holds at iteration t using the induction Hypotheses 1 and 2 in the previous iteration.

Claim 2 *We have*

$$\begin{aligned} &\frac{\delta_t}{2}\sqrt{\frac{k}{d}} \le \|v^{(t)}\| \le 2\sqrt{\frac{k}{d}}, \\ &\frac{\delta_t'}{2}\frac{\sqrt{k}}{d} \le \|u^{(t+1)}\| \le 2\Delta_t'\frac{\sqrt{k}}{d}. \end{aligned}$$

Proof Recall that $v^{(t)} := (B_{\text{res.}}^{(t-1,2)})^\top x^{(t)}$, and by applying the form of $B_{\text{res.}}^{(t-1,2)}$ in (19), we have

$$v^{(t)} \overset{(d)}{=} P_{\perp_{W^{[t-1]}}} B'^\top P_{\perp_{X^{[t-1]}}} x^{(t)}. \tag{22}$$

Since random matrix $B' \in \mathbb{R}^{d\times(k-1)}$ is an independent copy of B with i.i.d. Gaussian entries $B'_{ij} \sim \mathcal{N}(0, \frac{1}{d})$, we know $v^{(t)}$ is a random Gaussian vector in the subspace orthogonal to $W^{[t-1]}$. On the other hand, for any vector $z \in \mathbb{R}^d$, we have

$$\mathbb{E}\left[\|B'^\top z\|^2\right] = z^\top \mathbb{E}\left[B'B'^\top\right] z = \frac{k-1}{d}\|z\|^2,$$

where $\mathbb{E}\left[B'B'^\top\right] = \frac{k-1}{d} I$ is exploited. Let $z = P_{\perp_{X^{[t-1]}}} x^{(t)}$. Then, by applying the above equality to the expansion of $v^{(t)}$ in (22), we have

$$\mathbb{E}\left[\|v^{(t)}\|^2\right] = \frac{k-t}{k-1}\cdot\frac{k-1}{d}\cdot\|P_{\perp_{X^{[t-1]}}} x^{(t)}\|^2 = \frac{k-t}{d}\cdot\|P_{\perp_{X^{[t-1]}}} x^{(t)}\|^2 \in \left[\delta_t^2\frac{k}{d}\left(1-\frac{t}{k}\right), \frac{k}{d}\right],$$

where $\dim(W^{[t-1]}) = t-1$ is also used in the first step, and the last step is concluded from Hypothesis 1. Finally, by concentration property of random Gaussian vectors, when $t \ll d$ we have with high probability

$$\|v^{(t)}\| \in \left[\frac{\delta_t}{2}\sqrt{\frac{k}{d}}, 2\sqrt{\frac{k}{d}}\right].$$

Similarly, for $u^{(t+1)} := B_{\text{res.}}^{(t,1)} w^{(t)}$, and by applying the form of $B_{\text{res.}}^{(t,1)}$ in (17), we have

$$u^{(t+1)} \overset{(d)}{=} P_{\perp_{X^{[t]}}} \tilde{B} P_{\perp_{W^{[t-1]}}} w^{(t)}. \tag{23}$$

Since random matrix $\tilde{B} \in \mathbb{R}^{d\times(k-1)}$ is an independent copy of B with i.i.d. Gaussian entries $\tilde{B}_{ij} \sim \mathcal{N}(0, \frac{1}{d})$, we know $u^{(t+1)}$ is a random Gaussian vector in the subspace orthogonal to $X^{[t]}$. On the other hand, for any vector $z \in \mathbb{R}^{k-1}$, we have

$$\mathbb{E}\left[\|\tilde{B}z\|^2\right] = z^\top \mathbb{E}\left[\tilde{B}^\top \tilde{B}\right] z = \|z\|^2,$$

where $\mathbb{E}\left[\tilde{B}^\top \tilde{B}\right] = I$ is exploited. Let $z = P_{\perp_{W^{[t-1]}}} w^{(t)}$. Then, by applying the above equality to the expansion of $u^{(t+1)}$ in (23), we have

$$\mathbb{E}\left[\|u^{(t+1)}\|^2\right] = \frac{d-t}{d} \cdot \|P_{\perp_{W^{[t-1]}}} w^{(t)}\|^2 \in \left[(\delta_t')^2 \frac{k}{d^2}\left(1 - \frac{t}{d}\right), (\Delta_t')^2 \frac{k}{d^2}\right],$$

where $\dim(X^{[t]}) = t$ is also used in the first step, and the last step is concluded from Hypothesis 2. Finally, by concentration property of random Gaussian vectors, when $t \ll d$ we have with high probability

$$\|u^{(t+1)}\| \in \left[\frac{\delta_t'}{2}\frac{\sqrt{k}}{d}, 2\Delta_t'\frac{\sqrt{k}}{d}\right].$$

■

B.2.2 Hypothesis 2

Computing $y^{(t)}$: In the first step of iteration t, the algorithm computes $y^{(t)}$. By induction Hypothesis 3, we know $|y_1^{(t)}| = \tilde{\Theta}(d^{\beta 2^{t-1}}\sqrt{k}/d)$. The other coordinates of $y^{(t)} := A^\top x^{(t)}$ are $y_{-1}^{(t)} = B^\top x^{(t)}$ which conditioning on the previous iterations are equivalent (in distribution) to

$$\begin{aligned} y_{-1}^{(t)} &\overset{(d)}{=} \left(B^{(t-1,2)}\right)^\top x^{(t)} \\ &= \left(\sum_{i\in[t-1]} \left(\frac{u^{(i+1)}(P_{\perp_{W^{[i-1]}}} w^{(i)})^\top}{\|P_{\perp_{W^{[i-1]}}} w^{(i)}\|^2} + \frac{P_{\perp_{X^{[i-1]}}} x^{(i)}(v^{(i)})^\top}{\|P_{\perp_{X^{[i-1]}}} x^{(i)}\|^2}\right) + B_{\text{res.}}^{(t-1,2)}\right)^\top x^{(t)} \\ &= \sum_{i\in[t-1]} \left(\tilde{\Theta}\left(\frac{d^2}{k}\right) P_{\perp_{W^{[i-1]}}} w^{(i)} \langle u^{(i+1)}, x^{(t)}\rangle + \tilde{\Theta}(1) v^{(i)} \langle P_{\perp_{X^{[i-1]}}} x^{(i)}, x^{(t)}\rangle\right) + v^{(t)}, \quad (24) \end{aligned}$$

where form of $B^{(t-1,2)}$ in (18) is used in the second equality. The bounds on the norms come from Hypotheses 1 and 2. The last term is by definition $v^{(t)} := (B_{\text{res.}}^{(t-1,2)})^\top x^{(t)}$. Note that differences in polylog factors in the (upper and lower) bounds in Hypotheses 1 and 2 are represented by notation $\tilde{\Theta}(\cdot)$.

We will establish subsequently that if $k > d$, the terms involving $v^{(i)}$'s in the above expansion dominate, and the terms involving $P_{\perp_{W^{[i-1]}}} w^{(i)}$'s have norm of a smaller order; see Claim 3.

Computing $w^{(t)}$: In the next step of the algorithm at iteration t, $w^{(t)}$ is computed for which we now argue if the induction hypothesis holds up to iteration t, both lower and upper bounds at iteration t as $\|P_{\perp_{W^{[t-1]}}} w^{(t)}\| \in [\delta_t', \Delta_t']\frac{\sqrt{k}}{d}$ (see induction Hypothesis 2) also hold.

Lower bound: For the lower bound, intuitively the *fresh* random vector $v^{(t)}$ should bring enough randomness into $w^{(t)}$. We formulate that in the following lemma.

Lemma 11 *Suppose R and R' are two subspaces in $\mathbb{R}^k$ with dimension at most $t \le \frac{k}{16(\log k)^2}$. Let $p \in \mathbb{R}^k$ be an arbitrary vector, $z \in \mathbb{R}^k$ be a uniformly random Gaussian vector in the space orthogonal to R, and finally $w = (p+z)*(p+z)$. Then with high probability, we have*

$$\|P_{\perp_{R'}} w\| \ge \frac{\mathbb{E}[\|z\|^2]}{40\sqrt{k}}.$$

Recall that $w^{(t)} := y_{-1}^{(t)} * y_{-1}^{(t)}$, and $y_{-1}^{(t)}$ is expanded in (24) as sum of an arbitrary vector and a random Gaussian vector. Applying above lemma with $R = R' = \text{span}(W^{[t-1]})$, we have with high probability

$$\|P_{\perp_{W^{[t-1]}}} w^{(t)}\| \ge \frac{\mathbb{E}[\|v^{(t)}\|^2]}{40\sqrt{k}} \ge \frac{\delta_t^2}{160}\sqrt{k}/d,$$

where Hypothesis 4 gives lower bound $\|v^{(t)}\| \ge \delta_t/2\sqrt{k/d}$ (used in the second inequality). By choosing $\delta_t' = \delta_t^2/160$ the lower bound in Hypothesis 2 is proved.

Upper bound: In order to prove the upper bounds in Hypothesis 2, we follow the sequence of arguments below:

Claim 3: $\|y_{-1}^{(t)}\|_\infty \overset{(\cdot)^2}{\Longrightarrow} \|w^{(t)}\|_\infty \overset{\text{Lemma 12}}{\Longrightarrow} \|P_{\perp_{W^{[t-1]}}} w^{(t)}\|_\infty \Rightarrow \|P_{\perp_{W^{[t-1]}}} w^{(t)}\|$

First we prove a bound on the infinity norm of $y_{-1}^{(t)}$:

Claim 3 (Upper bound on $\|y_{-1}^{(t)}\|_\infty$) *We have*

$$\|y_{-1}^{(t)}\|_\infty \le \frac{t}{\delta_t}\frac{\log d}{\sqrt{d}} + (t-1)\left(\frac{\Delta_{t-1}'}{\delta_{t-1}'}\right)^2 \frac{1}{\sqrt{k}} = \tilde{O}\left(\frac{1}{\sqrt{d}}\right).$$

Proof We exploit the induction hypothesis to bound the ℓ_∞ norm of all the terms in the expansion of $y_{-1}^{(t)}$ in (24).

For the terms involving $v^{(i)}$, since they are random Gaussian vectors with expected square norm at most k/d, by Lemma 19 we know $\|v^{(i)}\|_\infty \le \frac{\log d}{\sqrt{d}}$ with high probability. In addition, for $v^{(i)}$, $i < t$, the coefficient is bounded as

$$\frac{\langle P_{\perp_{X^{[i-1]}}} x^{(i)}, x^{(t)}\rangle}{\|P_{\perp_{X^{[i-1]}}} x^{(i)}\|^2} \le \frac{1}{\|P_{\perp_{X^{[i-1]}}} x^{(i)}\|} \le \frac{1}{\delta_i}, \quad (25)$$

where the last step uses Hypothesis 1. Therefore, the total contribution from terms involving $v^{(i)}$ in $\|y_{-1}^{(t)}\|_\infty$ is bounded by $\frac{t}{\delta_t}\frac{\log d}{\sqrt{d}}$.

For the terms involving $P_{\perp_{W^{[i-1]}}} w^{(i)}, i \in [t-1]$, we have from Hypothesis 2 that the ℓ_∞ norm is bounded as $\|P_{\perp_{W^{[i-1]}}} w^{(i)}\|_\infty \le \Delta_i' \frac{1}{d}$. In addition, the corresponding coefficient is bounded by

$$\frac{\langle u^{(i+1)}, x^{(t)}\rangle}{\|P_{\perp_{W^{[i-1]}}} w^{(i)}\|^2} \le \frac{\|u^{(i+1)}\| \cdot \|x^{(t)}\|}{\|P_{\perp_{W^{[i-1]}}} w^{(i)}\|^2} \le \frac{2\Delta_i'}{\delta_i'^2}\frac{d}{\sqrt{k}}. \tag{26}$$

Again bounds in Hypotheses 2 and 4 are exploited in the last inequality. Hence, the total contribution from terms involving $P_{\perp_{W^{[i-1]}}} w^{(i)}, i \in [t-1]$ in $\|y_{-1}^{(t)}\|_\infty$ is bounded by $(t-1)\left(\frac{\Delta_{t-1}'}{\delta_{t-1}'}\right)^2 \frac{1}{\sqrt{k}}$.

Combining the above bounds finishes the proof. ■

Since $w^{(t)} := y_{-1}^{(t)} * y_{-1}^{(t)}$, the above claim immediately implies that

$$\|w^{(t)}\|_\infty \le \tilde{O}\left(\frac{1}{d}\right). \tag{27}$$

Now we have the ℓ_∞ norm on w, however we need to bound the ℓ_∞ norm of the projected vector $P_{\perp_{W^{[t-1]}}} w^{(t)}$. Intuitively this is clear as the vectors in the space $W^{[t-1]}$ all have small ℓ_∞ as guaranteed by induction hypothesis. We formalize this intuition using the following lemma.

Lemma 12 *Suppose R is a subspace in $\mathbb{R}^k$ of dimension t', such that there is a basis $\{r_1, \ldots, r_{t'}\}$ with $\|r_i\|_\infty \le \frac{\Delta}{\sqrt{k}}$ and $\|r_i\| = 1$. Let $p \in \mathbb{R}^k$ be an arbitrary vector, then*

$$\|P_{\perp_R} p\|_\infty \le \|p\|_\infty + \|p\|\Delta\frac{\sqrt{t'}}{\sqrt{k}}.$$

Let $R = \mathrm{span}(W^{[t-1]})$. Then the vectors $P_{\perp_{W^{[i-1]}}} w^{(i)} / \|P_{\perp_{W^{[i-1]}}} w^{(i)}\|$, $i \in [t-1]$ form a basis for subspace R, and we know from Hypothesis 2 that the ℓ_∞ norm of each of these basis vectors is bounded by $\frac{\Delta}{\sqrt{k}}$ for $\Delta := \frac{\Delta_{t-1}'}{\delta_{t-1}'}$ which is of order polylog d. Applying above lemma, we have

$$\|P_{\perp_{W^{[t-1]}}} w^{(t)}\|_\infty \le \|w^{(t)}\|_\infty(1 + \Delta\sqrt{t-1}) \le \frac{\Delta_t'}{d},$$

where the last inequality uses bound (27), and appropriate choosing for Δ_t' which is of order polylog d and only depends on t and $\log d$. This concludes the upper bound on the ℓ_∞ norm in Hypothesis 2. The upper bound on the ℓ_2 norm is also immediately argued using this ℓ_∞ norm bound where an additional $\sqrt{k}$ factor shows up.

B.2.3 Hypothesis 1

Computing $x^{(t+1)}$:

In the next step of iteration t, the algorithm computes $x^{(t+1)}$. Conditioning on the previous iterations, the unnormalized version $\tilde{x}^{(t+1)}$ is equivalent (in distribution) to

$$\begin{aligned}
\tilde{x}^{(t+1)} &\overset{(d)}{=} B^{(t,1)} w^{(t)} + (y_1^{(t)})^2 a_1 \\
&= \sum_{i\in[t-1]} \frac{u^{(i+1)}(P_{\perp_{W^{[i-1]}}} w^{(i)})^\top}{\|P_{\perp_{W^{[i-1]}}} w^{(i)}\|^2} w^{(t)} + \sum_{i\in[t]} \frac{P_{\perp_{X^{[i-1]}}} x^{(i)}(v^{(i)})^\top}{\|P_{\perp_{X^{[i-1]}}} x^{(i)}\|^2} w^{(t)} \\
&\quad + B_{\text{res.}}^{(t,1)} w^{(t)} + (y_1^{(t)})^2 a_1 \\
&= \sum_{i\in[t-1]} \tilde{\Theta}\left(\frac{d^2}{k}\right) u^{(i+1)} \langle P_{\perp_{W^{[i-1]}}} w^{(i)}, w^{(t)}\rangle + \sum_{i\in[t]} \tilde{\Theta}(1) P_{\perp_{X^{[i-1]}}} x^{(i)} \langle v^{(i)}, w^{(t)}\rangle \\
&\quad + u^{(t+1)} + (y_1^{(t)})^2 a_1,
\end{aligned} \tag{28}$$

where form of $B^{(t,1)}$ in (16) is used in the second equality. The bounds on the norms come from Hypotheses 1 and 2. The last term is the definition of $u^{(t+1)} := B_{\text{res.}}^{(t,1)} w^{(t)}$. Note that differences in polylog factors in the (upper and lower) bounds in Hypotheses 1 and 2 are represented by notation $\tilde{\Theta}(\cdot)$.

The goal is to prove Hypothesis 1 holds at t-th iteration (which is to show the desired lower and upper bounds on $\|P_{\perp_{X^{[t]}}} x^{(t+1)}\|$) assuming induction hypothesis holds for earlier iterations. Given the normalization $x^{(t+1)} := \tilde{x}^{(t+1)}/\|\tilde{x}^{(t+1)}\|$ in each iteration, we have

$$\|P_{\perp_{X^{[t]}}} x^{(t+1)}\| = \frac{1}{\|\tilde{x}^{(t+1)}\|}\|P_{\perp_{X^{[t]}}} \tilde{x}^{(t+1)}\|. \tag{29}$$

Therefore, we first bound the norm of $\tilde{x}^{(t+1)}$ which turns out to be $\|\tilde{x}^{(t+1)}\| = \tilde{\Theta}\left(\frac{\sqrt{k}}{d}\right)$ as argued in the following.

Lower bound: The lower bound on $\|\tilde{x}^{(t+1)}\|$ simply follows from the term $u^{(t+1)}$, which is an independent random Gaussian.

Claim 4 *If $t \le \frac{d}{10}$, then we have whp*

$$\|\tilde{x}^{(t+1)}\| \ge \frac{\delta_t'}{4}\frac{\sqrt{k}}{d}.$$

Proof We have

$$\|\tilde{x}^{(t+1)}\| \ge \|P_{\mathrm{span}(X^{[t]}, U^{[t]}, a_1)^\perp} \tilde{x}^{(t+1)}\| = \|P_{\mathrm{span}(X^{[t]}, U^{[t]}, a_1)^\perp} u^{(t+1)}\|.$$

Note that the equality is concluded from expansion of $\tilde{x}^{(t+1)}$ in (28) where all the components of $\tilde{x}^{(t+1)}$ in the subspace $\mathrm{span}(X^{[t]}, U^{[t]}, a_1)^\perp$ is represented by $u^{(t+1)}$. The vector $P_{\mathrm{span}(X^{[t]}, U^{[t]}, a_1)^\perp} u^{(t+1)}$ is the projection of a random Gaussian vector $u^{(t+1)}$ in to a subspace of dimention $d - o(d)$. Hence it is still a random Gaussian vector with expected square norm larger than $\frac{\delta_t'^2}{2}\frac{k}{d^2}$. By Lemma 18, with high probability the desired bound holds. ■

Upper bound: The upper bound is argued in the following claim.

Claim 5 *We have either*

$$\langle x^{(t+1)}, a_1\rangle \geq 1-\gamma,$$

for some constant $\gamma > 0$ or

$$\|\tilde{x}^{(t+1)}\| \leq \tilde{O}\left(\frac{\sqrt{k}}{d}\right).$$

Proof Let $\tilde{x}^{(t+1)}$ in (28) be written as $\tilde{x}^{(t+1)} = z + (y_1^{(t)})^2 a_1$ where vector $z \in \mathbb{R}^d$ represents all the other terms in the expansion. The analysis is done under two cases 1) $(y_1^{(t)})^2 \geq \frac{2}{\gamma}\|z\|$ and 2) $(y_1^{(t)})^2 < \frac{2}{\gamma}\|z\|$ for some constant $\gamma > 0$. Note that the left hand side is the norm of $(y_1^{(t)})^2 a_1$ since $\|a_1\| = 1$, and in addition $(y_1^{(t)})^2 = \langle x^{(t)}, a_1\rangle^2$.

Case 1 $\left((y_1^{(t)})^2 \geq \frac{2}{\gamma}\|z\|\right)$: For the $x^{(t+1)} := \tilde{x}^{(t+1)}/\|\tilde{x}^{(t+1)}\|$, we have

$$\begin{aligned}\langle x^{(t+1)}, a_1\rangle &= \frac{1}{\|z + (y_1^{(t)})^2 a_1\|}\langle z + (y_1^{(t)})^2 a_1, a_1\rangle \\ &\geq \frac{1}{\|z\| + (y_1^{(t)})^2}\left[(y_1^{(t)})^2 - \|z\|\right] \\ &\geq \frac{1-\frac{\gamma}{2}}{1+\frac{\gamma}{2}} \geq 1-\gamma,\end{aligned}$$

where triangle and Cauchy-Schwartz inequality are used in the first bound, and the second inequality is concluded from assumption $(y_1^{(t)})^2 \geq \frac{2}{\gamma}\|z\|$.

Case 2 $\left((y_1^{(t)})^2 < \frac{2}{\gamma}\|z\|\right)$: We exploit the induction hypothesis to bound the norm of all the terms in the expansion of $\tilde{x}^{(t+1)}$ in (28).

For the terms involving $u^{(i+1)}, i \in [t]$, we have $\|u^{(i+1)}\| \leq 2\Delta_i'\frac{\sqrt{k}}{d}$ from Hypothesis 4 and the argument for $\|u^{(t+1)}\|$. In addition, for $u^{(i+1)}$, $i \in [t-1]$, the coefficient is bounded as

$$\frac{\langle P_{\perp_{W^{[i-1]}}} w^{(i)}, w^{(t)}\rangle}{\|P_{\perp_{W^{[i-1]}}} w^{(i)}\|^2} \leq \frac{\|w^{(t)}\|}{\|P_{\perp_{W^{[i-1]}}} w^{(i)}\|} \leq \frac{\Delta_t'}{\delta_i'}, \tag{30}$$

where Cauchy-Schwartz inequality is used in the first bound, and the bound in Hypothesis 2 and (27) are exploited in the last inequality. Therefore, the total contribution from terms involving $u^{(i+1)}$ in $\|\tilde{x}^{(t+1)}\|$ is bounded by $\frac{2(t-1)\Delta_t'^2}{\delta_t'}\frac{\sqrt{k}}{d}$.

For the terms involving $P_{\perp_{X^{[i-1]}}} x^{(i)}, i \in [t]$, we have $\|P_{\perp_{X^{[i-1]}}} x^{(i)}\| \leq 1$, but the coefficient $\langle v^{(i)}, w^{(t)}\rangle$ needs further analysis to be bounded which is done in Lemma 13 saying $|\langle v^{(i)}, w^{(t)}\rangle| \leq \tilde{O}\left(\frac{\sqrt{k}}{d}\right)$. This implies that the total contribution from terms involving $P_{\perp_{X^{[i-1]}}} x^{(i)}$ in $\|\tilde{x}^{(t+1)}\|$ is bounded by $\tilde{O}\left(\frac{\sqrt{k}}{d}\right)$.

Combining the above bounds and considering the assumption that the norm of $(y_1^{(t)})^2 a_1$ in the expansion of $\tilde{x}^{(t+1)}$ is dominated by the norm of other terms argued above, the proof is complete concluding that $\|\tilde{x}^{(t+1)}\| \leq \tilde{O}\left(\frac{\sqrt{k}}{d}\right)$. ■

Lemma 13 *Under the induction hypothesis (up to update step $\tilde{x}^{(t+1)} := A(y^{(t)})^{*2}$ at iteration t), we have for $i \in [t]$,*

$$|\langle v^{(i)}, w^{(t)}\rangle| \leq O\left(t^3\frac{(\Delta_{t-1}')^4}{(\delta_{t-1}')^4\delta_t^2}(\log d)\frac{\sqrt{k}}{d}\right) = \tilde{O}\left(\frac{\sqrt{k}}{d}\right).$$

Using (29) and the fact that $\|\tilde{x}^{(t+1)}\| = \tilde{\Theta}\left(\frac{\sqrt{k}}{d}\right)$, we have

$$\|P_{\perp_{X^{[t]}}} x^{(t+1)}\| \geq \tilde{\Theta}\left(\frac{d}{\sqrt{k}}\right)\|P_{\mathrm{span}(X^{[t]}, U^{[t]}, a_1)^\perp} u^{(t+1)}\| \geq \frac{\delta_t'}{4},$$

where the bound $\|P_{\mathrm{span}(X^{[t]}, U^{[t]}, a_1)^\perp} u^{(t+1)}\| \geq \frac{\delta_t'}{4}\frac{\sqrt{k}}{d}$ is also used. This finishes the proof that Hypothesis 1 holds.

B.2.4 Hypothesis 3

Finally we prove Hypothesis 3 at iteration t given earlier induction hypothesis. The first part of the hypothesis is proved in the following claim.

Claim 6 *We have*

$$|\langle a_1, x^{(t+1)}\rangle| \in [\delta_{t+1}^*, \Delta_{t+1}^*]d^{\beta 2^t}\frac{\sqrt{k}}{d}.$$

Proof We first show the correlation bound on the unnormalized version as $\langle a_1, \tilde{x}^{(t+1)}\rangle$. Looking at the expansion of $\tilde{x}^{(t+1)}$ in (28), the correlation $\langle a_1, \tilde{x}^{(t+1)}\rangle$ involves three types of terms emerging from $(y_1^{(t)})^2 a_1$, $u^{(i+1)}$ and $P_{\perp_{X^{(i-1)}}} x^{(i)}$. In the following, we argue the correlation from each of these terms where we observe that the correlation is dominated by the term $(y_1^{(t)})^2 a_1$, and the rest of terms contribute much smaller amount.

For the term $(y_1^{(t)})^2 a_1$, we have

$$\langle a_1, (y_1^{(t)})^2 a_1\rangle = (y_1^{(t)})^2 \in [(\delta_t^*)^2, (\Delta_t^*)^2]d^{\beta 2^t}\frac{k}{d^2},$$

where the last part exploits induction Hypothesis 3 in the previous iteration.

For the terms involving $u^{(i+1)}$, these vectors are random Gaussian vectors in a subspace (with dimension $\Omega(d)$), and therefore, we have with high probability

$$\langle a_1, u^{(i+1)}\rangle \leq \mathbb{E}[\|u^{(i+1)}\|]\cdot O\left(\frac{\log d}{\sqrt{d}}\right) \leq \tilde{O}\left(\frac{\sqrt{k}}{d\sqrt{d}}\right) \leq \tilde{O}\left(\frac{k}{d^2}\right),$$

where the correlation bound between two independent random Gaussian vectors in $\Omega(d)$-dimension is used in the first inequality,[12] Hypothesis 4 is exploited in the second inequality, and finally last inequality is from assumption $k > d$. In addition, the coefficient associated

12. For two independent random Gaussian vectors $p, q \in \mathbb{R}^d$, we have with high probability $\langle p, q\rangle \leq \mathbb{E}[\|p\|]\cdot\mathbb{E}[\|q\|]\cdot O\left(\frac{\log d}{\sqrt{d}}\right)$.

with $u^{(i+1)}$ is bounded by Δ_t'/δ_i' argued in (30). Hence, the total contribution from terms involving $u^{(i+1)}$ in $\langle \tilde{x}^{(t+1)}, a_1\rangle$ is bounded by $\tilde{O}\left(\frac{k}{d^2}\right)$.

For the terms involving $P_{\perp_{X^{[i-1]}}} x^{(i)}$, by Hypothesis 3 we have

$$\langle a_1, P_{\perp_{X^{[i-1]}}} x^{(i)}\rangle \leq \Delta_i^* d^{\beta 2^{i-1}} \frac{\sqrt{k}}{d}.$$

In addition, the associated coefficient is bounded by $\tilde{O}\left(\frac{\sqrt{k}}{d}\right)$ from Lemma 13. Hence, the total contribution from terms involving $P_{\perp_{X^{[i-1]}}} x^{(i)}$ in $\langle \tilde{x}^{(t+1)}, a_1\rangle$ is bounded by $\tilde{O}\left(d^{\beta 2^{t-1}} \frac{k}{d^2}\right)$.

Combining the above bounds implies

$$|\langle a_1, \tilde{x}^{(t+1)}\rangle| \leq \tilde{O}\left(d^{\beta 2^t} \frac{k}{d^2}\right).$$

Finally, using the bound on the norm of $\tilde{x}^{(t+1)}$ argued as $\|\tilde{x}^{(t+1)}\| = \tilde{\Theta}\left(\frac{\sqrt{k}}{d}\right)$ finishes the proof. ■

To prove the last part of Hypothesis 3, we use the following lemma which is very similar to Lemma 12.

Lemma 14 *Suppose R is a subspace in $\mathbb{R}^d$ of dimension t', such that there is a basis $\{r_1, \ldots, r_{t'}\}$ with $|\langle r_i, a_1\rangle| \leq \Delta$ and $\|r_i\| = 1$. Let $p \in \mathbb{R}^d$ be an arbitrary vector, then*

$$|\langle P_{\perp_R} p, a_1\rangle| \leq |\langle p, a_1\rangle| + \|p\|\Delta\sqrt{t'}.$$

We apply this lemma with $R = \text{span}(X^{[t]})$, and the basis is $P_{\perp X^{[i-1]}} X^{(i)}/\|P_{\perp X^{[i-1]}} X^{(i)}\|$. By induction hypothesis Δ in the lemma is at most $\Delta_*^t d^{\beta 2^t}\sqrt{k}/d$, let $v = x^{(t+1)}$ then this gives the desired bound.

Let $R = \text{span}(X^{[t]})$. Then the vectors $P_{\perp_{X^{[i-1]}}} x^{(i)}/\|P_{\perp_{X^{[i-1]}}} x^{(i)}\|$, $i \in [t]$ form a basis for subspace R, and we know from Hypotheses 1 and 3 that the correlation between these basis vectors and a_1 is bounded by $\Delta := \Delta_t^* d^{\beta 2^{t-1}} \frac{\sqrt{k}}{d}$. Applying above lemma, we have

$$|\langle P_{\perp_{X^{[t]}}} x^{(t+1)}, a_1\rangle| \leq |\langle x^{(t+1)}, a_1\rangle| + \Delta\sqrt{t} \leq \Delta_{t+1}^* d^{\beta 2^t} \frac{\sqrt{k}}{d},$$

where the last inequality uses the first part of Hypothesis 3 proved earlier in this section. Note that Δ_{t+1}^* is a new polylog factor here.

B.3 Growth Rate of δ_t, δ_t', Δ_t', δ_t^*, Δ_t^*

We know that if the number of iterations t is a constant, then the δ and Δ parameters (i.e., δ_t, δ_t', Δ_t', δ_t^*, Δ_t^*) in the induction hypothesis are bounded by polylog factors of d. Here, we show that these parameters can be still bounded even when the number of steps is slightly larger than a constant. Let

$$R_t := \max\{1/\delta_t, \Delta_{t-1}'/\delta_{t-1}', \Delta_t^*/\delta_t^*\}.$$

We know $R_1 = 1$, and by the inductive step analysis we have the following polynomial recursion property.

Claim 7 $R_{t+1} = \text{poly}(R_t, t, \log d)$.

This claim follows from the proof of inductive step, where in every step the δ and Δ parameters are bounded by polynomial functions of previous δ's (Δ's), t, and $\log d$.

We now solve this recursion as follows.

Lemma 15 *Suppose $R_{t+1} \leq c_0 R_t^{c_1} t^{c_2} (\log d)^{c_3}$ where c_0, c_1, c_2, c_3 are positive constants, and we know $R_1 = 1$. Then*

$$R_t \leq (\log d)^{2^{c_4 t}},$$

for some constant $c_4 > 0$ depending on c_0, c_1, c_2, c_3.

Proof Without loss of generality assume $c_0 \geq 1$, $c_2 \geq 1$, $c_3 \geq 1$, and $R_1 \geq \log d$. Given these assumptions, we have $R_t \geq \max\{c_0, t, \log d\}$, for $t \geq 1$. Applying this to the assumption $R_{t+1} \leq c_0 R_t^{c_1} t^{c_2} (\log d)^{c_3}$, we have

$$R_{t+1} \leq R_t^{1+c_1+c_2+c_3}. \tag{31}$$

Pick some $q > 0$ such that $R_1 \leq (\log d)^{2^q}$, and pick some

$$c_4 \geq \max\{q, \log_2(1 + c_1 + c_2 + c_3)\}.$$

Now we prove the result by the induction argument. Since $c_4 \geq q$, the basis of induction holds for R_1. As the inductive step, suppose $R_t \leq (\log d)^{2^{c_4 t}}$. Applying this to (31), we have

$$R_{t+1} \leq (\log d)^{(1+c_1+c_2+c_3)2^{c_4 t}} \leq (\log d)^{2^{c_4(t+1)}},$$

where $2^{c_4} \geq (1 + c_1 + c_2 + c_3)$ is used in the last inequality. This finishes the inductive step and the result is proved. ■

Using the above bound, we show in the following corollary that the δ and Δ parameters in the induction hypothesis are bounded by polylog factors of d even if the number of steps t goes up to $c \log\log d$ for small enough constant c. In addition, we show that if $\beta \geq (\log d)^{-c_5}$ for some constant $c_5 > 0$, then the power method converges to a point $x^{(t)}$ which is constant close to the true component.

Corollary 16 *There exists a universal constant $c_5 > 0$ such that if*

$$\beta \geq (\log d)^{-c_5},$$

and the initial correlation is lower bounded by $d^\beta \frac{\sqrt{k}}{d}$ (see (13)), then with high probability the power method gets to a point that is constant close to the true component in $\Theta(\log\log d)$ number of steps.

Proof Pick the number of steps to be $t = (\log\log d)/2c_4$, where c_4 is the constant in Lemma 15. Then, from Lemma 15 we have

$$R_t \leq (\log d)^{\sqrt{\log d}} \leq o(d),$$

where the last inequality can be shown by taking the log of both sides. This says that the analysis of inductive step still holds after such number of iterations.

Finally, by progress bound in (20), we can see that if $\beta \geq (\log d)^{-c_5}$, then the power method converges to a point $x^{(t)}$ which is constant close to the true component. ■

Appendix C. Auxiliary Lemmas for Induction Argument

In this section we prove the lemmas used in arguing inductive step in Appendix B.2.

We first introduce the following lemma proposing a lower bound on the singular value of product of matrices.

Lemma 17 (Merikoski and Kumar 2004) *Let C and D be $k \times k$ matrices. If $1 \leq i \leq k$ and $1 \leq l \leq k - i + 1$, then*

$$\sigma_i(CD) \geq \sigma_{i+l-1}(C) \cdot \sigma_{k-l+1}(D),$$

where $\sigma_j(C)$ denotes the j-th singular value (in decreasing order) of matrix C.

C.1 Properties of Random Gaussian Vectors

We start with some basic properties of random Gaussian vectors. First as a simple fact, the norm of a random Gaussian vector is concentrated as follows which is proved via simple concentration inequalities.

Lemma 18 *Let $z \in \mathbb{R}^d$ be a random Gaussian vector with $\mathbb{E}[zz^\top] = \frac{1}{d} I$. Then we have with high probability $\frac{1}{2} \leq \|z\| \leq 2$.*

Next we show the ℓ_∞ norm of a Gaussian vector is small, even if it is projected on some subspace.

Lemma 19 *Let R be any linear subspace in $\mathbb{R}^d$ and $z \in \mathbb{R}^d$ be a random Gaussian vector with $\mathbb{E}[zz^T] = \frac{1}{d} I$. Then we have with high probability $\|P_{\perp_R} z\|_\infty \leq \frac{\log d}{\sqrt{d}}$.*

Proof Since $P_{\perp_R}$ is a projection matrix, in particular the norm of its columns is bounded by 1. Hence, each entry of $P_{\perp_R} z$ is a Gaussian random variable with variance bounded by $\frac{1}{d}$ implying that with high probability the absolute value of each coordinate is smaller than $\frac{\log d}{\sqrt{d}}$. Finally, the desired ℓ_∞ norm bound is argued by applying union bound. ■

We can also show that most of the entries are of size at least $\frac{1}{\sqrt{d}}$.

Lemma 20 *Let R be any linear subspace in $\mathbb{R}^d$ with dimension $t \leq \frac{d}{16(\log d)^2}$ and $z \in \mathbb{R}^d$ be a random Gaussian vector with $\mathbb{E}[zz^T] = \frac{1}{d} I$. Then we have with high probability at least $1/2$ of the entries $i \in [d]$ satisfy $|(P_{\perp_R} z)_i| \geq \frac{1}{4\sqrt{d}}$.*

Proof Since the entries of z are independent Gaussian random variables with standard deviation $\frac{1}{\sqrt{d}}$, we know with high probability at least $1/2$ of the entries have absolute value larger than $\frac{1}{2\sqrt{d}}$. On the other hand, $P_R z$ is also a random Gaussian vector with expected square norm bounded by

$$\mathbb{E}[\|P_R z\|^2] \leq \frac{\mathbb{E}[\|z\|^2]}{16(\log d)^2} = \frac{1}{16(\log d)^2},$$

where the assumption on the dimension of subspace R is used in the inequality. By Lemma 19 we know with high probability entries of $P_R z$ are bounded by $1/4\sqrt{d}$. Now $P_{\perp_R} z = z - P_R z$ must have at least $1/2$ of the entries with absolute value larger than $1/4\sqrt{d}$. ■

Using the above lemmas, we can prove Lemma 11.

Proof of Lemma 11 Let z, z' be two independent samples of z, and w, w' be the corresponding w vectors. We have

$$w - w' = (p + z) * (p + z) - (p + z') * (p + z') = (2p + z + z') * (z - z'). \tag{32}$$

By properties of Gaussian vectors, $z + z'$, $z - z'$ are two *independent* random Gaussian vectors in the subspace orthogonal to R each with expected square norm $2\mathbb{E}[\|z\|^2]$. We use $z_1 := z + z'$ and $z_2 := z - z'$ to denote these two random Gaussian vectors.

Next, we show that with high probability

$$\|P_{\perp_{R'}}(w - w')\| \geq \frac{\mathbb{E}[\|z\|^2]}{20\sqrt{k}}.$$

Note that this implies the result of lemma as follows. Suppose $\|P_{\perp_{R'}} w\| < \frac{1}{40}\mathbb{E}[\|z\|^2]/\sqrt{k}$ with probability δ. Since w' is an independent sample, with probability δ^2 this bound holds for both w and w'. When this happens, we have $\|P_{\perp_{R'}}(w - w')\| < \frac{1}{20}\mathbb{E}[\|z\|^2]/\sqrt{k}$ by triangle inequality. Since we showed δ^2 is negligible, δ is also negligible.

First we sample z_2. Let $R'' = \mathrm{span}(R', p * z_2)$. Then by expansion of $w - w'$ in (32), we have

$$\|P_{\perp_{R'}}(w - w')\| = \|P_{\perp_{R'}}\big(2(p * z_2) + (z_1 * z_2)\big)\| \geq \|P_{\perp_{R''}}(z_1 * z_2)\| = \|P_{\perp_{R''}} \mathrm{Diag}(z_2) P_{\perp_R} z_1\|, \tag{33}$$

where the inequality is concluded by ignoring the component along $p * z_2$ direction. The last equality is from[13] $u * v = \mathrm{Diag}(u) \cdot v$ (for two vectors u and v), and the assumption that $z_1 = z + z'$ is in the subspace orthogonal to R. For the matrix $P_{\perp_{R''}} \mathrm{Diag}(z_2) P_{\perp_R}$, we have[14]

$$\sigma_{k/4}\left(P_{\perp_{R''}} \mathrm{Diag}(z_2) P_{\perp_R}\right) \geq \sigma_{k/2}\left(\mathrm{Diag}(z_2)\right) \cdot \sigma_{7k/8}\left(P_{\perp_R}\right) \cdot \sigma_{7k/8}\left(P_{\perp_{R''}}\right) \geq \frac{\sqrt{\mathbb{E}[\|z\|^2]}}{4\sqrt{k}},$$

where the first inequality is from Lemma 17, and the last step is argued as follows. By Lemma 18, with high probability z_2 has square norm at least $\mathbb{E}[\|z_2\|^2]/2 = \mathbb{E}[\|z\|^2]$, and

13. For vector u, $\mathrm{Diag}(u)$ denotes the diagonal matrix with u as its main diagonal.
14. Recall that $\sigma_l(A)$ denotes the l-th singular value (in decreasing order) of matrix A.

therefore, by Lemma 20 at least $k/2$ of its entries have absolute value larger than $\frac{\sqrt{\mathbb{E}[\|z\|^2]}}{4\sqrt{k}}$. Therefore, we can restrict attention to the space spanned by the $k/4$ top singular vectors. In addition, within this subspace we have with high probability $\|z_1\|^2 \geq \mathbb{E}[\|z\|^2]/8$, and hence,

$$\|P_{\perp_{R''}}\mathrm{Diag}(z_2)P_{\perp_R}z_1\| \geq \frac{\mathbb{E}[\|z\|^2]}{20\sqrt{k}},$$

which finishes the proof by applying (33). ■

C.2 Properties of Projections

In this part we prove some basic properties of projections. Intuitively, if the whole subspace has small inner-product with some vector, then the projection of an arbitrary vector to the orthogonal subspace should not change the inner-product with that particular vector by too much. This is what we stated in Lemma 14 and prove it here.

Proof of Lemma 14 We have $P_{\perp_R}p = p - \sum_{i=1}^{t'}\langle p, r_i\rangle r_i$, and therefore

$$\begin{aligned}
|\langle P_{\perp_R}p, a_1\rangle| &\leq |\langle p, a_1\rangle| + \sum_{i=1}^{t'}|\langle p, r_i\rangle\langle a_1, r_i\rangle| \\
&\leq |\langle p, a_1\rangle| + \Delta\sum_{i=1}^{t'}|\langle p, r_i\rangle| \\
&\leq |\langle p, a_1\rangle| + \Delta\sqrt{t'\sum_{i=1}^{t'}\langle p, r_i\rangle^2} \\
&\leq |\langle p, a_1\rangle| + \Delta\|p\|\sqrt{t'}.
\end{aligned}$$

The first step is triangle inequality and the third is Cauchy-Schwartz. ■

Lemma 12 is very similar.

Proof of Lemma 12 This lemma essentially follows from Lemma 14, because ℓ_∞ norm is just the maximum inner-product to a basis vector. More specifically, the above lemma is applied for all $a_1 = e_j, j \in [k]$, where e_j denotes the j-th basis vector in $\mathbb{R}^k$. ■

C.3 Bounding Correlation Between v and w

We are only left with Lemma 13. The main difficulty in proving this lemma is that the later steps are dependent on the previous steps. In the proof we show the dependency is bounded and in fact we can treat them as independent.

Proof of Lemma 13 Recall $w^{(t)} = y_{-1}^{(t)} * y_{-1}^{(t)}$, and $y_{-1}^{(t)}$ is specified in (24). We now expand the Hadamard product in $w^{(t)}$ and bound all the resulting $O(t^2)$ terms.

The first type of terms has the form $\langle v^{(i)}, P_{\perp_{W^{[i_1-1]}}}w^{(i_1)} * P_{\perp_{W^{[i_2-1]}}}w^{(i_2)}\rangle$, which can be bounded as

$$\begin{aligned}
\langle v^{(i)}, P_{\perp_{W^{[i_1-1]}}}w^{(i_1)} * P_{\perp_{W^{[i_2-1]}}}w^{(i_2)}\rangle &\leq k\cdot\|v^{(i)}\|_\infty\cdot\|P_{\perp_{W^{[i_1-1]}}}w^{(i_1)} * P_{\perp_{W^{[i_2-1]}}}w^{(i_2)}\|_\infty \\
&\leq 2k\frac{\log d}{\sqrt{d}}\frac{(\Delta'_{t-1})^2}{d^2},
\end{aligned}$$

where $\|v^{(i)}\|_\infty$ is bounded by Lemma 19, and ℓ_∞ norm of other vector is bounded by induction Hypothesis 2. In addition, the corresponding coefficient is bounded by (see (26), and note that both $i_1, i_2 < t$)

$$\frac{4(\Delta'_{t-1})^2}{(\delta'_{t-1})^4}\frac{d^2}{k}.$$

Hence, the total contribution from such terms is bounded by

$$8t^2\frac{(\Delta'_{t-1})^4}{(\delta'_{t-1})^4}\frac{\log d}{\sqrt{d}}. \tag{34a}$$

The second type of terms has the form

$$\langle v^{(i)}, P_{\perp_{W^{[i_1-1]}}}w^{(i_1)} * v^{(i_2)}\rangle = \langle v^{(i)} * v^{(i_2)}, P_{\perp_{W^{[i_1-1]}}}w^{(i_1)}\rangle,$$

which can be bounded as

$$\|P_{\perp_{W^{[i_1-1]}}}w^{(i_1)}\|_\infty\cdot\|v^{(i)} * v^{(i_2)}\|_1 \leq \|P_{\perp_{W^{[i_1-1]}}}w^{(i_1)}\|_\infty\cdot\frac{\|v^{(i)}\|^2 + \|v^{(i_2)}\|^2}{2} \leq 4\Delta'_{t-1}\frac{k}{d^2},$$

where the last inequality is concluded from Hypotheses 2 and 4. In addition, the corresponding coefficient is bounded by (see (25) and (26), and note that both $i_1, i_2 < t$)

$$\frac{2\Delta'_{t-1}}{(\delta'_{t-1})^2\delta_{t-1}}\frac{d}{\sqrt{k}}.$$

Hence, the total contribution from such terms is bounded by

$$8t^2\frac{(\Delta'_{t-1})^2}{\delta_{t-1}(\delta'_{t-1})^2}\frac{\sqrt{k}}{d}. \tag{34b}$$

The third type of terms has the form $\langle v^{(i)}, v^{(i_1)} * v^{(i_2)}\rangle$, with coefficient bounded by $1/\delta_{t-1}^2$ (see (25)). For bounding these inner products, we need to use the fact that they are random Gaussian vectors, however the main difficulty is that they are correlated (if $i > j$, then the subspace that $v^{(i)}$ is in that depends on $v^{(j)}$). To resolve this difficulty, we treat $v^{(i)} \in \mathbb{R}^{k-1}$ as projection of $n^{(i)} \in \mathbb{R}^{k-1}$ into subspace orthogonal to $W^{[t-1]}$, where $n^{(i)}$'s are *independent* Gaussian vectors in the full $k-1$ dimensional space. Independent of the ordering of i, i_1, i_2, we have with high probability

$$\langle n^{(i)}, n^{(i_1)} * n^{(i_2)}\rangle \leq O\left(\frac{\sqrt{k}}{d\sqrt{d}}\right),$$

since it is a sum of $k-1$ independent mean-0 entries each with variance $\frac{1}{d^3}$. On the other hand, from Hypothesis 4, we have $\mathbb{E}[\|v^{(i)}\|^2] \le 4\frac{k}{d}$, and since vector $n^{(i)} - v^{(i)}$ is in the subspace $W^{[t-1]}$ with dimension t, we have

$$\mathbb{E}[\|n^{(i)} - v^{(i)}\|^2] \le O\left(\frac{t}{k}\right) \cdot \frac{4k}{d} = O\left(\frac{t}{d}\right),$$

and therefore, we have with high probability $\|n^{(i)} - v^{(i)}\| \le O(\sqrt{t/d})$ for all $i \in [t-1]$. Using this, the difference between $\langle n^{(i)}, n^{(i_1)} * n^{(i_2)}\rangle$ and $\langle v^{(i)}, v^{(i_1)} * v^{(i_2)}\rangle$ can be bounded as

$$|\langle n^{(i)}, n^{(i_1)} * n^{(i_2)}\rangle - \langle v^{(i)}, v^{(i_1)} * v^{(i_2)}\rangle| \le O\left((\log k) t \frac{\sqrt{k}}{d\sqrt{d}}\right),$$

where the right hand side is the bound on the dominant term in the expansion of difference as

$$\begin{aligned}|\langle n^{(i)}, (n^{(i_1)} - v^{(i_1)}) * (n^{(i_2)} - v^{(i_2)})\rangle| &\le \|n^{(i)}\| \cdot \|(n^{(i_1)} - v^{(i_1)}) * (n^{(i_2)} - v^{(i_2)})\| \\ &\le O\left((\log k)\sqrt{\frac{k}{d}}\right) \cdot O\left(\frac{t}{d}\right) \\ &= O\left((\log k) t \frac{\sqrt{k}}{d\sqrt{d}}\right).\end{aligned}$$

Here, the first inequality is the Cauchy-Schwartz, and the second inequality is from bound on the norm of random Gaussian vector $n^{(i)}$, and the bound on the norm of difference vectors $n^{(i_1)} - v^{(i_1)}$ stated earlier. Hence, the total contribution from such terms is bounded by

$$O\left(t^3 \frac{\log k}{\delta_{t-1}^2} \frac{\sqrt{k}}{d\sqrt{d}}\right). \tag{34c}$$

Taking the sum of all the terms in (34a)-(34c) gives the desired bound. ■

Appendix D. Additional Arguments for Noise Analysis

We first formally define norm $\|\cdot\|_*$ which was discussed in proof outline in Section 3.2.

Definition 21 (Norm $\|\cdot\|_*$) *Given a matrix $A = [a_1\ a_2\ \cdots\ a_k] \in \mathbb{R}^{d\times k}$, for any vector $u \in \mathbb{R}^d$, the norm $\|u\|_{A^*}$ is defined as*

$$\|u\|_{A^*} = \max_{i\in[k]} |\langle a_i, u\rangle|.$$

This norm satisfies a property shown in Lemma 22 which enables us to argue that $\xi^{(t)}$ is small enough as stated in Lemma 10.

We now provide the proof of main lemma for noise analysis.

Proof of Lemma 10 We prove this by an induction argument.

Basis of induction: For $t = 1$, $x^{(1)}$ is the initialization vector and thus, $\xi^{(1)} = 0$. Hence, the proposed bound holds for the basis of induction $t = 1$.

Inductive step: Assuming the inductive hypothesis holds for step t, we prove it also holds for step $t+1$. We have

$$\begin{aligned}x^{(t+1)} + \xi^{(t+1)} &= \text{Norm}\left(\hat{T}(x^{(t)} + \xi^{(t)}, x^{(t)} + \xi^{(t)}, I)\right) \\ &= \text{Norm}\left(T(x^{(t)}, x^{(t)}, I) + 2T(x^{(t)}, \xi^{(t)}, I) + T(\xi^{(t)}, \xi^{(t)}, I) + E(\hat{x}^{(t)}, \hat{x}^{(t)}, I)\right).\end{aligned} \tag{35}$$

The first term $T(x^{(t)}, x^{(t)}, I)$ corresponds to main signal; recall that $x^{(t+1)} = \text{Norm}(T(x^{(t)}, x^{(t)}, I))$ in the noiseless setting, where the unnormalized version $\tilde{x}^{(t+1)} := T(x^{(t)}, x^{(t)}, I)$ has norm at least $\tilde{\Omega}(\sqrt{k}/d)$ which is argued in the induction argument for Hypothesis 1. We now bound the desired property of noise terms in the above expansion.

For the second term, we break it into two terms as

$$2T(x^{(t)}, \xi^{(t)}, I) = 2\langle x^{(t)}, a_1\rangle\langle \xi^{(t)}, a_1\rangle a_1 + 2T'(x^{(t)}, \xi^{(t)}, I) =: p + q,$$

where $T' := \sum_{j>1} a_j \otimes a_j \otimes a_j$. Here $p := 2\langle x^{(t)}, a_1\rangle\langle \xi^{(t)}, a_1\rangle a_1$ corresponds to the multilinear form from first component of T, and $q := 2T'(x^{(t)}, \xi^{(t)}, I)$ corresponds to the multilinear form from the rest of components.

For q, we apply Lemma 22. Note that since $\|x^{(t)}\|_{B^*} \le \tilde{O}(1/\sqrt{d})$, we get an extra $1/\sqrt{d}$ factor in the bound provided by Lemma 22, and therefore we have

$$\|q\|_2 \le \tilde{O}(\epsilon d^{\beta 2^{t-1}} \sqrt{k}/d),$$

where we also used the induction hypothesis $\|\xi^{(t)}\| \le \tilde{O}(\epsilon d^{\beta 2^{t-1}})$.

For p, we have

$$\|p\| = 2|\langle x^{(t)}, a_1\rangle| \cdot |\langle \xi^{(t)}, a_1\rangle| \le \tilde{O}\left(\epsilon d^{\beta 2^t} \sqrt{k}/d\right),$$

where the inequality is from the signal and noise induction hypotheses; see Equation (20) for the signal induction hypothesis.

The third term $T(\xi^{(t)}, \xi^{(t)}, I)$ has ℓ_2 norm bounded as

$$\|T(\xi^{(t)}, \xi^{(t)}, I)\| \le \|T\|\|\xi^{(t)}\|^2 \le \tilde{O}(d^{\beta 2^t}\epsilon^2) \le \tilde{O}(\epsilon d^{\beta 2^t}\sqrt{k}/d),$$

where the first inequality uses the sub-multiplicative property, and the second inequality exploits the bounded norm of random tensor T as $\|T\| \le O(1)$, and the induction hypothesis in t-th step. The final inequality uses the assumption $\epsilon < o(\sqrt{k}/d)$ in the lemma.

The fourth term $E(\hat{x}^{(t)}, \hat{x}^{(t)}, I)$ has ℓ_2 norm bounded by

$$\|E(\hat{x}^{(t)}, \hat{x}^{(t)}, I)\| \le \|E\|\|\hat{x}^{(t)}\|^2 \le \epsilon\sqrt{k}/d,$$

where we use the sub-multiplicative property in the first inequality, and the assumption on the norm of error tensor E in the lemma, and the fact that $\|\hat{x}^{(t)}\| = 1$ are exploited in the second inequality.

Summarizing the above calculations on different terms of the update in (35), the signal plus noise vector before normalization is

$$T(x^{(t)}, x^{(t)}, I) + 2T(x^{(t)}, \xi^{(t)}, I) + T(\xi^{(t)}, \xi^{(t)}, I) + E(\hat{x}^{(t)}, \hat{x}^{(t)}, I) =: \alpha x^{(t+1)} + z,$$

where α is a coefficient which is lower bounded as $\alpha \geq \tilde{\Omega}(\sqrt{k}/d)$. The vector z also satisfies

$$\|z\| \leq \tilde{O}(\epsilon d^{\beta 2^t} \sqrt{k}/d), \tag{36}$$

which is derived by combining the bounds we argued on the second, third and fourth terms.

Note that until the very last step we always have $d^{\beta 2^t} \leq o(d/\sqrt{k})$ (otherwise we are constantly close to the true component, and we are done). In this case the norm of z is negligible compared to α since $\|z\| \leq o(\alpha)$, and thus, the normalization factor is equal to $\|\alpha x^{(t+1)} + z\| = \alpha(1 \pm o(1))$. Therefore, after the normalization, we have the noise vector $\xi^{(t+1)} = \alpha' x^{(t+1)} + \beta z$, where $|\alpha'| \leq \|z\|/\alpha \leq o(1)$ and $|\beta| \leq 2/\alpha \leq \tilde{O}(d/\sqrt{k})$, hence we know $\|\xi^{(t+1)}\| \leq \tilde{O}(\epsilon d^{\beta 2^t})$.

For the last step of the induction, the norm of $T(x^{(t)}, x^{(t)}, I)$ is also larger (it has norm $d^{\beta 2^t} k/d^2$, which is larger than $\sqrt{k}/d$ for the last step). Since $\epsilon < o(\sqrt{k}/d)$ we still know the noise is negligible.

■

Lemma on the property of $\|\cdot\|_*$ norm defined in Definition 21:

Lemma 22 *Consider a random tensor $T = \sum_{j\in[k]} a_j \otimes a_j \otimes a_j$ where a_j's are zero-mean random Gaussian with expected unit norm. Let $T' = \sum_{j>1} a_j \otimes a_j \otimes a_j$ and $B := [a_2, a_3, \ldots, a_k] \in \mathbb{R}^{d\times(k-1)}$. Then with high probability, for any vectors u, v such that $\|u\|_{B^*} \leq 1$ and $\|v\|_2 \leq 1$, we have*

$$\|T'(u, v, I)\|_2 \leq \tilde{O}\left(\sqrt{k/d}\right).$$

Proof We prove this lemma along similar ideas provided in the proof of Anandkumar et al. (2015, Claim 5) (Anandkumar et al. (2014b, Claim 1)). Let $A := [a_1, \ldots, a_k] \in \mathbb{R}^{d\times k}$, and η_j's be independent random ± 1 variables with $\Pr[\eta_j = 1] = 1/2$. We rewrite tensor T' as

$$T' = \sum_{j>1} \eta_j a_j \otimes a_j \otimes a_j. \tag{37}$$

Since a_j's are zero-mean random Gaussian vectors, we have $\eta_j a_j \sim a_j$, and thus, the new T' has the same distribution as the original one. We now first sample vectors a_j's, and this already makes the norm $\|\cdot\|_{B*}$ well-defined. In addition, the value of η_j's does not change the singular values of A or B. Also note that since a_j's are zero-mean random Gaussian vectors with expected norm 1, they also satisfy with high probability the incoherence condition such that $|\langle a_i, a_j\rangle| \leq \tilde{O}(1/\sqrt{d})$ for all $i \neq j$. Thus, we condition on all these fixed events, and the only remaining random variables are just the η_j's.

The proposed statement in the lemma is equivalent to bounding

$$\sup_{\|u\|_{B^*}=1, \|v\|=\|w\|=1} \left|T'(u, v, w)\right|.$$

In order to bound it, we provide an ϵ-net argument. We construct an ϵ-net such that for any vector $u \in \mathbb{R}^d$ with unit $\|\cdot\|_{B^*}$ norm, there is a vector u' in the net such that $\|B^\top(u - u')\| \leq 1/k^2$. We also construct standard ε-net for vectors $u, w \in \mathbb{R}^d$ with unit ℓ_2 norm. By standard construction, this ε-net has size $\exp(O(d \log d))$. We now show that for all u in ϵ-net with unit $\|\cdot\|_{B*}$ norm, and all v, w in ϵ-net with unit ℓ_2 norm, the desired bound $|T'(u, v, w)| \leq \tilde{O}\left(\sqrt{k/d}\right)$ holds with high probability. Then for the other vectors (u, v, w) not in the ε-net, the result follows from their closest points in the net.

Now for a fixed triple (u, v, w) in the ε-net, we have

$$T'(u, v, w) = \sum_{j>1} \eta_j \langle u, a_j\rangle\langle v, a_j\rangle\langle w, a_j\rangle,$$

which is a sum of independent random variables; recall that the randomness is from η_j's, and a_j's are already sampled and thus they are fixed here. We partition the above sum into *large* and *small* terms as $T'(u, v, w) = S_L + S_{L^c}$ such that the summation S_L is the sum of large terms including terms in set

$$L := \left\{j \in \{2, 3, \ldots, k\} : |\langle v, a_j\rangle| \geq \log d/\sqrt{d} \vee |\langle w, a_j\rangle| \geq \log d/\sqrt{d}\right\},$$

and the rest are the small terms forming S_{L^c}. Note that $|\langle u, a_j\rangle| \leq 1$ since $\|u\|_{B*} = 1$.

Bounding $|S_{L^c}|$: Since the variables are bounded in this summation corresponding to small terms, we use Bernstein's inequality, and thus with probability at least $1 - \delta$, we have $|S_{L^c}| \leq \frac{\sqrt{k \log 1/\delta}\cdot \text{polylog}\, d}{d}$ for the fixed point in the ε-net. By choosing small enough $\delta = \exp(-Cd \log d)$ (where C is large enough constant), we can apply the union bound on the ε-net, and conclude that for all the vectors in the net, $|S_{L^c}|$ is smaller than $\tilde{O}(\sqrt{k/d})$ with high probability.

Bounding $|S_L|$: Since the columns of matrix B are random Gaussian vectors, it satisfies the RIP property with high probability (see Remark 3 in Anandkumar et al. (2014b) for the precise definition of RIP), and thus by the definition of RIP and Lemma 3 in Anandkumar et al. (2014b), we have $\|B_L\| \leq 2$ where B_L is the sub-columns of matrix B specified by set L.

We now have

$$|S_L| \leq \sum_{j\in L} |\langle u, a_j\rangle| \cdot |\langle v, a_j\rangle| \cdot |\langle w, a_j\rangle| \leq \sum_{j\in L} |\langle v, a_j\rangle| \cdot |\langle w, a_j\rangle| \leq \left\|B_L^\top v\right\| \cdot \left\|B_L^\top w\right\| \leq 4,$$

where the second step uses the fact that $|\langle u, a_j\rangle| \leq 1$, the third step exploits Cauchy-Schwartz inequality, and the last step uses bound $\|B_L\| \leq 2$. Notice that matrix B is already sampled before we do the ε-net argument, and therefore, we do not need to do union bound over all u, v, w for this event.

Since we assume the overcomplete regime $k \geq d$, the bound on $|S_{L^c}|$ is dominant which finishes the proof.

■

References

A. Anandkumar, D. Hsu, and S. M. Kakade. A Method of Moments for Mixture Models and Hidden Markov Models. In *Proc. of Conf. on Learning Theory*, June 2012.

A. Anandkumar, R. Ge, and M. Janzamin. Learning Overcomplete Latent Variable Models through Tensor Methods. In *Proceedings of the Conference on Learning Theory (COLT)*, Paris, France, July 2015.

Anima Anandkumar, Rong Ge, and Majid Janzamin. Guaranteed Non-Orthogonal Tensor Decomposition via Alternating Rank-1 Updates. *arXiv preprint arXiv:1402.5180*, Feb. 2014a.

Anima Anandkumar, Rong Ge, and Majid Janzamin. Sample Complexity Analysis for Learning Overcomplete Latent Variable Models through Tensor Methods. *arXiv preprint arXiv:1408.0553*, Aug. 2014b.

Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M. Kakade, and Matus Telgarsky. Tensor Decompositions for Learning Latent Variable Models. *Journal of Machine Learning Research*, 15:2773–2832, 2014c.

Sanjeev Arora and Ravi Kannan. Learning mixtures of separated nonspherical gaussians. *The Annals of Applied Probability*, 15(1A):69–92, 2005.

Mohsen Bayati and Andrea Montanari. The dynamics of message passing on dense graphs, with applications to compressed sensing. *arXiv preprint arXiv:1001.3448*, Jan. 2010.

A. Bhaskara, M. Charikar, A. Moitra, and A. Vijayaraghavan. Smoothed analysis of tensor decompositions. *arXiv preprint arXiv:1311.3651*, 2013.

Emmanuel J Candes and Terence Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *Information Theory, IEEE Transactions on*, 52 (12):5406–5425, 2006.

Dustin Cartwright and Bernd Sturmfels. The number of eigenvalues of a tensor. *Linear Algebra and its Applications*, 438(2):942–952, January 2013.

Adam Coates, Andrew Y Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics*, pages 215–223, 2011.

Sanjoy Dasgupta. Learning mixutres of gaussians. In *FOCS*, 1999.

L. De Lathauwer, J. Castaing, and J.-F. Cardoso. Fourth-order cumulant-based blind identification of underdetermined mixtures. *Signal Processing, IEEE Transactions on*, 55(6):2965–2973, 2007.

Rong Ge and Tengyu Ma. Decomposing overcomplete 3rd order tensors using sum-of-squares algorithms. *arXiv preprint arXiv:1504.05287*, April 2015.

N. Goyal, S. Vempala, and Y. Xiao. Fourier pca. *arXiv preprint arXiv:1306.5825*, 2013.

Christopher J. Hillar and Lek-Heng Lim. Most tensor problems are NP hard. *Journal of ACM*, 60(6), November 2013.

Roger A. Horn and Charles R. Johnson. *Matrix analysis*. Cambridge university press, 2012.

D. Hsu and S. M. Kakade. Learning Mixtures of Spherical Gaussians: Moment Methods and Spectral Decompositions. *arXiv preprint arXiv:1206.5766*, 2012.

Daniel Hsu and Sham M Kakade. Learning mixtures of spherical gaussians: moment methods and spectral decompositions. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 11–20. ACM, 2013.

A. T. Kalai, A. Moitra, and G. Valiant. Efficiently learning mixtures of two gaussians. In *STOC*, 2010.

M. S. Lewicki and T. J. Sejnowski. Learning overcomplete representations. *Neural computation*, 12(2):337–365, 2000.

Jorma K. Merikoski and Ravinder Kumar. Inequalities for spreads of matrix sums and products. *Applied Mathematics E-Notes*, 4:150–159, 2004.

A. Moitra and G. Valiant. Settling the polynomial learnability of mixtures of gaussians. In *FOCS*, 2010.

Karl Pearson. Contributions to the mathematical theory of evolution. *Philosophical Transactions of the Royal Society of London. A*, 186:343–414, 1895.

S. Vempala and G. Wang. A spectral algorithm for learning mixtures of distributions. In *FOCS*, 2002.

T. Zhang and G. Golub. Rank-one approximation to high order tensors. *SIAM Journal on Matrix Analysis and Applications*, 23:534–550, 2001.

Journal of Machine Learning Research 18 (2017) 1-5 Submitted 3/16; Revised 3/17; Published 4/17

JSAT: Java Statistical Analysis Tool, a Library for Machine Learning

Edward Raff raff.edward@gmail.com
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County

Editor: Geoff Holmes

Abstract

Java **S**tatistical **A**nalysis **T**ool (JSAT) is a Machine Learning library written in pure Java. It works to fill a void in the Java ecosystem for a general purpose library that is relatively high performance and flexible, which is not adequately fulfilled by Weka (Hall et al., 2009) and Java-ML (Abeel et al., 2009). Almost all of the algorithms are independently implemented using an Object-Oriented framework. JSAT is made available under the GNU GPL license here: `https://github.com/EdwardRaff/JSAT`.

Keywords: java, machine learning, open source, java library, machine learning software.

1. Introduction

There exist relatively few general purpose Machine Learning libraries for use in the Java ecosystem. Weka is one of the only such libraries in this domain, and one of the oldest and most used in general. It seems that more modern and efficient libraries, such as Scikit-learn (Pedregosa et al., 2011), have been developed in every language but Java. While a number of distributed libraries, such as MLlib (Meng et al., 2016), exist in the Java ecosystem, their focus precludes the inclusion of many algorithms that do not scale out. These tools may add needless overhead and complexity for datasets and tasks that can be solved with a single machine. This leaves a void in the Java ecosystem for a tool that is both easy for developers, relatively fast, and allows more flexibility for researchers to develop and compare new algorithms.

JSAT attempts to fill this void. It is written in Java 6 and has no dependencies, making it easy to integrate into any Java project without conflict. Many common "bread and butter" algorithms, such as k-means, Random Forest, and SVMs are implemented with multiple different solvers. In every case one or more classical implementations is included, such as Platt's SMO algorithm for SVMs, that can be used as a baseline for correctness and comparison for researchers developing new solutions and to ensure performance is respectable compared to common baselines. Popular and more efficient algorithms, such as the algorithms for Linear-SVM and L1 regularized Logistic-Regression used in LIBLINEAR (Fan et al., 2008) are provided—as well as less common but advantageous solutions, such as accelerated k-means algorithms. JSAT is not limited to exact solvers like SMO, and includes many faster approximations such as the Projectron and BSGD.

Collectively, JSAT includes numerous algorithms for classification and regression (over 70), clustering (18), feature selection and engineering (over 20), visualization (5), and the tools for implementing them. These counts do not include cases where many algorithms are covered by a single class. For example, the `NNChainHAC` class provides $O(n^2)$ hierarchical clustering for any Lance Williams dissimilarity. This also includes a number of algorithms that are useful but not widely available—be it in Java or any other language. Some examples include a multi-class generalization of the popular Passive Aggressive classifier and the Adaptive Multi-Hyperplane Machine (AMM). Almost all code in JSAT has been implemented independently by the author based on the original papers, rather than porting other implementations. The three primary interfaces in JSAT are `Classifier`, `Regressor`, and `Clusterer` for the three primary tasks JSAT supports. Each interface defines the API for learning and inference, with extension interfaces for specializes abilities, such as an `UpdateableClassifier` for online learning. JSAT also has interfaces related to warm starting, change detection, text processing, building/querying metric spaces, loss functions, and optimization. JSAT comes out to about 202k lines of code, compared to 707k for Weka and just under 190k lines for Scikit-learn.[1]

2. Features

JSAT has a number of features in its implementation and design to try and balance ease of use for users, and the development of new algorithms by researchers. Both of these tasks benefit from a certain level of performance in terms of run-time, which JSAT strives to achieve for all algorithms. A small benchmark on MNIST is given in Table 1, where JSAT is compared against the same algorithm in other libraries, with JSAT adjusted to match the default parameters used by others. Error rate was measured from one run on the standard training and testing split of MNIST. JSAT's performance is generally better than Weka and BudgetedSVM (Djuric et al., 2013), and more mixed when compared to LIBLINEAR. That the latter two are written in C/C++ is important to demonstrate that Java code can perform on a similar level while offering the benefits of the Java platform and ecosystem. JSAT's large collection of algorithms also allows for better efficiency by selecting the most appropriate algorithm. For example, when using the accelerated k-means algorithm from Elkan (2003), JSAT becomes 170 times faster with respect to Weka's implementation. All of these numbers are for single-threaded executions, and JSAT supports multi-threaded execution of many algorithms, more so than Weka and many other libraries.

	Weka						*LIBLINEAR*		*BudgetedSVM*	
	Platt SMO	C45	RF	1-NN	LR	Lloyd's k-means	SVM by DCD	newGLMNET	AMM	BSGD
Other Time	7904	303	143	2537	3301	1011	161	14.5	59.5	160
JSAT Time	973	139	125	691	914	36	71	29.2	9.7	64
Other Error	1.55%	11.1%	3.26%	3.09%	8.21%	—	8.30%	8.35%	37.2%	21.8%
JSAT Error	1.56%	11.5%	4.19%	3.09%	7.76%	—	9.21%	8.53%	5.02%	10.5%

Table 1: Training time (in seconds) on MNIST

2.1 For Researchers

JSAT includes a wide breadth of algorithms, allowing for comparison against many other approaches in a unified framework. Comparisons also become more meaningful as all al-

1. Excluding comments and blanks lines, based on `https://www.openhub.net`

©2017 Edward Raff.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v18/16-131.html`.

gorithms are on an equal footing of speed, avoiding the need for alternative comparison metrics when using multiple libraries. Compared to Python, developing in Java can be a benefit for any algorithm that is not easy to implement in a vectorized approach, which is necessary for performance in most interpreted languages.

For high level implementations, JSAT includes the L-BFGS and OWL-QN solvers, requiring only the implementation of the desired objective function and its gradient to be used. It's also common to implement one's algorithm via SGD for efficiency, and JSAT provides common variants such as Nestrov Momentum SGD, Adam, and RMSprop.

When using lower level linear algebra constructs, JSAT continues to use an Object-Oriented approach. This allows for simple and common tricks to be added to an implementation by changing only the initialization of the object. For example the `ScaledVector` object allows one to implement the common trick for updating a vector by a scalar in constant time presented in Shalev-Shwartz et al. (2007). This works with both sparse and dense vectors. Similar objects exist for a vector shifted by a constant, keeping track of the 2-norm of a vector, or implicitly constructing the degree-2 polynomial expansion of the vector.

We also borrow the concept of the `kcentroid` from Dlib (King, 2009), called `KernelPoint` in JSAT. This object represents the result of performing math in the space induced by using the kernel trick. The object itself approximates the result with a compact solution by projecting new vectors to a basis set. JSAT extends it to support multiple strategies, such as the specialized merged result for the RBF kernel from Wang et al. (2010). This allows for a single piece of code to switch between multiple different approximation methods for implementing a kernelized algorithm. A `KernelPoints` further extends the concept to multiple kernelized weight vectors backed by a single set of basis vectors, which is useful for multi-class algorithms.

At the lowest level, JSAT includes interfaces for continuous and discrete probability distributions, and implementations of more specialized math functions such as the digamma function. Faster, but less accurate, implementations of common math functions are provided as well so authors may optimize their algorithms further. Useful tools are included at this level, such as a generic object for implementing functions via continued fractions.

2.2 For Developers / Users

Building Machine Learning-based solutions often revolves around parameter tuning and trying multiple different algorithms to see which obtains the best performance. Based on this fact, JSAT attempts to make this process simpler for both novices as well as experts. All data in JSAT is represented with a `DataSet` class, of which specialized instances exist for label-less, classification, and regression datasets—making the goal explicit by the type system. The creation of test and validation sets are simply one function call, for example `List<DataSet> splits = dataset.randomSplit(0.7, 0.2, 0.1)` will split a dataset into 3 subsets, with 70% for training, 20% for validation, and 10% for testing. Any number of ratios could be given, and it would return as many sets. It can also be used for selecting a random subset by passing in just one value less than 1.

Explicit classification and regression model evaluation objects exist to either use preselected test sets or perform cross validation. Each support the use of an arbitrary number of evaluation metrics for their respective types, which can be passed in as objects. This includes common metrics such as AUC, F_β score, Precision, Recall, and others. These metrics, as well as many of the algorithms in JSAT, support weighted data instances.

```
ClassificationDataSet dataset = LIBSVMLoader.loadC(new
    File("diabetes.libsvm"));
List<ClassificationDataSet> splits = dataset.randomSplit(0.75, 0.25);
ClassificationDataSet train = splits.get(0), test = splits.get(1);
PlattSMO model = new PlattSMO(new RBFKernel());//have data, now pick model
RandomSearch search = new RandomSearch((Classifier)model, 3);
search.autoAddParameters(train); //automatically find parameters to tune
search.trainC(train);//build model & tune parameters
ClassificationModelEvaluation cme = new
    ClassificationModelEvaluation(search.getTrainedClassifier(), train);
cme.evaluateTestSet(test);//evaluate tuned model
System.out.println("Tuned Error rate: " + cme.getErrorRate());
```

Listing 1: Loading a dataset and performing a parameter search in JSAT

Built upon these is a framework for model evaluation, currently supporting the classic Grid-Search approach and the Random-Search method (Bergstra and Bengio, 2012). Setting the search parameters and their ranges is both cumbersome from a coding standpoint, and difficult for novice users. To alleviate this issue, every object with tunable parameters in JSAT may specify a distribution for each parameter it wants included in the search. This is done with respect to an input dataset, and the methods are discovered through Java's reflection mechanism. This allows JSAT to automatically populate the search parameters with reasonable values, even when the user may not have properly normalized their data. An abridged example of this is provided in Listing 1. In this example, the RBF kernel will automatically be searched over a wider space since the data has not been normalized into the $[-1, 1]$ range, a common issue overlooked by beginners (Hsu et al., 2003). It also has the benefit of being concise, and does not need to be changed when the algorithm used changes. When desired, the user can manually specify the searched parameters and values.

The GridSearch object also has additional logic for models that can be warm-started in their training. The warm-start interface indicates the main parameter that warm-starts should be based off of—automatically making use of this functionality for faster training. This is also helpful for building regularization paths, as is common with L_1 regularized models (Schmidt et al., 2009). This is an improvement over many packages that support warm-starting for some models, but require explicit cross validation objects for each algorithm.

3. Conclusion and Future Plans

JSAT has a wide breadth in the algorithms and tools it makes available to users and researchers, and fills a vital performance need while maintaining a relatively small code base. Future development goals are to further refine the API for consistency and ease of use. A move to Java 8 is planned, and will be used to perform more significant refactoring.

References

T Abeel, Y Peer, and Y Saeys. Java-ML: A Machine Learning Library. *Journal of Machine Learning Research*, 10:931–934, 2009. ISSN 15324435.

J Bergstra and Y Bengio. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, 13:281–305, 2012. ISSN 1532-4435.

N Djuric, L Lan, S Vucetic, and Z Wang. BudgetedSVM: A Toolbox for Scalable SVM Approximations. *Journal of Machine Learning Research*, 14:3813–3817, 2013.

C Elkan. Using the Triangle Inequality to Accelerate k-Means. In *International Conference on Machine Learning (ICML)*, pages 147–153. AAAI Press, 2003.

R Fan, K Chang, C Hsieh, X Wang, and C Lin. LIBLINEAR: A Library for Large Linear Classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

M Hall, E Frank, G Holmes, B Pfahringer, P Reutemann, and I Witten. The WEKA Data Mining Software: An Update Mark. *ACM SIGKDD Explorations Newsletter*, 11 (1):10–18, nov 2009. ISSN 19310145. doi: 10.1145/1656274.1656278.

C Hsu, C Chang, and C Lin. A Practical Guide to Support Vector Classification. Technical report, National Taiwan University, 2003.

D. E. King. Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009. ISSN 15324435. doi: 10.1145/1577069.1755843.

X Meng, J Bradley, B Yavuz, E Sparks, S Venkataraman, D Liu, J Freeman, DB Tsai, M Amde, S Owen, D Xin, R Xin, M J. Franklin, R Zadeh, M Zaharia, and A Talwalkar. Mllib: Machine learning in apache spark. *Journal of Machine Learning Research*, 17(34): 1–7, 2016. URL `http://jmlr.org/papers/v17/15-237.html`.

F Pedregosa, G Varoquaux, A Gramfort, V Michel, B Thirion, O Grisel, M Blondel, P Prettenhofer, R Weiss, V Dubourg, J Vanderplas, A Passos, D Cournapeau, M Brucher, M Perrot, and E Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

M Schmidt, G Fung, and R Rosaless. Optimization Methods for L1-Regularization. Technical report, University of British Columbia, 2009.

S Shalev-Shwartz, Y Singer, and N Srebro. Pegasos : Primal Estimated sub-GrAdient SOlver for SVM. In *International Conference on Machine Learning (ICML)*, pages 807–814, New York, NY, 2007. ACM. doi: 10.1145/1273496.1273598.

Z Wang, K Crammer, and S Vucetic. Multi-class pegasos on a budget. In *International Conference on Machine Learning (ICML)*, pages 1143–1150, 2010.

Journal of Machine Learning Research 18 (2017) 1-29 Submitted 4/16; Revised 1/17; Published 4/17

Identifying a Minimal Class of Models for High–dimensional Data

Daniel Nevo DANIELNEVO@GMAIL.COM
Department of Statistics
The Hebrew University of Jerusalem
Mt. Scopus, Jerusalem, Israel
and
Current address: Departments of Biostatistics and Epidemiology
Harvard T.H. Chan School of Public Health
Boston, MA 02115, USA

Ya'acov Ritov YAACOV.RITOV@GMAIL.COM
Department of Statistics
The Hebrew University of Jerusalem
Mt. Scopus, Jerusalem, Israel
and
Department of Statistics
University of Michigan
Ann Arbor, MI 48109–1107, USA

Editor: Nicolai Meinshausen

Abstract

Model selection consistency in the high–dimensional regression setting can be achieved only if strong assumptions are fulfilled. We therefore suggest to pursue a different goal, which we call a minimal class of models. The minimal class of models includes models that are similar in their prediction accuracy but not necessarily in their elements. We suggest a random search algorithm to reveal candidate models. The algorithm implements simulated annealing while using a score for each predictor that we suggest to derive using a combination of the lasso and the elastic net. The utility of using a minimal class of models is demonstrated in the analysis of two data sets.

Keywords. Model Selection; High–dimensional Data; Lasso; Elastic Net; Simulated Annealing

1. Introduction

High–dimensional statistical problems have been arising as a result of the vast amount of data gathered today. A more specific problem is that estimation of the usual linear regression coefficients vector cannot be performed when the number of predictors exceeds the number of observations. Therefore, a sparsity assumption is often added. For example, the number of regression coefficients that are not equal to zero is assumed to be small. If it was known in advance which predictors have non zero coefficients, the classical linear regression estimator could have been used. Unfortunately, it is not known. Even worse, the natural relevant discrete optimization problem is usually not computationally feasible.

The lasso estimator (Tibshirani, 1996), which solves the problem of minimizing prediction error together with an ℓ_1–norm penalty, is possibly the most popular method to address this problem, since it results in a sparse estimator. Various algorithms are available to compute this estimator (e.g., Friedman et al., 2010). The theoretical properties of the lasso have been thoroughly researched in the past 15 years. For the high–dimensional problem, prediction rates were established in various manners (Greenshtein and Ritov, 2004; Bunea et al., 2006; Bickel et al., 2009; Bunea et al., 2007; Meinshausen and Yu, 2009). The capability of the lasso to choose the correct model depends on the true coefficient vector and the matrix of the predictors (Meinshausen and Bühlmann, 2006; Zhao and Yu, 2006; Zhang and Huang, 2008). However, the underlying assumptions are typically rather restrictive, and cannot be checked in practice.

In the high–dimensional setting, the task of finding the true model might be too ambitious, if meaningful at all. Only in certain situations, which could not be identified in practice, model selection consistency is guaranteed. Even in the classical setup, with more observations than predictors, there is no model selection consistent estimator unless further assumptions are fulfilled. This leads us to present a different objective. Instead of searching for a single "true" model, we aim to present a number of possible models a researcher should look at. Our goal, therefore, is to find potentially good prediction models. In short, we suggest to find the best models for each small model size. Then, by looking at these models one may reach interesting conclusions regarding the underlying problem. Some of these, as we demonstrate in applications, can be concluded using statistical reasoning, but most of these should be reasoned by a subject matter expert.

In order to find these models, we implement a search algorithm that uses simulated annealing (Kirkpatrick et al., 1983). The suggested algorithm is provided with a "score" for each predictor. We suggest to get these scores using a multi–step procedure that implements both the lasso and the elastic net (Zou and Hastie, 2005) (and then the lasso again). Multi–step procedures in the high–dimensional setting have drawn some attention and were demonstrated to be better than the standard lasso (Zou, 2006; Bickel et al., 2010).

The rest of the paper is organized as follows. Section 2 presents the concept of minimal class of models and the notations. Section 3 describes a search algorithm for relevant models, and gives motivation for the sequential use of the lasso and the elastic net when calculating scores for the predictors. Section 4 investigates the performance of the suggested search algorithm in simulation studies and then Section 5 illustrates data analysis using a minimal class of models in two examples. Section 6 suggests a short discussion. Technical proofs and supplementary data are provided in the appendix.

2. Description of the problem

We start with notations. First, denote $||v||_q := (\sum v_j^q)^{1/q}$, $q > 0$ for the ℓ_q (*pseudo*) norm of any vector v and $\|v\|_0 = \lim_{q\to 0} ||v||_q$ for its cardinality. The data consist of a matrix of predictors $X_{n\times p} = (X^{(1)}\ X^{(2)}\ ...\ X^{(p)})$ and a response vector $Y_{n\times 1}$. WLOG, X is centered and scaled and Y is centered as well. We are mainly interested in the case $p > n$. The

©2017 Daniel Nevo and Ya'acov Ritov.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-172.html.

underling model is $Y = X\beta^0 + \epsilon$ where $\epsilon_{n\times 1}$ is a random error, $E(\epsilon) = 0$, $V(\epsilon) = \sigma^2 I$, I is the identity matrix.

Denote $S \subseteq \{1, ..., p\}$ for a set of indices of X. We call S *a model*. We use $s = |S|$ to denote the cardinality of the set S. Denote also $S_0 := \{j : \beta^0 \neq 0\}$ and $s_0 = |S_0|$ for the true model, and its size, respectively. For any model S, we define X_S to be the submatrix of X which includes only the columns specified by S. Let $\hat{\beta}_S^{LS}$ be the usual least squares (LS) estimator corresponding to a model S, that is,

$$\hat{\beta}_S^{LS} = (X_S^T X_S)^{-1} X_S^T Y,$$

provided $X_S^T X_S$ is non singular.

Now, the straightforward approach to estimate S_0 given a model size κ is to consider the following optimization problem:

$$\min_{\beta} \frac{1}{n}||Y - X\beta||_2^2, \qquad \text{s.t} \quad ||\beta||_0 = \kappa. \tag{1}$$

Unfortunately, typically, solving (1) is computationally infeasible. Therefore, other methods were developed and are commonly used. These methods produce sparse estimators and can be implemented relatively fast. We first present here the lasso (Tibshirani, 1996), defined as

$$\hat{\beta}^L = \underset{\beta}{\operatorname{argmin}}\Big(\frac{1}{n}||Y - X\beta||_2^2 + \lambda||\beta||_1\Big)$$

where $\lambda > 0$ is a tuning constant. For some applications, a different amount of regularization is applied for each predictor. This is done using the weighted lasso, defined by

$$\hat{\beta}_w^L = \underset{\beta}{\operatorname{argmin}}\Big(\frac{1}{n}||Y - X\beta||_2^2 + \lambda||w \cdot \beta||_1\Big) \tag{2}$$

where w is a vector of p weights, $w_j \geq 0$ for all j, and $a \cdot b$ is the Hadamard (Schur, entrywise) product of two vectors a and b. Next is the elastic net estimator

$$\hat{\beta}^{EN} = \underset{\beta}{\operatorname{argmin}}\Big(\frac{1}{n}||Y - X\beta||_2^2 + \lambda_1||\beta||_1 + \lambda_2||\beta||_2^2\Big). \tag{3}$$

This estimator is often described as a compromise between the lasso and the well known Ridge regression (Hoerl and Kennard, 1970) since it could be rewritten as

$$\hat{\beta}^{EN} = \underset{\beta}{\operatorname{argmin}}\Big(\frac{1}{n}||Y - X\beta||_2^2 + \lambda\big(\alpha||\beta||_1 + (1-\alpha)||\beta||_2^2\big)\Big). \tag{4}$$

Let $\hat{\beta}_n$ be a sequence of estimators for β^0 and let $\hat{S}_n$ be the sequence of corresponding models. Model selection consistency is commonly defined as

$$\lim_{n\to\infty} P(\hat{S}_n = S_0) = 1. \tag{5}$$

If $p \ll n$ and small, then criteria based methods (e.g., BIC, Schwarz, 1978) are model selection consistent if p is fixed or if suitable conditions are fulfilled, see Wang et al. (2009)

and references therein. However, these methods are rarely computationally feasible for large p. For $p > n$, it turns out that practically strong and unverifiable conditions are needed to achieve (5) for popular regularization based estimators (Zhao and Yu, 2006; Meinshausen and Bühlmann, 2006; Huang et al., 2008; Jia and Yu, 2010; Tropp, 2004; Zhang, 2009).

In light of these established results, we suggest to pursue a different goal. Instead of finding a single model, we suggest to look for a group of models. Each of these models should include low number of predictors, but it should also be capable of predicting Y well enough. Therefore, $\mathcal{G}_0^n = \mathcal{G}_0^n(\kappa, \eta)$ is called a minimal class of models of size κ and efficiency η if

$$\mathcal{G}_0^n = \Big\{S : |S| = \kappa \;\&\; E||Y - X_S\beta_S^{0\star}||_2^2 \leq \min_{|S'|=\kappa} E||Y - X_{S'}\beta_{S'}^{0\star}||_2^2\} + \eta\Big\}.$$

where $\beta_S^{0\star}$ minimizes $E||Y - X_S\beta_S||_2^2$. Note that $\mathcal{G}_0^n$ depends on n, similarly to the way the true model has been considered previously (Greenshtein and Ritov, 2004; Meinshausen and Bühlmann, 2006; Zhao and Yu, 2006). In fact, if $p > n$ then, necessarily, p grows with n and, at the least, new potential predictors are added; the size of $\mathcal{G}_0^n$ may also change. Clearly, $\mathcal{G}_0^n$ is unknown. However, at a first sight, it is unclear how to refer to this set. Even when carrying out model selection, one hardly treats the true model as a parameter, even though it can be looked as such. In terms of inference, in simple, low dimensional, models, likelihood ratio tests can be used in some frequentist cases, or posterior probabilities for the Bayesians. More generally, model selection criteria, such as AIC (Akaike, 1974) or BIC, are typically used, without assigning formal statistical tools to address uncertainty. In this sense, the true minimal class of models is some type of a weak equivalence class, with respect to the true model S_0. Given η, it answers the following question. Are there additional models, other than S_0, that can be considered satisfactory? If the answer is yes, which are these models? Existence of alternative models may help researchers to question the strength of certain conclusions. Furthermore, when high–dimensional regression is used as part of a pilot study to determine which predictors should be measured regularly, alternative models may be compared in terms of cost. On the other hand, if there are multiple models explaining the data in a satisfactory manner, is there such a thing as the true model? If not, then conceptual concerns may arise. We further discuss these and related important issues in Section 6.

From a practical point of view, it would be useful to consider instead a sample version of $\mathcal{G}_0^n$, $\mathcal{G}^n(\kappa, \eta)$, defined as

$$\mathcal{G}^n(\kappa, \eta) = \Big\{S : |S| = \kappa \;\&\; \frac{1}{n}||Y - X_S\hat{\beta}_S^{LS}||_2^2 \leq \min_{|S'|=\kappa}\{\frac{1}{n}||Y - X_{S'}\hat{\beta}_{S'}^{LS}||_2^2\} + \eta\Big\}. \tag{6}$$

One could control how similar the models in $\mathcal{G} = \mathcal{G}^n$ are to each other in terms of prediction, using the tuning parameter η. A reasonable choice is $\eta = c\sigma^2$ with some $c > 0$. If σ^2 is unknown, it could be replaced with an estimate, e.g., using the scaled lasso (Sun and Zhang, 2012). An alternative to $\mathcal{G}$ is to generate the set of models by simply choosing for each κ the M models having the lowest sample mean square error (MSE), for some number M. The LS estimator, $\hat{\beta}_S^{LS}$, minimizes the sample prediction error for any model S with size $s \leq n$. Thus, this estimator is used for each of the considered models.

In practice, one may find $\mathcal{G}$ for a few values of κ, e.g., $\kappa = 1, ..., 10$, and then examines the pooled results, $\bigcup_{j=1}^{k} \mathcal{G}(j, \eta)$. Alternatively, the empirical MSE $n^{-1}||Y - X_S\hat{\beta}_S^{LS}||_2^2$ in the

definition of $\mathcal{G}$ can be replaced with one of the available model selection criteria, e.g., AIC, BIC or lasso. Then, models of varying size can be included in the class. Note that we are interested in situations where there is a fair number of models with a relatively very small number of variables (predictors) out of the available p.

At this point, a natural question is how can we benefit from using a minimal class of models. Examining the models in $\mathcal{G}$ may allow us to derive conclusions regarding the importance of different explanatory variables. We demonstrate this kind of analysis in Section 5 using two real data examples.

A minimal class of models could be also used in conjunction with the available models aggregation procedures. Aggregation of estimates obtained by different models was suggested both for the frequentist (Hjort and Claeskens, 2003), and for the Bayesian, (Hoeting et al., 1999). The well–known "Bagging" (Breiman, 1996) is also a technique to combine results from various models. Averaging across estimates obtained by multiple models is usually carried out to account for the uncertainty in the model selection process. We, however, are not interested in improving prediction per se, but in identifying good models. Nor are we interested in identifying the best model, since this is not possible or even meaningful in our setup, but in identifying predictors (and models) that are potentially relevant and important.

2.1 Relation to other work

A similar point of view on the relevance of a predictor was given by Bickel and Cai (2012). They considered a predictor to be important if its relative contribution to the predictive power of a set of predictors is high enough. Their next step was to consider only specific type of sets, such that their prediction error is low, yet they do not contain too many variables.

Rigollet and Tsybakov (2012) investigated the question of prediction under minimal conditions. They showed that linear aggregation of estimators is beneficial for high–dimensional regression when assuming sparsity of the number of estimators included in the aggregation. They also showed that choosing exponential weights for the aggregation corresponds to minimizing a specific, yet relevant, penalized problem. Their estimator, however, is computationally impossible and they have little interest in predictors and model identification.

As we describe in Section 3, our suggested search algorithm for candidate models travels through the model space. We choose to use simulated annealing to prevent the algorithm from getting stuck in a local minimum. Some Bayesian model selection procedures move along the model space, usually using a relevant posterior distribution, cf. O'Hara and Sillanpää (2009). We, however, do not assume any prior distribution for the coefficient values. Our use of the algorithm is only as a search mechanism, simply to find as many as possible models included in $\mathcal{G}$. Convergence properties of the classical simulated annealing algorithm are not of interest to our use of it. We are interested in the path generated by the algorithm and not in its final state.

3. A search algorithm

In this section, we suggest an algorithm to find $\mathcal{G}$ for a given κ and η. The problem is that $||Y - X_S\hat{\beta}_S^{LS}||_2^2$ is unknown for all S, and since p is large, even for a relatively small κ,

the number of possible models is huge (e.g., for $p = 200, k = 4$ there are almost 65 million possible models). We therefore suggest to focus our attention on smaller set of models, denoted by $\mathcal{M}(\kappa)$. $\mathcal{M}$ is a large set of models, but not too large so we can calculate MSEs for all the models within $\mathcal{M}$ in a reasonable computer running time. Once we have $\mathcal{M}$ and the corresponding MSEs, we can form $\mathcal{G}$ by choosing the relevant models out of $\mathcal{M}$.

The remaining question is how to assemble $\mathcal{M}$ for a given κ. Any greedy algorithm is bound to find models that are all very similar. Our purpose is to find models that are similar in their predictive power, but heterogeneous in their structure. For this we propose a simulated annealing algorithm (Kirkpatrick et al., 1983) which we now describe.

3.1 Simulated annealing algorithm

Our approach therefore is to implement a search algorithm which travels between potentially attractive models. We use a simulated annealing algorithm (Kirkpatrick et al., 1983), originally suggested for function optimization. The maximizer of a function $f(\theta)$ is of interest. Let $T = (t_1, t_2, ..., t_R)$ be a decreasing set of positive "temperatures". For every temperature level $t \in T$, iterative steps are carried out, before moving to the next, lower, temperature level. In each step, a random move from the current θ to another $\theta' \neq \theta$ is suggested. The move is then accepted with a probability that depends on $\exp\big[\big(f(\theta') - f(\theta)\big)/t\big]$. Typically, although not necessarily, a Metroplis–Hastings criterion (Metropolis et al., 1953; Hastings, 1970) is used to decide whether to accept the suggested move θ' or to stay at θ. After a predetermined number ,N_t, of such iterations, the algorithm moves to the next $t' < t$ in T, taking the final state in temperature t as the initial state for t'. The motivation for using this algorithm is that for high "temperatures", moves that do not improve the target function are possible, so the algorithm does not get stuck in a small area of the parameter space. However, as we lower the temperature, the decision to move to a suggested point is based almost solely on the criterion of improvement in the target function value. The name of the algorithm and its motivation come from annealing in metallurgy (or glass processing), where a strained piece of metal is heated, so that a reorganization of its atoms is possible, and then it colds off slowly so the atoms can settle down in low energy position. See Brooks and Morgan (1995) for a general review of simulated annealing in the context of statistical problems.

In our case, the parameter of interest is the model S and the objective function is

$$f(S) = -\frac{1}{n}||Y - X_S\hat{\beta}_S^{LS}||_2^2.$$

We now describe the proposed algorithm in more detail. We use simulated annealing with Metropolis–Hastings acceptance criterion as a search mechanism for good models. That is, we are not looking for the settling point of the algorithm; instead, we are following its path, hoping that much of it will be in neighborhood of good models.

We say the algorithm is in step (t, i) if the current temperature is $t \in T$ and the current iteration in this temperature is $i \in \{1, ..., N_t\}$. For simplicity, we describe here the algorithm for $N_t = N$ for all t. Let S_t^i and $\hat{\beta}_t^i$ be the model and the corresponding LS estimator in the beginning of the state (t, i), respectively. An iteration includes a suggested model S_t^{i+}, a LS estimator for this model, $\hat{\beta}_t^{i+}$, and a decision whether to move to S_t^{i+} and $\hat{\beta}_t^{i+}$ or to stay

at S_t^i and $\hat{\beta}_t^i$. We now define how S_t^{i+} is suggested and what is the probability of accepting this move.

For each S_t^i, we suggest S_t^{i+} by a minor change, i.e., we take one predictor out and we add another instead, and then obtain $\hat{\beta}_t^{i+}$ by standard linear regression. Assume that for every variable $j \in (1, ..., p)$, we have a score γ_j, such that higher value of γ_j reflects that the variable j should be included in a model, comparing with other possible variables. WLOG, assume $0 \leq \gamma_j \leq 1$ for all j. We choose a variable $r^* \in S_t^i$ and take it out with the probability function

$$p_{i,r}^{out} = \frac{{\gamma_r}^{-1}}{\sum\limits_{u \in S_t^i} {\gamma_u}^{-1}}, \quad \forall r \in S_t^i. \tag{7}$$

Next, we choose a variable $\ell^* \notin S_t^i$ and add it to the model with the probability function

$$p_{i,\ell}^{in} = \frac{\gamma_\ell}{\sum\limits_{u \notin S_t^i} \gamma_u}, \quad \forall \ell \notin S_t^i. \tag{8}$$

Thus,

$$S_t^{i+} = \{S_t^i \setminus r^*\} \cup \{\ell^*\}$$

and we may calculate the LS solution $\hat{\beta}_t^{i+}$ for the model S_t^{i+}. The first part of our iteration is over; a potential candidate was chosen. The second part is the decision whether to move to the new model or to stay at the current one. Following the scheme of simulated annealing algorithm with Metropolis–Hastings criterion, we calculate

$$q = \exp\Big(\frac{1}{nt}\big(||Y - X_{S_t^i}\hat{\beta}_t^i||_2^2 - ||Y - X_{S_t^{i+}}\hat{\beta}_t^{i+}||_2^2\big)\Big)\frac{Pr(S_t^{i+} \to S_t^i)}{Pr(S_t^i \to S_t^{i+})}$$

where

$$\begin{aligned} Pr(S_t^i \to S_t^{i+}) &= p_{i,r^*}^{out}\ p_{i,\ell^*}^{in} \\ Pr(S_t^{i+} \to S_t^i) &= p_{i^+,\ell^*}^{out}\ p_{i^+,r^*}^{in}. \end{aligned} \tag{9}$$

We are now ready for the next iteration $i+1$ by setting

$$(S_t^{i+1}, \hat{\beta}_t^{i+1}) = \begin{cases} (S_t^{i+}, \hat{\beta}_t^{i+}) & w.p \quad \min(1, q) \\ (S_t^i, \hat{\beta}_t^i) & w.p \quad \max(0, 1-q). \end{cases}$$

Along the run of the algorithm, the suggested models and their corresponding MSEs are kept. These models are used to form $\mathcal{M}(\kappa)$, and $\mathcal{G}$ can be then identified for a given value of η.

We now point out several issues for the practical application of the algorithm. First, the algorithm was described above for one single value of κ. In practice, one may run the algorithm separately for different values of κ. Another consideration is the tuning parameters of the algorithm that are provided by the user: The temperatures T; the number of iterations N; the starting point $S_{t_1}^1$; and the vector $\gamma = (\gamma_1, ..., \gamma_p)$. Our empirical experience is that the first three can be managed without too many concerns. In particular, the algorithm should be started from more than one initial point, and the results from

each run should be kept and compared; see Sections 4 and 5. Regarding the vector γ, a wise choice of this vector should improve the chance of the algorithm to move in desired directions. We deal with this question in Section 3.2. However, in what follows we show that, under suitable conditions, the algorithm can work well even with a general choice of γ.

Define S_0, s_0 and β^0 as before and let $\mu = X\beta^0$. That is, $Y = \mu + \epsilon$. We first introduce a few simple and common assumptions:

(A1) $||\mu||_2^2 = O(n)$

(A2) s_0 is small, i.e., $s_0 = O(1)$.

(A3) $p = n^a, a > 1$

(A4) $\epsilon \sim N_n(0, \sigma^2 I)$

Denote A_γ for the set of positive entries in γ. That is, $A_\gamma \subseteq \{1, .2, ...p\}$ is a (potentially) smaller group of predictors than all the p variables. Denote also $h_\gamma = |A_\gamma|$ for the cardinality of A_γ and $\gamma_{min} := \min\limits_{i \in A_\gamma} \gamma_i$ for the lowest positive entry in γ.

To motivate our next assumption, we note that, informally, the algorithm is expected to preform reasonably well if:

1. The true model is relatively small (e.g., with 10 active variables).
2. A variable in the true model is adding to the prediction of a set of variables if a very few (e.g., 2) other variables are in the set.

Our next assumption is more restrictive. Let $\bar{S}$ be an interesting model of size s_0–a model with not too many predictors and with a low MSE. The models we are looking for are of this nature. We facilitate the idea of $\bar{S}$ being an interesting model by assuming that $X_{\bar{S}}\hat{\beta}_{\bar{S}}$ is close to μ (in the asymptotic sense). We virtually assume that for every model of size $|\bar{S}|$, which is not $\bar{S}$, if we take out a predictor that is not part of $\bar{S}$, and replace it with a predictor from $\bar{S}$, the subspace spanned by the new model is not much further from μ, comparing with the subspace spanned by the original model. Formally, denote $\mathcal{P}_{\mathcal{S}}$ for the projection matrix onto the subspace spanned by the columns of the submatrix X_S.

(B1) There exist $t_0 > 0$ and a constant $c > 0$, such that for all S, $|S| = s_0 - 1$, for all $j \in \bar{S} \cap S^c$, $j' \in \bar{S}^c \cap S^c$, and for a large enough n

$$\frac{1}{n}\left[||\mathcal{P}_{S_j^\star}\mu||_2^2 - ||\mathcal{P}_{S_{j'}^\star}\mu||_2^2\right] > 4t_0 \log c, \tag{10}$$

where $S_r^\star \equiv S \cup \{r\}$.

We note that since c could be lower than one, the right hand side of (10) can be negative. The following theorem gives conditions under which the simulated annealing algorithm is passing through an interesting model $\bar{S}$. More accurately, the theorem states that there is always strictly positive probability to pass through $\bar{S}$ in the next few moves. This result covers all models that Assumption (B1) holds for. Note however, that we do not claim that

the algorithm finds all the models in a minimal class. Proving such a result would probably require complicated assumptions on models with larger size than s_0, and their relation to $\bar{S}$ and other interesting models.

Let $P_t^m(S'|S)$ be the probability of passing through model S' in the next m iterations of the algorithm, given the current temperature is t, and the current state of the algorithm is the model S.

Theorem 1 *Consider the simulated annealing algorithm with $\kappa = s_0$ and with a vector γ such that $\gamma_{min} \geq c_\gamma$. Let Assumptions (A1)–(A4) hold and let Assumption (B1) hold for some temperature t_0 and with $c = c_\gamma$. If $\bar{S} \subseteq A_\gamma$ then for all $S \subseteq A_\gamma$ with $s = s_0$, for all $m \geq s_0 - |\bar{S} \cap S|$ and for large enough n,*

$$P_{t_0}^m(\bar{S}|S) > \left[\frac{c_\gamma^2}{s_0(h_\gamma - s_0)}\right]^{s_0}. \tag{11}$$

A proof is given in the appendix. Theorem 1 states that for any specification of the vector γ, such that the entries in γ are positive for all the predictors in $\bar{S}$, the probability that the algorithm would visit $\bar{S}$ in the next m moves is always positive, provided the temperature is high enough, and provided it is possible to move from the current model to $\bar{S}$ in m moves.

For the classical model selection setting with $p < n$, a similar method was suggested by Brooks et al. (2003). Their motivation is as follows. When searching for the most appropriate model, likelihood based criteria are often used. However, maximizing the likelihood to get parameters estimates for each model becomes infeasible as the number of possible models increases. They therefore suggest to simplify the process by maximizing simultaneously over the parameter space and the model space. They suggest a simulated annealing type algorithm to implement this optimization. This algorithm is essentially an automatic model selection procedure.

3.2 Choosing γ

The simulated annealing algorithm described above is provided with the vector γ. The values $\gamma_1, ..., \gamma_p$ should represent the knowledge regarding the importance of the predictors, although we do not assume that any prior knowledge is available. As it can be seen in equations (7)–(8), predictors with high γ values have larger probability to enter the model if they are not part of the current model, and lower probability to be suggested for replacement if they are already part of it. Since p is large, we may also benefit if γ includes many zeros.

One simple choice of γ is to take the absolute values of the univariate correlations of the different predictors with Y. We could also threshold the correlations in order to keep only predictors having large enough correlation (in absolute value) with Y. However, using univariate correlations is clearly problematic since it overlooks the covariance structure of the predictors in X.

Another possibility is to first use the lasso with a relatively low penalty, and then to set $\gamma_j = |\hat{\beta}_j^L|/||\hat{\beta}^L||_1$. The idea behind this suggestion is that predictors with large coefficient value may be more important for prediction of Y.

However, as discussed in Section 2, the lasso might miss some potentially good predictors. It is well known that the elastic net may add these predictors to the solution, although

it might also add unnecessary predictors. Moreover, it is not clear how to choose γ_j using solely the elastic net. The lasso and the elastic net estimators are not model selection consistent in many situations. However, for our purpose, combining both methods together may help us get a reservoir of promising predictors.

Zou and Hastie (2005) provided motivation and results that justify the common knowledge that the elastic net is better to use with correlated predictors. Since we intend to exploit this property of the elastic net, this paper offers an additional theoretical background. We present a more general result later on this section, but for now, the following proposition demonstrates why the elastic net tends to include correlated predictors in its model.

Proposition 2 *Define X and Y as before, and define $\hat{\beta}^{EN}$ by (3). Let $X^{(1)}$ and $X^{(2)}$ be two columns of X and denote $\rho = n^{-1}(X^{(1)})^T X^{(2)}$. Assume $|\hat{\beta}_1^{EN}| \geq c_\beta$ for some $c_\beta > 0$. If $|\rho| > 1 - \lambda_2^2 c_\beta^2/||Y||_2^2$ then $|\hat{\beta}_2^{EN}| > 0$.*

A proof is given in the appendix. Proposition 2 gives motivation for why $\hat{\beta}^{EN}$ has typically a larger model than $\hat{\beta}^L$. It also quantifies how much correlated two predictors need to be so the elastic net would either include both predictors or none of them.

Going back to our γ vector, the next question is how to use the lasso and the elastic net in order to assign a "score" to each predictor. Let S_L and S_{EN} be the models that correspond to $\hat{\beta}^L$ and $\hat{\beta}^{EN}$, respectively. Define S_+ for the group of predictors that were part of the elastic net model but not part of the lasso model and S_{out} for the predictors that were not included in any of them. Note that $S_L \cap S_+ = S_L \cap S_{out} = S_+ \cap S_{out} = \emptyset$ and $S_L \cup S_+ \cup S_{out}$ is $\{1, ..., p\}$. Define

$$\hat{\beta}_+^L(\delta) = \arg\min_\beta \Big(\frac{1}{n}||Y - X\beta||_2^2 + \lambda \sum_{j=1}^p \delta^{\mathbf{1}\{j \in S_+\}}|\beta_j|\Big), \qquad \delta \in [0, 1],$$

and let $S_+^L(\delta)$ be the appropriate model. In this procedure, a reduced penalty is given for predictors that $\hat{\beta}^L$ might have missed. Thus, these predictors are encouraged to enter the model, and since they may take the place of others, predictors in S_L that their explanation power is not high enough are pushed out of the model. Note that $\hat{\beta}_+^L(\delta)$ is a special case of $\hat{\beta}_w^L$, as defined in (2), with $w_j = \delta^{\mathbf{1}\{j \in S_+\}}$.

We demonstrate how the reduced–penalty procedure works using a toy example. A data set with $n = 30$ and $p = 50$ is simulated. We take the coefficient vector to be $\beta^0 = (0.5\ 0.5\ 1\ 1\ 1\ 0\ 0\ ...\ 0)^T$ and σ^2 is taken to be one. The predictors are independent normal variables with the exception of 0.8 correlation between $X^{(1)}$ and $X^{(2)}$. Predictor 1 is included in the lasso model, however predictor 2 is not. Figure 1 presents the coefficients' estimates of $X^{(1)}, X^{(2)}$ and $X^{(3)}$ when lowering the penalty of $X^{(2)}$. Note how $X^{(2)}$ enters the model for low enough penalty while $X^{(1)}$ leaves the model for low enough penalty (on $X^{(2)}$).

We suggest to measure the importance of a predictor $j \in S_+$ by the highest δ such that $j \in S_+^L(\delta)$. On the other hand, the importance of a predictor $j' \in S_L$ can be measured by the highest δ such that $j' \notin S_+^L(\delta)$ (now, smaller δ reflects j' is more important). With this in our mind, we continue to the derivation of γ.

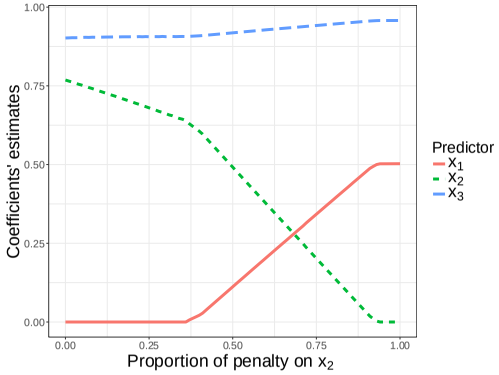


Figure 1: Toy example: coefficients' estimates for predictors $X^{(1)}, X^{(2)}$ and $X^{(3)}$ when lowering the lasso penalty for $X^{(2)}$ only. The rightmost point corresponds to a lasso procedure with equal penalties for all predictors

Let $\Delta = (\delta_0 < \delta_1 < ... < \delta_h)$ be a grid of $[0, 1]$, with $\delta_0 = 0$ and $\delta_h = 1$. For each $\delta \in \Delta$, we obtain $\hat{\beta}^L_+(\delta)$. Define

$$i^\star_j = \begin{cases} \underset{i}{\text{argmax}}\{i : \hat{\beta}^L_{+j}(\delta_i) \neq 0\} & j \notin S_L \\ \underset{i}{\text{argmax}}\{i : \hat{\beta}^L_{+j}(\delta_i) = 0\} & j \in S_L \end{cases}$$

and if the argmax is over an empty set, define $i^\star_j = 0$. We suggest to choose γ_j as follows:

$$\gamma_j = \begin{cases} 0 & j \in S_{out} \\ \delta_{i^\star_j}/2 & j \in S_+ \\ \\ 1 - \delta_{i^\star_j}/2 & j \in S_L, \end{cases}$$

for all $j \in \{1, ..., p\}$. This choice of γ has the following nice properties.

- A predictor $j \notin S_L$ with $i^\star_j = 0$ is excluded from consideration.
- On the other hand, for a predictor $j \in S_L$, if $i^\star_j = 0$ than $\gamma_j = 1$, which is the maximal possible value. Even when the penalty for other predictors is dramatically reduced, leading to their entrance to the model, j remains part of the solution and hence it is essential for prediction of Y.
- Since predictors in S_L were picked when equal penalties were assigned to all predictors, they get priority over predictors in S_+.

- However, for two identical predictors, $X^{(j)} = X^{(j')}$ (or highly correlated predictors) such that $j \in S_L$ and $j' \notin S_L$, we get a desirable result. By Proposition 2, we know that $X'_j \in S_+$. Now, for $\delta_{h-1} < 1$ it is clear that $j' \in S^L_+(\delta_{h-1})$ and $j \notin S^L_+(\delta_{h-1})$. Therefore. $i^\star_j = i^\star_{j'} = h-1$ hence if δ_{h-1} is taken to be close to one, then $\gamma_j \simeq \gamma_{j'} \simeq 0.5$ as one might want.

Proposition 2 concerns two correlated predictors. In practice, the covariance structure of X may be much more complicated. Therefore the question arises: can we say something more general on the elastic net in the presence of competing models? Apparently we can. Let M_1 and M_2 be two models, that is, two sets of predictors, that possibly intersect. Assume that the elastic net solution chose all the predictors in M_1. What can we say about the predictors in M_2? Are there conditions on X_{M_2}, X_{M_1} and Y such that all the predictors in M_2 are also chosen? If the answer is yes (and it is, as Theorem 3 states), it justifies our use of the elastic net to reveal more relevant predictors. In our case, the relevant predictors are the building blocks of models in $\mathcal{G}$.

In order to reveal this property of the elastic net, we analyze $\hat{\beta}^{EN}$, the solution of (3), when assuming all the predictors in M_1 have non-zero values. Denote $M^{(-)}$ for $(M_1 \cup M_2)^c$, the set of predictors that are not included in M_1 or M_2 and $\tilde{X} = X_{M^{(-)}}$ for the appropriate submatrix of X. Let $\hat{\beta}^{EN}_{M_1}$, $\hat{\beta}^{EN}_{M_2}$ and $\hat{\beta}^{EN}_{M^-}$ be the coordinates of $\hat{\beta}^{EN}$ that correspond to M_1, M_2 and $(M_1 \cup M_2)^c$, respectively. Let $\tilde{Y} = Y - \tilde{X}\hat{\beta}^{EN}_{M^-}$ be the unexplained residual of Y, after taking into account $\tilde{X}$. Finally, we show that both M_1 and M_2 are chosen by the elastic net if the prediction of $\tilde{Y}$ using M_1, namely $X_{M_1}\hat{\beta}^{EN}_{M_1}$, projected onto the subspace spanned by the columns of M_2 is correlated enough with $\tilde{Y}$. Formally,

Theorem 3 *Define $\hat{\beta}^{EN}$ as before. Let M_1 and M_2 be two models with the appropriate submatrices X_{M_1} and X_{M_2}. Define $\tilde{X}$ and $\tilde{Y}$ as before. Define $\hat{\beta}^{EN}_{M_1}$ and $\hat{\beta}^{EN}_{M_2}$ as before. Denote $\mathcal{P}_{M_2}$ for the projection matrix onto the subspace spanned by the columns of X_{M_2}. WLOG, assume $|M_2| \leq |M_1|$ and that all the coordinates of $\hat{\beta}^{EN}_{M_1}$ are different than zero. Finally, if*

$$\tilde{Y}^T \mathcal{P}_{M_2} X_{M_1} \hat{\beta}^{EN}_{M_1} > c_1(\lambda_1, \lambda_2, X_{M_1}, \tilde{Y}, \hat{\beta}^{EN}_{M_1}), \tag{12}$$

then all the coordinates of $\hat{\beta}^{EN}_{M_2}$ are different than zero.

A proof and a discussion on the technical aspects of condition (12) and the constant c_1 are given in the appendix. Theorem 3 states that under a suitable condition, predictors belonging to at least one of two competing models are chosen by the elastic net. In our context, when we have a model M_1 with a good prediction accuracy, i.e., $X_{M_1}\hat{\beta}^{EN}_{M_1}$ is close to $\tilde{Y}$, then predictors in any another model M_2 which has similar prediction, that is $\mathcal{P}_{M_2} X_{M_1}\hat{\beta}^{EN}_{M_1}$ is also close to $\tilde{Y}$, would be chosen by the elastic net. Hence, these predictors are expected to have a positive value in γ, and our simulated annealing algorithm would pass through these models, provided the conditions in Theorem 1 are met. Therefore, these models are expected to appear in $\mathcal{G}$.

4. Simulation Studies

In order to examine the performance of the suggested search algorithm, we present in this section results from two simulation studies. The first investigates whether models similar in

their capability to describe the underlying population can be found by the algorithm. The second simulation study examines the out–of–sample prediction error of models included in the minimal class of models.

We present below a scenario in which, other then the model used to create the data, there are three more models, that are not nested within the underlying model, with population prediction error close to the underlying model. In practice, all of these four models are of interest.

We consider $Y = X\beta^0 + \epsilon$, $\epsilon \sim N(0, I)$ with β_j^0 equals to C for $j = 1, 2, ..., 6$ and zero for $j > 6$. C is a constant chosen to get a desired signal to noise ratio (SNR) $||X\beta^0||_2$. The predictors in X are all i.i.d. $N(0, I)$ with the exception of $X^{(7)}$ and $X^{(8)}$, which are defined by

$$X^{(7)} = \frac{2}{3}[X^{(1)} + X^{(2)}] + \xi_1, \qquad \xi_1 \sim N_n\left(0, \frac{1}{9}I\right)$$
$$X^{(8)} = \frac{2}{3}[X^{(3)} + X^{(4)}] + \xi_2, \qquad \xi_2 \sim N_n\left(0, \frac{1}{9}I\right)$$

where ξ_1 and ξ_2 are independent. In this scenario, there are 4 models we would like to find: (I) {1,2,3,4,5,6}; (II) {5,6,7,8}; (III) {3,4,5,6,7}; and (IV) {1,2,5,6,8}. In particular, each of these models minimizes the population prediction error $E||Y - X_S\beta_S||_2^2\}$ for models with its size. That is, Model (I) minimizes the population prediction error for models of size $k = 6$, Model (II) minimizes the prediction error for models of size $k = 4$, and both models (III) and (IV) minimize the prediction error for models of size $k = 5$. Furthermore, it can be shown, for example, that the population prediction error of Model (III) is just 3.7% larger than the error of true model for $SNR = 1$, 14.8% for $SNR = 2$, and larger values for $SNR \geq 4$

Note that while models (II)–(IV) do not describe the data as well as Model (I), they contain less predictors. Furthermore, in an hypothetical real-life situation where future measurements were to be made on the predictors and the outcome, and, for the sake of the example, $X^{(1)}$ or $X^{(2)}$ were more expensive to measure, models (II)–(IV) could have provided a frugal alternative, while preserving a reasonable prediction error.

As part of this simulation study, for each simulated data set, we do the following:

1. Obtain γ as explained in Section 3.2. The tuning parameter of the lasso is taken to be the minimizer of the cross–validation MSE. For the elastic net, α in (4) is taken to be 0.4.

2. Run the simulated annealing algorithm for $\kappa = 4, 5, 6$. The tuning parameters of the algorithm are chosen quite arbitrarily: $T = (10 \times 0.7^1, 10 \times 0.7^2, ..., 10 \times 0.7^{20})$; $\Delta = (0, 0.02, 0.04, ..., 0.98, 1)$; $N_t = N = 100$ for all $t \in T$.

3. Then, for each model (I)–(IV), we check whether the model is the best model obtained (as measured by MSE) among models with the same size. For example, we check if Model (II) is the best model out of all models that were found with $\kappa = 4$. We also check whether the model is one of the top five models among models with the same size.

SNR	Model	$p = 200$ Best	$p = 200$ Top 5	$p = 500$ Best	$p = 500$ Top 5	$p = 1000$ Best	$p = 1000$ Top 5
1	(I)	0.00	0.01	0.00	0.00	0.00	0.00
	(II)	0.42	0.62	0.28	0.46	0.23	0.38
	(III)	0.04	0.08	0.01	0.02	0.00	0.00
	(IV)	0.04	0.08	0.02	0.03	0.00	0.01
2	(I)	0.10	0.12	0.05	0.06	0.04	0.05
	(II)	0.94	0.96	0.92	0.94	0.94	0.95
	(III)	0.27	0.34	0.18	0.24	0.15	0.17
	(IV)	0.28	0.37	0.18	0.22	0.14	0.17
4	(I)	0.38	0.38	0.20	0.20	0.11	0.11
	(II)	0.96	0.96	0.96	0.96	0.96	0.95
	(III)	0.38	0.46	0.31	0.36	0.22	0.24
	(IV)	0.39	0.46	0.28	0.31	0.24	0.26
8	(I)	0.72	0.72	0.46	0.46	0.32	0.32
	(II)	0.97	0.97	0.97	0.97	0.96	0.96
	(III)	0.41	0.48	0.36	0.40	0.30	0.31
	(IV)	0.44	0.50	0.34	0.37	0.29	0.31
12	(I)	0.86	0.86	0.66	0.66	0.49	0.49
	(II)	0.98	0.98	0.97	0.97	0.96	0.96
	(III)	0.49	0.55	0.41	0.44	0.32	0.34
	(IV)	0.42	0.48	0.37	0.40	0.32	0.34

Table 1: Proportion that each model is chosen as best model or as one of top five models for different number of potential predictors (p) and various SNR values.

A 1000 simulated data sets were generated for each different scenario: For $n = 100$, $p = 200, 500, 1000$ and for SNR $= 1, 2, 4, 8, 12, 16$. Table 1 displays the proportion of times each model was chosen, either as the best one, or as one of the top five models. The results are as one might expect. For large SNR, the models are chosen more frequently. However, models (III) and (IV) are competing, in the sense that they both include five predictors. Even for large SNR, each of the models, (III) and (IV), is chosen in about 50% of the cases. Therefore, as recommended in Section 3.1, we repeat the simulations while starting the algorithm from different initial points, that is, different initial models.

Figure 2 presents comparison between running the algorithm from one and three starting points. When we initiated the algorithm from three different models, the results improved for all models, and in particular for models (III) and (IV). The results described in this section are quite similar to the results we obtained when forming $\mathcal{G}(\kappa, \eta)$ as defined in (6), for each $\kappa = 4, 5, 6$ separately and using arbitrary small values of η.

We now turn to describe the second simulation study, that concerns out–of–sample prediction of the models found by the algorithm. Here we consider three scenarios. The true model consisting 20 predictors for all scenarios. In Scenario 1, X consists p iid normally distributed predictors with mean zero and variance one. The 20 first entries in β^0 are created from iid normal distribution, and then rescaled to achieve the desired SNR. In Scenario 2,

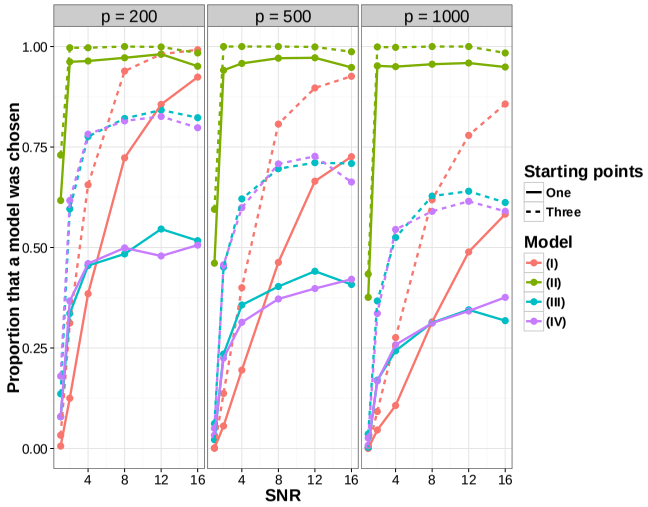


Figure 2: Proportion that each model is chosen as one of top five models for different number of potential predictors (p) and various SNR values. There is an apparent improvement when running the algorithm from three starting points.

β^0 is created in the same as in Scenario 1, but a more complex correlation structure is considered for X. First, define $\Sigma^{(m)}$ to be an $m \times m$ covariance matrix defined so for all $i = 1, ..., m$ and $j = 1, ., m$, $\Sigma_{ij}^{(m)} = 0.75^{|i-j|}$. The subvector containing the first 5 predictors, $(X^{(1)}, X^{(2)}, ..., X^{(5)})$ is created with the covariance matrix $\Sigma^{(5)}$. Then, each of the next 10 predictors $X^{(j)}, j = 6, 7..., 15$ is correlated with two predictors that are not part of the true model. For example, $(X^{(6)}, X^{(21)}, X^{(22)})$ is simulated from $N_3(0, \Sigma^{(3)})$ and $(X^{(7)}, X^{(23)}, X^{(24)})$ is simulated from $N_3(0, \Sigma^{(3)})$. The remaining predictors are simulated as iid $N(0, 1)$. In Scenario 3, X is created as in Scenario 2, but the coefficients in the true model are now all equal. That is, $\beta_j^0 = C, j = 1, 2, ..., 20$, with C being a constant chosen to get a desired SNR.

As in the previous study, $Var(\epsilon) = I, n = 100$ and we consider $p = 200, 500, 1000$. We report the results for SNR=1,2,4,6,8. For each synthetic data set, the simulated annealing is used for $\kappa = 15, 20, 25$. The algorithm is initiated three times per κ value, and the best five models per run are kept, and then combined to have a minimal class of models containing up to 15 models (per κ value). Other tuning parameters are chosen as in the first simulation study. Figure 3 compares mean (over the synthetic data sets) out–of–sample prediction error for the best model (in terms of prediction error) within the minimal class of models, the average of out–of–sample prediction error among the models in the minimal class, as well as the errors of the lasso, elastic net and adaptive lasso. Results are presented

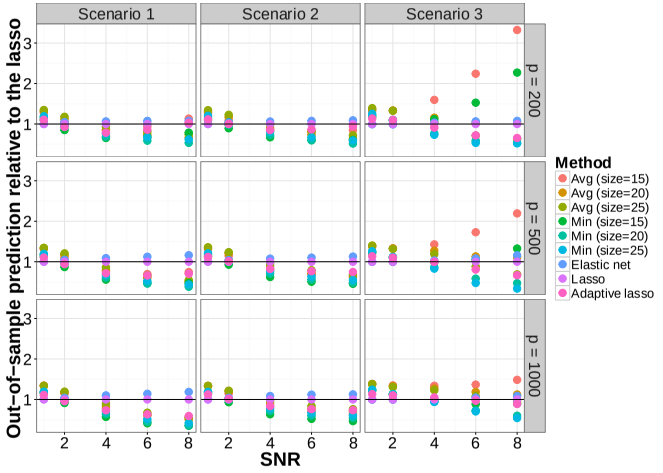
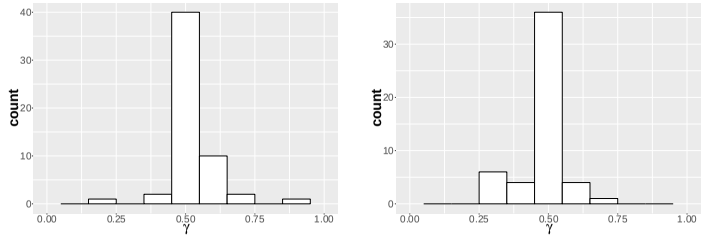


Figure 3: Out–of–sample prediction error for different scenarios, different number of potential predictors (p) and various SNR values. Methods compared are lasso, elastic net and adaptive lasso, as well as the average (avg) and minimal (min) error in the minimal class of models, with model size of 15,20 and 25 predictors. The classes were built by combining 5 best models obtained from three runs of the simulated annealing search algorithm.

relative to the out–of–sample prediction error of the lasso. For SNR=1, the lasso performs the best, for all three scenarios considered, although the other methods show comparable performance. Under Scenarios 1 and 2, when the non–zero parameters are created from a normal distribution, so some predictors are stronger and others are weaker, the best model identified by the minimal class, for either $\kappa = 20$ or $\kappa = 25$, outperforms other methods for SNR>1. Under Scenario 3, where all coefficients (in the true model) are equal, the same pattern is observed for SNR>4. For large SNR values, under Scenario 3, the models in the minimal class of size 15 are underperforming. While under Scenarios 1 and 2, the predictors missed by the minimal class with $\kappa = 15$ are likely to be those with small effect, under Scenario 3 all predictors have an equal effect. We also considered scenarios with X simulated from $N_p(0, \Sigma^{(p)})$. The results are similar to those observed for Scenario 1 and hence are omitted.

5. Real data sets

We demonstrate the utility of using a minimal class of models in the analysis of two real data sets. The tuning parameters of the lasso and the elastic net were taken to be the same as in Section 4. The tuning parameters of the simulated annealing algorithm were



(a) Riboflavin data (b) Air pollution data

Figure 4: Histograms of the values of γ for positive entries only in the two data set analysis examples.

$T = 10 \times (0.7^1, 0.7^2, ..., 0.7^{20})$, $\Delta = (0, 0.01, 0.02, ..., 0.98, 0.99, 1)$, and $N_t = N = 100$ for all $t \in T$.

5.1 Riboflavin

We use a high–dimensional data about the production of riboflavin (vitamin B2) in Bacillus subtilis that were recently published (Bühlmann et al., 2014). The data consist $p = 4088$ predictors. These are measures of log expression levels of genes in $n = 71$ observations. The target variable is the (log) riboflavin production rate.

The lasso model S_L includes 40 predictors (and intercept), and the elastic net model S_{EN} includes 59 predictors. In total, we get 61 different predictors (i.e., genes). Panel (a) of Figure 4 presents the histogram of the positive values in γ.

We run the algorithm from three random starting points for each model size between 1 and 10. We keep the five best models for each size and starting point, to get, after removal of duplicates, a total of 112 models. See Table 2 for the number of unique models as a function of the model size. Following a referee comment, we note here that in practice models with low model size may have unacceptably low R^2, and this can be checked in practice. In our analysis here we keep these models since the R^2 values are all larger than 0.35 and to ease the exposition, so we would be able to compare the search results to models obtained by other methods. The models found by other methods, for the riboflavin data, are of a relatively small size; see Bühlmann et al. (2014)

The following insights are drawn from examining more carefully the models we obtained (see Table 1 in the online appendix):

- In total, the models include 53 different predictors. Out of these, 35 predictors appear in less than 10% of the models, meaning they are probably less important as predictors of riboflavin production rate.

- Gene number 2564 appears in all models of size larger than 3 and in 5 out of 8 models of size 3. However, this gene is not included in any of the smaller models. This gene is the only one that appears in more than half of our models. We can infer that while

Model size	1	2	3	4	5	6	7	8	9	10
Number of models	5	5	8	6	13	15	15	15	15	15

Table 2: Riboflavin data: Number of unique models for each model size after running the algorithm from 3 different starting points

this gene does not hold an effect strong enough comparing to other genes in order to stand out, it has a unique relation with the outcome predictor that could not be mimicked using other combination of genes.

- At least one gene from the group $\{4002, 4003, 4004, 4006\}$ is contained in all models of size larger than one, although never more than one of these genes. Genes number 4003 and 4004 appear more frequently than genes number 4002 and 4006. Looking at the correlation matrix of these genes only, we see they are all highly correlated (pairwise correlations > 0.97). Future research could take this finding into account by using, e.g., the group lasso (Yuan and Lin, 2006).

- Similarly, either gene number 1278 or gene number 1279 appear in about half of the models. They are also strongly correlated (0.984). The same statement holds for genes number 69 and 73 (correlation of 0.945) as well.

- The importance of genes number 792,1131, and possibly others, should be also examined since each of them appears in a variety of different models.

We now compare our results to models obtained using other methods, as reported in Bühlmann et al. (2014). The multiple sample splitting method to get p–values (Meinshausen et al., 2009) yields only one significant predictor. Indeed, a model that includes only this predictor is part of our models. If one constructs his model using the stability selection (Meinshausen and Bühlmann, 2010) as a screening process for the predictors, he would get a model consisting three genes, which correspond to columns number $625, 2565$ and 4004 in our X matrix. However, this model is not included in our top models. In fact, the highest MSE for a model in our 8 models of size 3 is 0.2047 while the MSE of the model suggested using the stability selection is 0.2703, more than 30% difference!

5.2 Air pollution

We now demonstrate how the proposed procedure can be used for traditional, purportedly simpler, problem. The air pollution data set (McDonald and Schwing, 1973) includes 58 Standard Metropolitan Statistical Areas (SMSAs) of the US (after removal of outliers). The outcome variable is age-adjusted mortality rate. There are 15 potential predictors including air pollution, environmental, demographic and socioeconomic predictors. Description of the predictors is given in Table 4 in the appendix.

There is no guarantee that the relationship between the predictors and the outcome variable has a linear form. We therefore include commonly used transformations of each

variable, namely natural logarithm, square root and power of two transformations. Considering also all possible two way interactions, we have a total of 165 predictors.

High–dimensional regression model that includes transformations and interactions has been dealt with in the literature. For example, by using two step procedures (Bickel et al., 2010) or by solving a relevant optimization problem (Bien et al., 2013). Our procedure has a different goal, since we are not looking for the best predictive model, but rather for meaningful insights about the data.

Following the lasso and elastic net step, we are left with 51 predictors with positive γ_j (6 untransformed predictors, 4 log transformations, 6 square root transformations, 9 power of two transformations and the rest are interactions). Panel (b) of Figure 4 presents the histogram of the positive values in γ.

We modify the search algorithm to make it produce results typical to an analysis involving interaction terms. In particular, we search for models such that an interaction term is included only if at least one of the corresponding main terms is included in the model. In order to do so, the definitions in (7) and (8) are modified such that the probability of proposing an interaction term is zero if none of the corresponding main effects is in this model (excluding the variable chosen to be taken out). On the other hand, main effects are "protected" of being suggested to be excluded from the model, if any interaction involving them is part of the current model. We achieve this by setting to zero the probability of suggesting a predictor when an interaction term involving this predictor is included.

Following our earlier comment on considering satisfactory models, and since we are looking for more complex models in this example, considering the option of transformations and interactions, we run the algorithm for $\kappa = 5, 6, ..., 15$, for each κ, from three starting points, and then we keep the 5 best models. In total, we get 164 unique models. Table 3 summarizes the results for prominent main effect predictors, that is, predictors that appear in at least third of the models we obtained. The table presents a matrix of the joint frequency of each two predictors. Each cell in the table is the number of models including both the predictor listed in the row and the predictor listed in the column. The diagonal is simply the number of models that a predictor appears in. Considering Table 3, the nitric oxide pollution is invaluable for prediction of mortality rate. This predictor (in a log shape) appears in a large majority of the models. The percentage of non–white population also appears in most models. In almost half of them, it appears untransformed, but the same could be said about this predictor after square root transformation. However, in less than 10% of the models, this predictor appears in both forms. We conclude that this predictor should be used for prediction of the mortality rate, but the question of transformation remains unsolved. Similar comments can be made about hydrocarbon pollution.

We turn to the interactions. Because of the way we searched for interactions, it becomes "harder" for an interaction term to enter a model, and we therefore analyze them separately. Out of the 26 interactions considered, none clearly stood out above the rest. Five of the interaction terms appear in about 15% of the models. Two of those involve the percentage of non–white population, one with the prevalence of low–income, and the other with percentage of housing units with all facilities. Another of those five interaction terms is the interaction between percentage of elderly population and the average temperature in January. The latter main effect appears in about 50 models, and hence did not make the cutoff for Table 3. The percentage of elderly population was not included in either the lasso or elastic net

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
(1) **prec**	57	25	25	52	27	29	18	18	27
(2) **nwht**		81	33	75	34	15	39	37	39
(3) log(**HC**)			61	60	21	34	9	29	33
(4) log(**NOx**)				158	67	85	64	56	80
(5) $\sqrt{\textbf{jant}}$					71	37	25	23	37
(6) $\sqrt{\textbf{nwht}}$						85	33	27	45
(7) $\sqrt{\textbf{HC}}$							66	29	22
(8) $\textbf{jult}^2$								61	24
(9) $\textbf{educ}^2$									83

Table 3: Frequency that each two predictors together in the 164 models. The diagonal is simply the number of models that a predictor appears in. For example, in 67 models both log(**NOx**) and $\sqrt{\textbf{jant}}$ appear.

preliminary steps. However, the absence of age related effect is not so surprising since the outcome variable, the mortality rate, is age corrected. Nevertheless, since the interaction term appeared in about half of the models that included January temperature as a predictor, it can be considered if the main effect is also considered.

6. Discussion

Model selection consistency is an ambitious goal to achieve when dealing with high–dimensional data. A "minimal class of models" was defined to be a set of models that should be considered as candidates for prediction of the outcome variable. A search algorithm to identify these models was developed using a simulated annealing method. Under suitable conditions, that are outlined in Theorem 1, the algorithm passes through models of interest.

A score for each predictor is given using the lasso, the elastic net and a reduced–penalty lasso. These scores are used by the search algorithm. They are not necessarily optimal but we claim that they are sensible. Other scoring methods may achieve better results. On the other hand, the scores we use here may be used for other purposes. Theoretical justification for using the elastic net to unveil predictors the lasso might have missed was also presented. A simulation study demonstrated the capability of the search algorithm to detect relevant models.

One possible limitation of the proposed algorithm is the number of tuning parameters to be chosen. In our simulation studies and data analyses, we have found the algorithm to be quite robust to the parameters' specification. For ease of applications, we now list suggested default values for some of the parameters. The tuning parameter of the lasso λ, can be chosen by cross validation. Since the subsequent use of the elastic net is to bring into surface potential predictors, relatively low value for α in (4) can be taken, say $\alpha \in (0.2, 0.5)$. A possible choice for Δ is $(0, 0.01, 0.02, ..., 1)$. As always when using simulated annealing, T and N_t may be more complicated to choose. We used for all simulations and data analyses the sequence $T = (10 \times 0.7^1, 10 \times 0.7^2, ..., 10 \times 0.7^{20})$ and $N_t = 100$. This leaves us with the decision of which models sizes κ to consider. A preliminary run with, say $\kappa = 1, 2, ..., 10$

and, for example, comparing the MSEs for $\kappa = 5$ and $\kappa = 10$, may imply if there is a substantial benefit when adding variables, before moving to larger models. Similarly to many algorithms, the proposed simulated annealing algorithm should be initiated from multiple points.

As illustrated using real data examples, a class of minimal models can be used to derive conclusions regarding the problem at hand. This is rarely the case that a researcher believes a one true model exists, especially in the $p > n$ regime. Therefore, we suggest to abandon the search for this "holy grail", and to analyze the class of minimal models instead. While retreating from the choice of a single model, and instead analyzing group of models, offers advantages, it may complicate standard data analysis in at least two ways, even when putting computational issues aside. The first is a practical one and concerns the question of which analyses should be carried out once a minimal class of models, or at least its sample version, is obtained. Model averaging can be used to estimate association parameters. Similarly, predictions from multiple models can be averaged. More precise predictions or estimates may be obtained by coupling the obtained models with weights, possibly using the empirical MSEs. More robust aggregation can be obtained by using medians instead of averages, to deal with the fact that each model consists of a different number of predictors. Another, less formal, data analysis strategy is to combine subject matter knowledge with wealth of many models. We have demonstrated such exploratory analyses in Section 5, by comparing the proportion of models each predictor, or multiple predictors, are included.

The second issue is a conceptual one. If we indeed quit from searching for a single model, i.e., a single group of predictors, what are our assumptions about the underlying mechanism that the data came from. Do we believe such a mechanism exists? One answer is that we do not need to think about how the data was created, but how can we describe the data. This, however, offers only a partial answer because we are focused on exploratory analysis, while model selection is of interest from inferential perspective as well. If more than one model describes the data adequately, which one is the true one? Does it matter? While most data are not created following one's computer program, disbelieving in the existence of a true model certainly gives rise to further questions.

It is well known that achieving good prediction and successful model selection simultaneously, in a reasonable computation time, is impossible, especially in the high–dimensional setting. We therefore suggested here to make a compromise. Our approach is not necessarily optimal for prediction, nor for model selection. However, it offers a data analysis method that takes into account the uncertainty in model selection, but ensures reasonable prediction accuracy. This method can be used for either prediction, parameter estimation or model selection.

Acknowledgments

We thank two anonymous reviewers for useful comments and suggestions that improved the paper. This research was supported by ISF grant 1770/15

Appendix

Appendix A. Proofs

A.1 Proof of Theorem 1

We start with the following lemma.

Lemma 4 *Assume $Y = \mu + \epsilon$ and assume also (A1)–(A4). Let $\mathcal{S}_k = \{S : |S| = k, \hat{\beta}_S = (X_S^T X_S)^{-1} X_S^T Y\}$ be the set of all models with k variables, such that $\hat{\beta}_S$, the LS estimate, is unique. Denote $S_j^\star = S \cup \{j\}, j \notin S$ for a model that includes S and additional variable j not in S. We have*

$$\max_{\substack{S \in \mathcal{S}_k \\ 1 \le j \le p}} \epsilon^T (X_{S_j^\star} \hat{\beta}_{S_j^\star} - X_S \hat{\beta}_S) = o_p(n)$$

Let ξ_j be the vector of coefficients obtained by regressing $X^{(j)}$, the j^{th} column in X, on X_S and let $\mathcal{P}_j$ be the projection operator on the subspace spanned by the part of $X^{(j)}$ which is orthogonal to the subspace spanned by X_S. That is,

$$\mathcal{P}_j = \frac{(X^{(j)} - X_S \xi_j)(X^{(j)} - X_S \xi_j)^T}{||X^{(j)} - X_S \xi_j||_2^2}.$$

Let $\hat{\beta}^j_{S_j^\star}$ be the coefficient estimate of $X^{(j)}$ in model $S_j^\star$, and let $\hat{\beta}^{-j}_{S_j^\star}$ be the coefficient estimates of the variables in S but for the model $S_j^\star$. Since $(X^{(j)} - X_S \xi_j)$ is orthogonal to the subspace spanned by the columns of X_S we have

$$\begin{aligned} X_{S_j^\star} \hat{\beta}_{S_j^\star} &= X^{(j)} \hat{\beta}^j_{S_j^\star} + X_S \hat{\beta}^{-j}_{S_j^\star} \\ &= (X^{(j)} - X_S \xi_j) \hat{\beta}^j_{S_j^\star} + X_S (\hat{\beta}^{-j}_{S_j^\star} + \xi_j \hat{\beta}^j_{S_j^\star}) \\ &= (X^{(j)} - X_S \xi_j) \hat{\beta}^j_{S_j^\star} + X_S \hat{\beta}_S \\ &= \mathcal{P}_j y + X_S \hat{\beta}_S. \end{aligned}$$

Therefore,

$$\epsilon^T (X_{S_j^\star} \hat{\beta}_{S_j^\star} - X_S \hat{\beta}_S) = \epsilon^T \mathcal{P}_j \mu + \epsilon^T \mathcal{P}_j \epsilon.$$

Now, since $||\mathcal{P}_j \mu||_2^2 \le ||\mu||_2^2 = O(n)$, we get that for all j, $\epsilon^T \mathcal{P}_j \mu = O_p(\sqrt{n})$. Next, let $Z_1, ..., Z_{p^{k+1}}$ be $N(0, \sigma^2)$ random variables and observe that the approximate size of the set $\{S_k\} \times \{1, ..., p\}$ is p^{k+1}. We have for any $a > 0$

$$P\left(\max_{\substack{S \in \mathcal{S}_k \\ 1 \le j \le p}} \frac{1}{n} \epsilon^T \mathcal{P}_j \epsilon \ge a \right) \le P\left(\max_{1 \le j \le p^{k+1}} |Z_j| \ge \sqrt{\frac{an}{\sigma^2}} \right) \le \sigma \sqrt{\frac{2(k+1)\log p + o(1)}{an}}.$$

Now, since $p = n^\alpha$ and $k = o(n/\log n)$ we get that

$$P\left(\max_{\substack{S \in \mathcal{S}_k \\ 1 \le j \le p}} \frac{1}{n} \epsilon^T \mathcal{P}_j \epsilon \ge a \right) = o(1)$$

and we are done. □

We can now move to the proof of Theorem 1. For simplicity, the notation of i as the iteration number for the current temperature t is suppressed. Note that it is enough to only consider models such that $S \cap \bar{S} = \emptyset$ and to consider $m = s_0$. Denote $Q_t(S, g, j)$ for the probability of a move in the direction of $\bar{S}$ in the next iteration, that is, the probability of choosing a variable $j \in S \cap \bar{S}^c$ and replace it with a variable $g \in S^c \cap \bar{S}$. Denote $S' = \{S/\{j\}\} \cup \{g\}$ for this new model. We have

$$\begin{aligned} &Q_t(S,g,j) \\ &\quad = Pr(S \to S') \min\left[1, \exp\left(\frac{||Y - X_S\hat{\beta}_S||_2^2 - ||Y - X_{S'}\hat{\beta}_{S'}||_2^2}{t}\right)\frac{Pr(S' \to S)}{Pr(S \to S')}\right] \end{aligned} \tag{13}$$

where $Pr(S \to S')$ is the probability of suggesting S', given current model is S. Now, since $\gamma_{min} \geq c_\gamma$ and since the maximal value in γ equals to one by definition, we have for all $S \subseteq A_\gamma$,

$$\begin{aligned} c_\gamma(h_\gamma - s_0) &\leq \sum_{u \notin S} \gamma_u \leq h_\gamma - s_0 \\ s_0 &\leq \sum_{v \in S} \frac{1}{\gamma_v} \leq \frac{s_0}{c_\gamma}. \end{aligned} \tag{14}$$

Now, by substituting (14) into (7)–(9) we get

$$\begin{aligned} Pr(S \to S') &= \frac{\gamma_g}{\sum_{u \notin S}\gamma_u}\frac{1/\gamma_j}{\sum_{v \in S}\frac{1}{\gamma_v}} \geq \frac{c_\gamma^2}{s_0(h_\gamma - s_0)}, \\ \frac{Pr(S' \to S)}{Pr(S \to S')} &= \frac{\gamma_j^2}{\gamma_g^2}\frac{\sum_{u \notin S}\gamma_u \sum_{v \in S}\frac{1}{\gamma_v}}{\sum_{u \notin S'}\gamma_u \sum_{v \in S'}\frac{1}{\gamma_v}} \geq c_\gamma^4. \end{aligned} \tag{15}$$

Next, we have

$$\begin{aligned} &\frac{1}{n}||Y - X_S\hat{\beta}_S||_2^2 - \frac{1}{n}||Y - X_{S'}\hat{\beta}_{S'}||_2^2 \\ &\quad = \frac{1}{n}\left[(Y - X_{S'}\hat{\beta}_{S'}) + (Y - X_S\hat{\beta}_S)\right]^T\left(X_{S'}\hat{\beta}_{S'} - X_S\hat{\beta}_S\right) \\ &\quad = \frac{1}{n}Y^T\left(X_{S'}\hat{\beta}_{S'} - X_S\hat{\beta}_S\right) \\ &\quad = \frac{1}{n}\mu^T\left(X_{S'}\hat{\beta}_{S'} - X_S\hat{\beta}_S\right) + \frac{1}{n}\epsilon^T\left(X_{S'}\hat{\beta}_{S'} - X_S\hat{\beta}_S\right) \\ &\quad = \frac{1}{n}\mu^T\left(X_{S'}\hat{\beta}_{S'} - X_S\hat{\beta}_S\right) + \Delta_n(S, S') \end{aligned} \tag{16}$$

where the second equality is due to $\hat{\beta}_S$ and $\hat{\beta}_{S'}$ being LS estimators. We get that an estimator in linear model achieves better (lower) sample MSE, if the correlation of the prediction using this estimator with Y is larger. Now, denote $S'' = S' \cup S$. We have

$$\Delta_n(S, S') = \frac{1}{n}\epsilon^T\left[(X_{S''}\hat{\beta}_{S''} - X_S\hat{\beta}_S) - (X_{S''}\hat{\beta}_{S''} - X_{S'}\hat{\beta}_{S'})\right]$$

and if we apply Lemma 4 twice we get that $\Delta_n(S, S') = o_p(1)$. Now, regarding the first term in (16),

$$\begin{aligned} \frac{1}{n}\mu^T\left(X_{S'}\hat{\beta}_{S'} - X_S\hat{\beta}_S\right) &= \frac{1}{n}\mu^T\left(\mathcal{P}_{\mathcal{S}'}y - \mathcal{P}_{\mathcal{S}}y\right) \\ &= \frac{1}{n}\left(||\mathcal{P}_{\mathcal{S}'}\mu||_2^2 - ||\mathcal{P}_{\mathcal{S}}\mu||_2^2\right) + \Delta_n'(S, S') \end{aligned} \tag{17}$$

where $\Delta_n'(S, S') = \frac{1}{n}\mu^T\left[\mathcal{P}_{\mathcal{S}'}\epsilon - \mathcal{P}_{\mathcal{S}}\epsilon\right]$. The content of the proof of Lemma 4 implies that $\Delta_n'(S, S') = o_p(1)$. Now, by (16) and (17) and since Assumption (B1) holds for t_0 we get that for large enough n

$$\frac{1}{n}\left(||Y - X_S\hat{\beta}_S||_2^2 - ||Y - X_{S'}\hat{\beta}_{S'}||_2^2\right) \geq 4t\log c_\gamma. \tag{18}$$

Now, by substituting (15) and (18) into (13) we get that for large enough n,

$$Q_{t_0}(S, g, j) \geq \frac{c_\gamma^2}{s_0(h_\gamma - s_0)}$$

for all $S \neq \bar{S}$, $j \in S \cap \bar{S}^c$ and $g \in S^c \cap \bar{S}$. (11) follows from this immediately since for any integer m and for all $S \neq \bar{S}$,

$$P_{t_0}^m(S'|S) \geq \min_{\substack{\{S: S\cap\bar{S}=\emptyset\} \\ j \in S\cap\bar{S}^c \\ g \in S^c\cap\bar{S}}}[Q_t(S, g, j)]^{s_0} \geq \left[\frac{c_\gamma^2}{s_0(h_\gamma - s_0)}\right]^{s_0}.$$

A.2 Proof of Proposition 2

Recall that the elastic net estimator $\hat{\beta}^{EN}$ minimizes

$$||Y - X\beta||_2^2 + \lambda_1|\beta| + \lambda_2||\beta||_2^2 \tag{19}$$

Now, WLOG assume that $\hat{\beta}^{EN}$ is a solution such that $\hat{\beta}_1^{EN} > 0$. For convenience, we omit the "EN" superscript from now on (i.e., $\hat{\beta} = \hat{\beta}^{EN}$). Define the subspace

$$\mathcal{B} := \{\beta : \forall i \neq 1, 2 \;\; \beta_i = \hat{\beta}_i, \;\; \beta_1 = \tau\hat{\beta}_1, \;\; \beta_2 = (1 - \tau)\hat{\beta}_1\}. \tag{20}$$

If the minimum of (19) over $\mathcal{B}$ is obtained for $\tau \neq 1$, then given that $X^{(1)}$ is part of the elastic net model, predictor $X^{(2)}$ is also part of this model.

WLOG, write down X as $X = (X_{(12)} \quad X_{-(12)})$ where $X_{(12)} = (X^{(1)} \quad X^{(2)})$ are the first two columns of X and $X_{-(12)}$ are the rest of its columns. Similarly, we have $\beta^T = (\beta_{(12)}^T \quad \beta_{-(12)}^T)$ where $\beta_{(12)}$ is the first two entries in the vector β and $\beta_{-(12)}$ is the rest of the vector. Define $\tilde{Y} = Y - X_{-(12)}\beta_{-(12)}$. We can rewrite (19) as

$$||\tilde{Y} - X_{(12)}\beta_{(12)}||_2^2 + \lambda_1(|\beta_{-(12)}| + |\beta_{(12)}|) + \lambda_2(||\beta_{-(12)}||_2^2 + ||\beta_{(12)}||_2^2) \tag{21}$$

If the minimum of (21), on $\mathcal{B}$, is achieved at $0 < \tau^* < 1$ then $\hat{\beta}_2$ must be non zero. Minimizing (21) on $\mathcal{B}$ is essentially minimizing

$$-2\tilde{Y}^T X_{(12)}\beta_{(12)} + ||X_{(12)}\beta_{(12)}||_2^2 + \lambda_2||\beta_{(12)}||_2^2 \tag{22}$$

on $\mathcal{B}$. Now, by the definition of $\mathcal{B}$ in (20) and using simple algebra we get that (22) equals to

$$2\left[\hat{\beta}_1\tilde{Y}^T\left(\tau(X^{(2)}-X^{(1)})-X^{(2)}\right)-\hat{\beta}_1^2\tau(1-\tau)(1-\rho)+\lambda_2\hat{\beta}_1^2\left(\frac{1}{2}-\tau(1-\tau)\right)\right].$$

This is a quadratic function of τ, and by equating its derivative to zero we get that

$$\tau^*=\frac{1}{2}-\frac{\tilde{Y}^T(X^{(2)}-X^{(1)})}{2\hat{\beta}_1(\lambda_2+1-\rho)}$$

is the minimizer of (19) (the coefficient of the quadratic term is positive). Note that for $X^{(2)}=X^{(1)}$ we get the expected $\tau^*=\frac{1}{2}$ solution. Note also that this reveals no information regarding the lasso where $\lambda_2=0$. Next, we get that $0<\tau^*<1$ if

$$\left|\frac{\tilde{Y}^T(X^{(2)}-X^{(1)})}{\hat{\beta}_1(\lambda_2+1-\rho)}\right|<1. \tag{23}$$

Since $||X^{(2)}-X^{(1)}||_2^2=2(1-\rho)$ we have

$$|\tilde{Y}^T(X^{(2)}-X^{(1)})|\leq\sum_{i=1}^{n}|\tilde{Y}_i||X_i^{(2)}-X_i^{(1)}|\leq||\tilde{Y}||_2\sqrt{2(1-\rho)},$$

using the triangle inequality and then Cauchy–Schwartz inequality. It is assumed that $\hat{\beta}_1\geq c_\beta>0$ and it is known that $||\tilde{Y}||_2\leq||Y||_2$. Therefore, we may rewrite (23) as

$$\frac{\sqrt{2}||Y||_2\sqrt{1-\rho}}{c_\beta(\lambda_2+1-\rho)}<1.$$

Now, Denote $t=\sqrt{1-\rho}, u=\frac{||Y||_2}{c_\beta}$, we have

$$t^2-\sqrt{2}ut+\lambda_2>0.$$

For $\lambda_2>\frac{1}{2}u^2$, we get the result we want for all ρ's. For $\lambda_2<\frac{u^2}{2}$ we have

$$\sqrt{1-\rho}>\frac{1}{\sqrt{2}}(u+\sqrt{u^2-2\lambda_2}), \tag{24}$$

$$\sqrt{1-\rho}<\frac{1}{\sqrt{2}}(u-\sqrt{u^2-2\lambda_2}). \tag{25}$$

The RHS of (24) is larger than 1 if $\lambda_2<\sqrt{2}u-1$. That is, there is no suitable ρ for this case. The RHS of (25) is always positive, and for the same condition $\lambda_2<\sqrt{2}u-1$, it also meaningful, i.e., $(u-\sqrt{u^2-2\lambda_2})<\sqrt{2}$ and in terms of ρ,

$$\rho>1-\frac{1}{2}(u-\sqrt{u^2-2\lambda_2})^2$$

or alternatively,

$$\rho>1-\frac{u^2}{2}\left(1-\sqrt{1-\frac{2\lambda_2}{u}}\right)^2$$

and by Taylor expansion for $2\lambda_2/u$ we get

$$\rho>1-\frac{\lambda_2^2}{2u^2}$$

■

A.3 Proof of Theorem 3

The proof is similar to the proof of Proposition 2. Let $\hat{\beta}=\hat{\beta}^{EN}$ be the elastic net estimator and denote $\hat{\beta}_M$ for the values in $\hat{\beta}$ corresponding to the set of predictors M. We can partition the set of potential predictors $\{1,2,...,p\}$ to four disjoint subsets: $M^{(-)}$; $M_1\cap M_2^c$; $M_1^c\cap M_2$ and $M_1\cap M_2$. We replace (20) with

$$\mathcal{B}:=\{\beta:\beta_{M^{(-)}}=\hat{\beta}_{M^{(-)}}\ \ \beta_{M_1\cap M_2}=\hat{\beta}_{M_1\cap M_2}, \tag{26}$$

$$\beta_{M_1\cap M_2^c}=\tau\hat{\beta}_{M_1\cap M_2^c},\ \ \beta_{M_1^c\cap M_2}=(1-\tau)\Theta'\hat{\beta}_{M_1\cap M_2^c}\}. \tag{27}$$

where β_M is defined as the values in $\hat{\beta}$ corresponding to the set M and Θ' is the matrix of coefficients obtained from regressing $X_{M_1\cap M_2^c}$ on $X_{M_1^c\cap M_2}$. We define Θ to be an augmented version of Θ', which we obtain by regressing X_{M_1} on X_{M_2}. That is,

$$X_{M_2}\Theta=\mathcal{P}_{M_2}X_{M_1} \tag{28}$$

Note that on $\mathcal{B}$,

$$X\beta=\tilde{X}\hat{\beta}_{M^{(-)}}+\tau X_{M_1}\hat{\beta}_{M_1}+(1-\tau)X_{M_2}\Theta\hat{\beta}_{M_1}$$

Recalling that $\tilde{Y}=Y-\tilde{X}\hat{\beta}_{M^{(-)}}$, minimizing (3) on $\mathcal{B}$ is equivalent to minimize

$$\begin{aligned}||\tilde{Y}-\tau X_{M_1}\hat{\beta}_{M_1}-(1-\tau)X_{M_2}\Theta\hat{\beta}_{M_1}||_2^2+\lambda_1[\tau||\hat{\beta}_{M_1}||_1+(1-\tau)||\Theta\hat{\beta}_{M_1}||_1]\\+\lambda_2[\tau^2||\hat{\beta}_{M_1}||_2^2+(1-\tau)^2||\Theta\hat{\beta}_{M_1}||_2^2]\end{aligned} \tag{29}$$

as a function of τ. Using a first–order condition and substituting (28) we find that (29) is minimized for

$$\tau^*=\frac{-(\tilde{Y}-\mathcal{P}_{M_2}X_{M_1}\hat{\beta}_{M_1})^T(I-\mathcal{P}_{M_2})X_{M_1}\hat{\beta}_{M_1}+\frac{\lambda_1}{2}(||\hat{\beta}_{M_1}||_1-||\Theta\hat{\beta}_{M_1}||_1)-\lambda_2||\Theta\hat{\beta}_{M_1}||_2^2}{||(I-\mathcal{P}_{M_2})X_{M_1}\hat{\beta}_{M_1}||_2^2-\lambda_2||\hat{\beta}_{M_1}||_2^2-\lambda_2||\Theta\hat{\beta}_{M_1}||_2^2}. \tag{30}$$

Before we continue, note that if $X_2=X_1$ then Θ is the identity matrix and $\mathcal{P}_{M_2}X_{M_1}=X_1$. Substituting these facts into (30), we get that $\tau^*=\frac{1}{2}$ as one might expect. Same result is obtained for the case $M_2\subseteq M_1$.

As it can be seen in (26), the coordinates of $\hat{\beta}_{M_2}$ are all different than zero if $\tau^*<1$. Now, since $\mathcal{P}_{M_2}(I-\mathcal{P}_{M_2})=0$ we get that $\tau^*<1$ if

$$\begin{aligned}-\tilde{Y}^T(I-\mathcal{P}_{M_2})X_{M_1}\hat{\beta}_{M_1}+||(I-\mathcal{P}_{M_2})X_{M_1}\hat{\beta}_{M_1}||_2^2-\frac{\lambda_1}{2}||\Theta\hat{\beta}_{M_1}||_1\\>-\frac{\lambda_1}{2}||\hat{\beta}_{M_1}||_1-\lambda_2||\hat{\beta}_{M_1}||_2^2\end{aligned}$$

which is certainly true if

$$\tilde{Y}^T\mathcal{P}_{M_2}X_{M_1}\hat{\beta}_{M_1}-\frac{\lambda_1}{2}||\Theta\hat{\beta}_{M_1}||_1>-\frac{\lambda_1}{2}||\hat{\beta}_{M_1}||_1-\lambda_2||\hat{\beta}_{M_1}||_2^2+\tilde{Y}^TX_{M_1}\hat{\beta}_{M_1} \tag{31}$$

which is true if the condition in (12) is fulfilled for the appropriate c_1. ■

Appendix B. Supplementary table for Section 5.2

Predictor	Description
prec	Mean annual precipitation in inches
jant	Mean January temperature in degrees F
jult	Mean July temperature in degrees F
age65	Percentage of population aged 65 or older
pphs	Population per household
educ	Median school years completed by those over 22
h **facl**	Percentage of housing units which are sound and with all facilities
dens	Population per square mile in urbanized areas
nwht	Percentage of non–white population in urbanized areas
wtcl	Percentage of employed in white collar occupations
linc	Percentage of families with income $<$ 3,000 dollars in urbanized areas
HC	Relative pollution potential of hydrocarbon
NOx	Relative pollution potential of nitric oxides
SUL	Relative pollution potential of sulfur dioxide
hum	Annual average percentage of relative humidity at 1pm

Table 4: Potential predictors for mortality rate in Section 5.2

References

Hirotugu Akaike. A new look at the statistical model identification. *IEEE transactions on automatic control*, 19(6):716–723, 1974.

Peter J Bickel and Mu Cai. Discussion of Sara van de Geer: Generic chaining and the ℓ_1 penalty. *Journal of Statistical Planning and Inference*, 143(6):1013–1018, 2012.

Peter J Bickel, Yaacov Ritov, and Alexandre B Tsybakov. Simultaneous analysis of lasso and dantzig selector. *Annals of Statistics*, 37(4):1705–1732, 2009.

Peter J Bickel, Yaacov Ritov, and Alexandre B Tsybakov. Hierarchical selection of variables in sparse high-dimensional regression. In *Borrowing strength: theory powering applications–a Festschrift for Lawrence D. Brown*, volume 6, pages 56–69. Institute of Mathematical Statistics, 2010.

Jacob Bien, Jonathan Taylor, and Robert Tibshirani. A lasso for hierarchical interactions. *Annals of Statistics*, 41(3):1111–1141, 2013.

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

S. P. Brooks and B. J. T. Morgan. Optimization using simulated annealing. *The Statistician*, pages 241–257, 1995.

S. P. Brooks, N Friel, and R King. Classical model selection via simulated annealing. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(2):503–520, 2003.

Peter Bühlmann, Markus Kalisch, and Lukas Meier. High-dimensional statistics with a view toward applications in biology. *Annual Review of Statistics and Its Application*, 1: 255–278, 2014.

Florentina Bunea, AlexandreB. Tsybakov, and MartenH. Wegkamp. Aggregation and Sparsity Via $\ell 1$ Penalized Least Squares. In *Learning Theory*, volume 4005 of *Lecture Notes in Computer Science*, pages 379–391. Springer Berlin Heidelberg, 2006. URL `http://dx.doi.org/10.1007/11776420_29`.

Florentina Bunea, Alexandre Tsybakov, and Marten Wegkamp. Sparsity oracle inequalities for the lasso. *Electronic Journal of Statistics*, 1:169–194, 2007.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1–22, 2010.

Eitan Greenshtein and Ya'Acov Ritov. Persistence in high-dimensional linear predictor selection and the virtue of overparametrization. *Bernoulli*, 10(6):971–988, 2004.

W Keith Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

Nils Lid Hjort and Gerda Claeskens. Frequentist model average estimators. *Journal of the American Statistical Association*, 98(464):879–899, 2003.

Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

Jennifer A Hoeting, David Madigan, Adrian E Raftery, and Chris T Volinsky. Bayesian model averaging: A tutorial. *Statistical Science*, pages 382–401, 1999.

Jian Huang, Shuangge Ma, and Cun-Hui Zhang. Adaptive lasso for sparse high-dimensional regression models. *Statistica Sinica*, 18(4):1603, 2008.

Jinzhu Jia and Bin Yu. On model selection consistency of the Elastic Net when $p >> n$. *Statistica Sinica*, 20:595–611, 2010.

Scott Kirkpatrick, D. Jr. Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

Gary C McDonald and Richard C Schwing. Instabilities of regression estimates relating air pollution to mortality. *Technometrics*, 15(3):463–481, 1973.

Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. *Annals of Statistics*, 34(3):1436–1462, 2006.

Nicolai Meinshausen and Peter Bühlmann. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):417–473, 2010.

Nicolai Meinshausen and Bin Yu. Lasso-type recovery of sparse representations for high-dimensional data. *Annals of Statistics*, 37:246–270, 2009.

Nicolai Meinshausen, Lukas Meier, and Peter Bühlmann. P-values for high-dimensional regression. *Journal of the American Statistical Association*, 104(488), 2009.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21:1087, 1953.

Robert B O'Hara and Mikko J Sillanpää. A review of Bayesian variable selection methods: what, how and which. *Bayesian analysis*, 4(1):85–117, 2009.

Philippe Rigollet and Alexandre B Tsybakov. Sparse estimation by exponential weighting. *Statistical Science*, 27(4):558–575, 2012.

Gideon Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

Tingni Sun and Cun-Hui Zhang. Scaled sparse linear regression. *Biometrika*, 71:879–898, 2012.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Joel A Tropp. Greed is good: Algorithmic results for sparse approximation. *Information Theory, IEEE Transactions on*, 50(10):2231–2242, 2004.

Hansheng Wang, Bo Li, and Chenlei Leng. Shrinkage tuning parameter selection with a diverging number of parameters. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):671–683, 2009.

Ming Yuan and Yi Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

Cun-Hui Zhang and Jian Huang. The sparsity and bias of the lasso selection in high-dimensional linear regression. *Annals of Statistics*, 36(4):1567–1594, 2008.

Tong Zhang. On the consistency of feature selection using greedy least squares regression. *Journal of Machine Learning Research*, 10(3), 2009.

Peng Zhao and Bin Yu. On model selection consistency of lasso. *The Journal of Machine Learning Research*, 7:2541–2563, 2006.

Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429, 2006.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.

Journal of Machine Learning Research 18 (2017) 1-5 Submitted 5/16; Revised 11/16; Published 3/17

Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA

Lars Kotthoff larsko@cs.ubc.ca
Chris Thornton cwthornt@cs.ubc.ca
Holger H. Hoos hoos@cs.ubc.ca
Frank Hutter fh@cs.uni-freiburg.de
Kevin Leyton-Brown kevinlb@cs.ubc.ca
Department of Computer Science
University of British Columbia
2366 Main Mall, Vancouver, B.C. V6T 1Z4 Canada

Editor: Geoff Holmes

Abstract

WEKA is a widely used, open-source machine learning platform. Due to its intuitive interface, it is particularly popular with novice users. However, such users often find it hard to identify the best approach for their particular dataset among the many available. We describe the new version of *Auto-WEKA*, a system designed to help such users by automatically searching through the joint space of WEKA's learning algorithms and their respective hyperparameter settings to maximize performance, using a state-of-the-art Bayesian optimization method. Our new package is tightly integrated with WEKA, making it just as accessible to end users as any other learning algorithm.

Keywords: Hyperparameter Optimization, Model Selection, Feature Selection

1. The Principles Behind Auto-WEKA

The WEKA machine learning software (Hall et al., 2009) puts state-of-the-art machine learning techniques at the disposal of even novice users. However, such users do not typically know how to choose among the dozens of machine learning procedures implemented in WEKA and each procedure's hyperparameter settings to achieve good performance.

Auto-WEKA[1] addresses this problem by treating all of WEKA as a single, highly parametric machine learning framework, and using Bayesian optimization to find a strong instantiation for a given dataset. Specifically, it considers the combined space of WEKA's learning algorithms $\mathcal{A} = \{A^{(1)}, \ldots, A^{(k)}\}$ and their associated hyperparameter spaces $\Lambda^{(1)}, \ldots, \Lambda^{(k)}$ and aims to identify the combination of algorithm $A^{(j)} \in \mathcal{A}$ and hyperparameters $\lambda \in \Lambda^{(j)}$ that minimizes cross-validation loss,

$$A^*_{\lambda^*} \in \operatorname*{argmin}_{A^{(j)} \in \mathcal{A}, \lambda \in \Lambda^{(j)}} \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}\left(A^{(j)}_{\lambda}, \mathcal{D}^{(i)}_{\text{train}}, \mathcal{D}^{(i)}_{\text{test}}\right),$$

where $\mathcal{L}(A_\lambda, \mathcal{D}^{(i)}_{\text{train}}, \mathcal{D}^{(i)}_{\text{test}})$ denotes the loss achieved by algorithm A with hyperparameters λ when trained on $\mathcal{D}^{(i)}_{\text{train}}$ and evaluated on $\mathcal{D}^{(i)}_{\text{test}}$. We call this the *combined algorithm selection and hyperparameter optimization (CASH)* problem. CASH can be seen as a blackbox function optimization problem: determining $\operatorname{argmin}_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} f(\boldsymbol{\theta})$, where each configuration $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ comprises the choice of algorithm $A^{(j)} \in \mathcal{A}$ and its hyperparameter settings $\lambda \in \Lambda^{(j)}$. In this formulation, the hyperparameters of algorithm $A^{(j)}$ are *conditional* on $A^{(j)}$ being selected. For a given $\boldsymbol{\theta}$ representing algorithm $A^{(j)} \in \mathcal{A}$ and hyperparameter settings $\lambda \in \Lambda^{(j)}$, $f(\boldsymbol{\theta})$ is then defined as the cross-validation loss $\frac{1}{k} \sum_{i=1}^{k} \mathcal{L}(A^{(j)}_{\lambda}, \mathcal{D}^{(i)}_{\text{train}}, \mathcal{D}^{(i)}_{\text{test}})$.[2]

Bayesian optimization (see, e.g., Brochu et al., 2010), also known as sequential model-based optimization, is an iterative method for solving such blackbox optimization problems. In its n-th iteration, it fits a probabilistic model based on the first $n-1$ function evaluations $\langle \boldsymbol{\theta}_i, f(\boldsymbol{\theta}_i) \rangle_{i=1}^{n-1}$, uses this model to select the next $\boldsymbol{\theta}_n$ to evaluate (trading off exploration of new parts of the space *vs* exploitation of regions known to be good) and evaluates $f(\boldsymbol{\theta}_n)$. While Bayesian optimization based on Gaussian process models is known to perform well for low-dimensional problems with numerical hyperparameters (see, e.g., Snoek et al., 2012), tree-based models have been shown to be more effective for high-dimensional, structured, and partly discrete problems (Eggensperger et al., 2013), such as the highly conditional space of WEKA's learning algorithms and their corresponding hyperparameters we face here.[3] Thornton et al. (2013) showed that tree-based Bayesian optimization methods yielded the best performance in Auto-WEKA, with the random-forest-based SMAC (Hutter et al., 2011) performing better than the tree-structured Parzen estimator, TPE (Bergstra et al., 2011). Auto-WEKA uses SMAC to determine the classifier with the best performance on the given data.

2. Auto-WEKA 2.0

Since the initial release of a usable research prototype in 2013, we have made substantial improvements to the Auto-WEKA package described by Thornton et al. (2013). At a prosaic level, we have fixed bugs, improved tests and documentation, and updated the software to work with the latest versions of WEKA and Java. We have also added four major features.

First, we now support regression algorithms, expanding Auto-WEKA beyond its previous focus on classification (starred entries in Fig. 1). Second, we now support the optimization of all performance metrics WEKA supports. Third, we now natively support parallel runs (on a single machine) to find good configurations faster and save the N best configurations of each run instead of just the single best. Fourth, Auto-WEKA 2.0 is now fully integrated with WEKA. This is important, because the crux of Auto-WEKA lies in its simplicity: providing a push-button interface that requires no knowledge about the available learning algorithms or their hyperparameters, asking the user to provide, in addition to the dataset to be processed, only a memory bound (1 GB by default) and the overall time

1. Thornton et al. (2013) first introduced Auto-WEKA and empirically demonstrated state-of-the-art performance. Here we describe an improved and more broadly accessible implementation of Auto-WEKA, focussing on usability and software design.

2. In fact, on top of machine learning algorithms and their respective hyperparameters, we also include attribute selection methods and their respective hyperparameters in the configurations $\boldsymbol{\theta}$, thereby jointly optimizing over their choice and the choice of algorithms.

3. Conditional dependencies can also be accommodated in the Gaussian process framework (Hutter and Osborne, 2013; Swersky et al., 2013), but currently, tree-based methods achieve better performance.

©2017 Lars Kotthoff, Chris Thornton, Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-261.html.

Learners

BayesNet	2	Logistic	1	REPTree*	6
DecisionStump*	0	M5P	4	SGD*	5
DecisionTable*	4	M5Rules	4	SimpleLinearRegression*	0
GaussianProcesses*	10	MultilayerPerceptron*	8	SimpleLogistic	5
IBk*	5	NaiveBayes	2	SMO	11
J48	9	NaiveBayesMultinomial	0	SMOreg*	13
JRip	4	OneR	1	VotedPerceptron	3
KStar*	3	PART	4	ZeroR*	0
LinearRegression*	3	RandomForest	7		
LMT	9	RandomTree*	11		

Ensemble Methods

Stacking	2	Vote	2		

Meta-Methods

LWL	5	AttributeSelectedClassifier	2	RandomSubSpace	3
AdaBoostM1	6	Bagging	4		
AdditiveRegression	4	RandomCommittee	2		

Attribute Selection Methods

BestFirst	2	GreedyStepwise	4		

Figure 1: Learners and methods supported by Auto-WEKA 2.0, along with number of hyperparameters $|\Lambda|$. Every learner supports classification; starred learners also support regression.

budget available for the entire learning process.[4] The overall budget is set to 15 minutes by default to accommodate impatient users; longer runs allow the Bayesian optimizer to search the space more thoroughly; we recommend at least several hours for production runs.

The usability of the earlier research prototype was hampered by the fact that users had to download Auto-WEKA manually and run it separately from WEKA. In contrast, Auto-WEKA 2.0 is now available through WEKA's package manager. Users do not need to install software separately; everything is included in the package and installed automatically upon request. After installation, Auto-WEKA 2.0 can be used in two different ways:

1. *As a meta-classifier:* Auto-WEKA can be run like any other machine learning algorithm in WEKA: via the GUI, the command-line interface, or the public API. Figure 2 shows how to run it from the command line.
2. *Through the Auto-WEKA tab:* This provides a customized interface that hides some of the complexity. Figure 3 shows the output of an example run.

Source code for Auto-WEKA is hosted on GitHub (`https://github.com/automl/autoweka`) and is available under the GPL license (version 3). Releases are published to the WEKA package repository and available both through the WEKA package manager and from the Auto-WEKA project website (`http://automl.org/autoweka`). A manual describes how to use the WEKA package and gives a high-level overview for developers; we also provide lower-level Javadoc documentation. An issue tracker on GitHub, JUnit tests and the continuous integration system Travis facilitate bug tracking and correctness of the code. Since its release on March 1, 2016, Auto-WEKA 2.0 has been downloaded more than 15 000 times, with an average of about 400 downloads per week.

4. Internally, to avoid using all its budget for executing a single slow learner, Auto-WEKA limits individual runs of any learner to 1/12 of the overall budget; it further limits feature search to 1/60 of the budget.


```
java -cp autoweka.jar weka.classifiers.meta.AutoWEKAClassifier
    -timeLimit 5 -t iris.arff -no-cv
```

Figure 2: Command-line call for running Auto-WEKA with a time limit of 5 minutes on training dataset `iris.arff`. Auto-WEKA performs cross-validation internally, so we disable WEKA's cross-validation (`-no-cv`). Running with `-h` lists the available options.

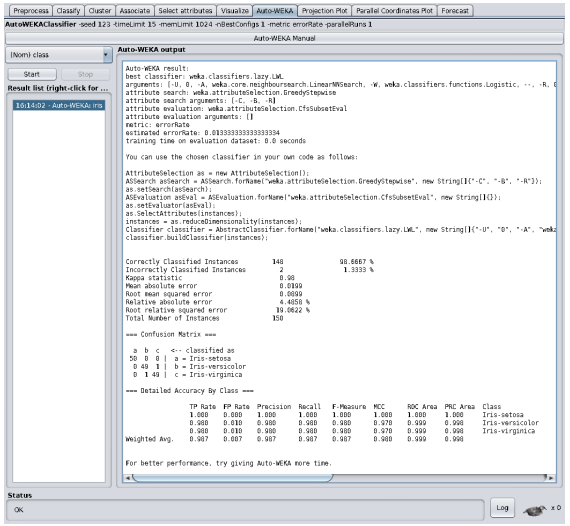


Figure 3: Example Auto-WEKA run on the iris dataset. The resulting best classifier along with its parameter settings is printed first, followed by its performance. While Auto-WEKA runs, it logs to the status bar how many configurations it has evaluated so far.

3. Related Implementations

Auto-WEKA was the first method to use Bayesian optimization to automatically instantiate a highly parametric machine learning framework at the push of a button. This *automated machine learning (AutoML)* approach has recently also been applied to Python and scikit-learn (Pedregosa et al., 2011) in Auto-WEKA's sister package, *Auto-sklearn* (Feurer et al., 2015). Auto-sklearn uses the same Bayesian optimizer as Auto-WEKA, but comprises a smaller space of models and hyperparameters, since scikit-learn does not implement as many different machine learning techniques as WEKA; however, Auto-sklearn includes additional meta-learning techniques.

It is also possible to optimize hyperparameters using WEKA's own grid search and MultiSearch packages. However, these packages only permit tuning one learner and one filtering method at a time. Grid search handles only one hyperparameter. Furthermore, hyperparameter names and possible values have to be specified by the user.

References

J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems 24 (NIPS'11)*, pages 2546–2554, 2011.

E. Brochu, V. Cora, and N. de Freitas. A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *Computing Research Repository (arXiv)*, abs/1012.2599, 2010.

K. Eggensperger, M. Feurer, F. Hutter, J. Bergstra, J. Snoek, H. Hoos, and K. Leyton-Brown. Towards an empirical foundation for assessing Bayesian optimization of hyperparameters. In *NIPS Workshop on Bayesian Optimization (BayesOpt'13)*, 2013.

M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter. Efficient and Robust Automated Machine Learning. In *Advances in Neural Information Processing Systems 28 (NIPS'15)*, pages 2944–2952, 2015.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA Data Mining Software: An Update. *SIGKDD Explor. Newsl.*, 11(1):10–18, Nov. 2009. ISSN 1931-0145.

F. Hutter and M. Osborne. A Kernel for Hierarchical Parameter Spaces. *Computing Research Repository (arXiv)*, abs/1310.5738, Oct. 2013.

F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential Model-Based Optimization for General Algorithm Configuration. In *Learning and Intelligent OptimizatioN Conference (LION 5)*, pages 507–523, 2011.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems 25 (NIPS'12)*, pages 2951–2959, 2012.

K. Swersky, D. Duvenaud, J. Snoek, F. Hutter, and M. Osborne. Raiders of the lost architecture: Kernels for Bayesian optimization in conditional parameter spaces. In *NIPS Workshop on Bayesian Optimization (BayesOpt'13)*, 2013.

C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *19th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'13)*, 2013.

Journal of Machine Learning Research 18 (2017) 1-5 Submitted 6/16; Revised 1/17; Published 4/17

POMDPs.jl: A Framework for Sequential Decision Making under Uncertainty

Maxim Egorov MEGOROV@STANFORD.EDU
Zachary N. Sunberg ZSUNBERG@STANFORD.EDU
Edward Balaban EBALABAN@STANFORD.EDU
Tim A. Wheeler WHEELERT@STANFORD.EDU
Jayesh K. Gupta JKG@CS.STANFORD.EDU
Mykel J. Kochenderfer MYKEL@STANFORD.EDU
Department of Aeronautics and Astronautics
Stanford University
Stanford, CA 94305, USA

Editor: Antti Honkela

Abstract

POMDPs.jl is an open-source framework for solving Markov decision processes (MDPs) and partially observable MDPs (POMDPs). POMDPs.jl allows users to specify sequential decision making problems with minimal effort without sacrificing the expressive nature of POMDPs, making this framework viable for both educational and research purposes. It is written in the Julia language to allow flexible prototyping and large-scale computation that leverages the high-performance nature of the language. The associated JuliaPOMDP community also provides a number of state-of-the-art MDP and POMDP solvers and a rich library of support tools to help with implementing new solvers and evaluating the solution results. The most recent version of POMDPs.jl, the related packages, and documentation can be found at `https://github.com/JuliaPOMDP/POMDPs.jl`.

Keywords: POMDP, MDP, sequential decision making, Julia, open-source

1. Introduction

Recent advances in algorithms and hardware make partially observable Markov decision processes (POMDPs) a viable model for a variety of applications ranging from aircraft collision avoidance (Wolf and Kochenderfer, 2011) to spoken dialog systems (Young et al., 2013). In general, POMDPs can be applied to a wide variety of problems that include state and dynamic uncertainties as well as continuous and discrete domains (Kochenderfer, 2015). State-of-the-art algorithms like POMCP (Silver and Veness, 2010) and MCVI (Bai et al., 2010) can solve problems with millions of states and with continuous state spaces. However, working with POMDPs is challenging due to their probabilistic nature and the complexity of the algorithms used to solve them.

POMDPs.jl is an interface for solving POMDPs written in the Julia programming language (Bezanson et al., 2012). It is designed to support users in three different roles: 1) defining problems, 2) creating solvers, and 3) running experiments. The goals of POMDPs.jl are to allow users in these roles to easily build on code written by others through a single unified interface, simplify the process of benchmarking algorithms against one another, and encourage growth of the software ecosystem through open-source contributions. This ecosystem is organized as an online community, JuliaPOMDP. The centerpiece of the community is the POMDPs.jl package — a unified, implementation-free interface providing access to a growing collection of state-of-the-art solvers, benchmark problems, and libraries of support tools. An outline of the POMDPs.jl architecture and its core concepts are shown in Fig. 1. POMDPs.jl provides abstract types corresponding to each of the core concepts and a list of functions that defines a standard interface. Users implement concrete types that represent problems or solvers and specify their behavior by defining methods.

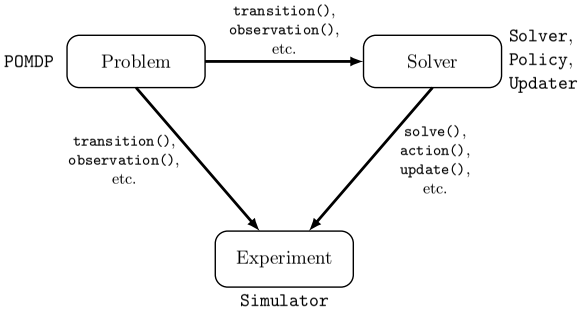


Figure 1: POMDPs.jl framework architecture. The three main concepts in the architecture: problem, solver and experiment are supported by abstract types in the framework as well as necessary function implementations.

2. Existing Frameworks and Related Work

There are a number of frameworks that are capable of solving decision making problems in fully observable settings, but few accommodate partially observable domains. The most closely related frameworks to POMDPs.jl are APPL (Kurniawati et al.), AI-Toolbox (Bargiacchi), and ZMDP (Smith) in that they can handle problems with partial observability. However, these frameworks either support only a generative model problem definition (APPL for MCVI) or explicit problem definitions using probability tables (APPL for SARSOP, AI-Toolbox and ZMDP), but not both. This limits the flexibility of the frameworks to handle both discrete and continuous domains. While the fact that these frameworks are all written in C++ leads to performant code, it makes prototyping slow and challenging for new users. It is also comparatively more difficult to extend these frameworks to new domains and implement new algorithms due to their complicated hierarchies and lack of modularity in the source code. A number of reinforcement learning frameworks exist that are designed to be easily extendable such as BURLAP (Diuk et al., 2008), RLPy (Geramifard et al., 2015), and rllab (Duan et al., 2016). However, these frameworks primarily cater to problems that are fully observable.

©2017 Maxim Egorov, Zachary N. Sunberg, Edward Balaban, Tim A. Wheeler, Jayesh K. Gupta, and Mykel J. Kochenderfer.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v18/16-300.html`.

3. Why POMDPs.jl?

POMDPs.jl aims to simplify the task of writing problems, implementing solvers, and running experiments in the context of POMDPs. The advantages of POMDPs.jl are as follows:
Simplicity: The POMDPs.jl interface provides a minimal set of types and functions necessary to define a problem, a solver, and experiments in a partially observable setting. The POMDPs.jl interface contains 10 abstract types and 37 functions, of which, each user will define a subset for their problem. For example, if a user wants to solve an MDP rather than a POMDP, they do not need to implement functions involving observations. Examples of function requirements for the MCTS and SARSOP solvers are shown in Table 1.

Table 1: A section of the function table for two solvers that use POMDPs.jl

functions	`states`	`actions`	`observations`	`transition`	`observation`	`reward`	⋯
MCTS	n/a	✓	n/a	✓	n/a	✓	⋯
SARSOP	✓	✓	✓	✓	✓	✓	
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Expressiveness: The POMDPs.jl interface provides flexibility to handle problems that are fully or partially observable, problems that are continuous or discrete, and problems with infinite and finite horizons. Solvers may leverage explicitly-defined distributions, if they are available, or only use samples from a generative model. Moreover, all POMDPs.jl problems can be defined in code. Julia makes it easy to prototype new code, removing the need for problem specification or solver configuration files written in another format.
Extensibility: The interface allows new algorithms to be implemented with minimal effort. By providing an expressive interface to a POMDP model, POMDPs.jl allows algorithm writers to access any component of the model with simple function calls. Since all solvers are implemented within a unified framework, POMDPs.jl also allows for standardized benchmarking across algorithms. It is also simple to call programs written in other languages from Julia, such as Python and C, allowing existing solvers to be wrapped using POMDPs.jl.
Usability: A number of ready-to-use solvers supporting the POMDPs.jl interface are available from the JuliaPOMDP community. To install the SARSOP solver, for example, the user only needs to run `POMDPs.add("SARSOP")`, and the package manager will download and compile any dependencies on all three major platforms — Linux, Windows, and OS X.

4. Concepts and Use Case Examples

This section describes the three main concepts outlined in Fig. 1, namely the problem, the solver, and the experiment. The POMDPs.jl interface consists of several abstract types and a set of functions that support these concepts.
Problem: The problem defines the MDP or POMDP model to be solved. Below is a definition for the Tiger POMDP (Kaelbling et al., 1998).

```
immutable TigerPOMDP <: POMDP{Bool, Int64, Bool}
    p_correct::Float64      # probability of hearing the tiger correctly
    discount::Float64       # discount factor
end
```


The `TigerPOMDP` type inherits from the abstract `POMDP` type (part of the POMDPs.jl interface), which is parametrized by the state, action, and observation types. In this case, the state is represented by the native `Bool` type, but, in general, these types may be user-defined. The same flexibility is available for the other components of the problem, giving the user the flexibility to define their problem in the form of their choosing.
Solver: A solver implementation typically requires three type definitions: a `Solver` that contains the parameters that define solver behavior, a `Policy` that defines a mapping from beliefs to actions, and an `Updater` that defines how the belief is updated with new observations. An example of a QMDP (Littman et al., 1995) policy type is shown below.

```
type QMDPPolicy{Action} <: Policy
    alphas::Matrix{Float64}     # policy alpha vectors
    action_map::Vector{Action} # indices to actions
    pomdp::POMDP                # POMDP model
end
```

A solver implementation usually includes three top-level functions: `solve`, that creates a policy for the problem given the solver, `action`, that emits an action for a belief based on the policy, and `update`, that updates the belief based on the action taken and the observation received, given the updater.
Experiment: Experiments combine problems and solvers to evaluate the quality of a solver policy. For example, the experimenter might create a simulator type, `Sim`, and a corresponding `simulate` function (an important part of the main loop is shown below).

```
function simulate(sim::Sim, pomdp, policy, updater, initial_dist)
   ...
   for t in 1:sim.max_steps
       a = action(policy, b)
       sp = rand(sim.rng, transition(pomdp, s, a))
       r_total += discount(pomdp)^t*reward(pomdp, s, a, sp)
       o = rand(sim.rng, observation(pomdp, s, a, sp))
       b = update(updater, b, a, o)
       ...
```

A simulation can then be run as follows:

```
using QMDP, POMDPModels, POMDPToolbox # import JuliaPOMDP packages
pomdp = TigerPOMDP()                  # initialize the tiger problem
solver = QMDPSolver()                 # initialize QMDP solver
policy = solve(solver, pomdp)         # compute a policy
r = simulate(Sim(), pomdp, policy, updater(policy), DiscreteBelief(2))
```

5. Conclusion

POMDPs.jl is a high level interface for working with POMDPs that allows users to easily define their problems, create new solvers, and run experiments. This manuscript provides a brief overview of the framework and its features; extensive documentation and examples for this software package can be found at `https://github.com/JuliaPOMDP/POMDPs.jl`. Future work includes integrating Julia's shared memory parallelism into sampling based solvers to improve computational efficiency, as well as introducing support for reinforcement learning algorithms.

References

Haoyu Bai, David Hsu, Wee Sun Lee, and Vien A Ngo. Monte Carlo value iteration for continuous-state POMDPs. In *Algorithmic Foundations of Robotics IX*, pages 175–191. Springer, 2010.

Eugenio Bargiacchi. AI-Toolbox. `https://github.com/Svalorzen/AI-Toolbox`.

Jeff Bezanson, Stefan Karpinski, Viral B Shah, and Alan Edelman. Julia: A fast dynamic language for technical computing. *arXiv:1209.5145*, 2012.

Carlos Diuk, Andre Cohen, and Michael L Littman. An object-oriented representation for efficient reinforcement learning. In *International Conference on Machine Learning (ICML)*, pages 240–247. ACM, 2008.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. *arXiv:1604.06778*, 2016.

Alborz Geramifard, Christoph Dann, Robert H Klein, William Dabney, and Jonathan P How. RLPy: A value-function-based reinforcement learning framework for education and research. *Journal of Machine Learning Research*, 16:1573–1578, 2015.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134, 1998.

Mykel J. Kochenderfer. *Decision Making Under Uncertainty: Theory and Application*. MIT Press, 2015.

Hanna Kurniawati, David Hsu, Wee Sun Lee, and Haoyu Bai. Approximate POMDP Planning Toolkit. `http://bigbird.comp.nus.edu.sg/pmwiki/farm/appl/`.

Michael L. Littman, Anthony R. Cassandra, and Leslie Pack Kaelbling. Learning policies for partially observable environments: scaling up. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 362–371, Tahoe City, California, 1995.

David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2164–2172, 2010.

Trey Smith. ZMDP software for POMDP planning. `https://github.com/trey0/zmdp`.

Travis B. Wolf and Mykel J. Kochenderfer. Aircraft collision avoidance using Monte Carlo real-time belief space search. *Journal of Intelligent and Robotic Systems*, 64(2):277–298, 2011.

Steve Young, Milica Gasic, Blaise Thomson, and Jason Williams. POMDP-based statistical spoken dialogue systems: a review. *Proceedings of the IEEE*, PP(99):1–20, 2013.

Journal of Machine Learning Research 18 (2017) 1-32 Submitted 8/10; Revised 12/14; Published 4/17

Generalized Pólya Urn for Time-Varying Pitman-Yor Processes

François Caron* CARON@STATS.OX.AC.UK
Department of Statistics
University of Oxford
Oxford, UK

Willie Neiswanger* WILLIE@CS.CMU.EDU
Machine Learning Department
Carnegie Mellon University
Pittsburgh, USA

Frank Wood FWOOD@ROBOTS.OX.AC.UK
Department of Engineering Science
University of Oxford
Oxford, UK

Arnaud Doucet DOUCET@STATS.OX.AC.UK
Department of Statistics
University of Oxford
Oxford, UK

Manuel Davy MANUEL.DAVY@VEKIA.FR
VEKIA
165 Avenue de Bretagne
59044 Lille, France

*Joint first authorship.

Editor: David Dunson

Abstract

This article introduces a class of first-order stationary time-varying Pitman-Yor processes. Subsuming our construction of time-varying Dirichlet processes presented in (Caron et al., 2007), these models can be used for time-dynamic density estimation and clustering. Our intuitive and simple construction relies on a generalized Pólya urn scheme. Significantly, this construction yields marginal distributions at each time point that can be explicitly characterized and easily controlled. Inference is performed using Markov chain Monte Carlo and sequential Monte Carlo methods. We demonstrate our models and algorithms on epidemiological and video tracking data.

Keywords: Bayesian nonparametrics, clustering, mixture models, sequential Monte Carlo, particle Markov chain Monte Carlo, dynamic models

©2017 François Caron, Willie Neiswanger, Frank Wood, Arnaud Doucet and Manuel Davy.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/10-231.html.

1. Introduction

This paper introduces a class of dependent Pitman-Yor processes[1] that can be used for time-varying density estimation and classification in a variety of settings. The construction of this class of dependent Pitman-Yor processes is in terms of a generalised Pólya urn scheme where dependencies between distributions that evolve over time, for instance, are induced by simple urn-like operations on counts and the parameters to which they are associated.

Our Pólya urn for time-varying Pitman-Yor processes is expressive per dependent slice, as each is represented by a Pitman-Yor process (infinite) mixture distribution of which the component densities may as usual take any form. The dependency-inducing mechanism is also flexible and easy to control, a claim supported by an applied literature (see Gasthaus et al. (2008); Ji and West (2009); Ozkan et al. (2009); Bartlett et al. (2010); Neiswanger et al. (2014); Jaoua et al. (2014) among others) that has grown around the subsumed version of this work (Caron et al., 2007). The generalised Pólya urn dependent Dirichlet process can be recovered by a specific settings of the parameters in this generalized model.

Most of the emphasis of this paper is on defining the model and describing its statistical characteristics. While additional model complexity does not automatically beget useful expressivity, in the case of this model and others like it (see Lin et al. (2010) for example), we assert that it does. For example, in applications like fully unsupervised visual object detection and tracking there is an increasing need for top-down models that introduce coherence through bias towards physical plausibility. While dependent density estimation techniques abound in the literature, there are few that possess the right combination of interpretability and flexibility to fill the role of top-down priors for such complex applications. We illustrate experimentally that this model succeeds at filling this role.

The remainder of the paper is organized as follows: In Section 2 we review Pitman Yor processes and stationarity. In Section 3, we present our main contribution, first-order stationary Pitman-Yor mixture models. Section 4 describes sequential Monte Carlo (SMC) and Markov chain Monte Carlo (MCMC) algorithms for inference. We demonstrate the models and algorithms in Section 5 on various applications. Finally we discuss some potential extensions of this class of models in Section 6 and Appendix A.

2. Background

Pitman-Yor processes include a wide class of distributions on random measures such as the popular Dirichlet process (Ferguson, 1973) and the finite symmetric Dirichlet-multinomial prior (Green and Richardson, 2001); see for example (Pitman and Yor, 1997). In particular, Dirichlet process mixtures can be interpreted as a generalization of finite mixture models to infinite mixture models and have become very popular over the past few years in statistics and related areas to perform clustering and density estimation (Escobar and West, 1995; Müller and Quintana, 2004; Teh and Jordan, 2010). More general Pitman-Yor processes enjoy greater flexibility and have been shown to provide a better fit to text or image data due to their ability to capture power-law properties of such data (Teh, 2006).

However, there are many situations where we cannot assume that the distribution of the observations is fixed and instead this latter evolves over time. For example, in a clustering

1. A preliminary version of this work has been presented as a conference paper (Caron et al., 2007)

application, the number of clusters and the locations of these clusters may vary over time. This situation also occurs in spatial statistics where the spatial distribution of localized events such as diseases or earthquakes changes over time (Hall et al., 2006). To address such problems, this article introduces a novel class of time-varying first-order stationary Pitman-Yor process mixtures; that is processes which have marginals following the same Pitman-Yor process mixture.

We first briefly recall here standard results about Pitman-Yor process mixtures. Let $t = 1, 2, ...$ denote a discrete-time index. Note that this index is not needed in this section but will become essential in what is to come. For any generic sequence $\{x_m\}$, we define $x_{k:l} = (x_k, x_{k+1}, \ldots, x_l)$. For ease of presentation, we assume that we receive a fixed number n of observations at each time t denoted $\mathbf{z}_t = \mathbf{z}_{1:n,t}$ which are independent and identically distributed samples from

$$F_t(\cdot) = \int_{\mathcal{Y}} f(\cdot|\mathbf{y}) d\mathbb{G}_t(\mathbf{y}) \tag{1}$$

where $f(\cdot|\mathbf{y})$ is the mixed probability density function and $\mathbb{G}_t$ is the mixing distribution distributed according to a Pitman-Yor process

$$\mathbb{G}_t \sim \mathcal{PY}(\alpha, \theta, \mathbb{H}). \tag{2}$$

Here $\mathbb{H}$ is a base probability measure and the real parameters α and θ satisfy either

$$0 \leq \alpha < 1 \text{ and } \theta > -\alpha \tag{3}$$
$$\text{or } \alpha < 0 \text{ and } \theta = -m\alpha \text{ for } m \in \mathbb{N}. \tag{4}$$

The case $\alpha = 0$ and $\theta > 0$ corresponds to the Dirichlet process denoted $DP(\theta, \mathbb{H})$. The random measure $\mathbb{G}_t$ satisfies the following *stick-breaking* representation (Sethuraman, 1994; Pitman, 1996)

$$\mathbb{G}_t = \sum_{j=1}^{\infty} V_{j,t} \delta_{U_{j,t}} \tag{5}$$

with $V_{j,t} = \beta_{j,t} \prod_{i=1}^{j-1}(1 - \beta_{i,t})$, $\beta_{j,t} \overset{\text{ind}}{\sim} \mathcal{B}(1 - \alpha, \theta + j\alpha)$, $U_{j,t} \overset{\text{iid}}{\sim} \mathbb{H}$ where $\mathcal{B}$ denotes the Beta distribution. From (1), we have equivalently, for $k = 1, \ldots, n$, the following hierarchical model

$$\mathbf{y}_{k,t}|\mathbb{G}_t \overset{\text{iid}}{\sim} \mathbb{G}_t, \tag{6}$$
$$\mathbf{z}_{k,t}|\mathbf{y}_{k,t} \overset{\text{iid}}{\sim} f(\cdot|\mathbf{y}_{k,t}). \tag{7}$$

We can also reformulate the Pitman-Yor process mixture by integrating out the mixing measure $\mathbb{G}_t$ and introducing allocation variables $\mathbf{c}_t = c_{1:n,t}$. For any $j \in \mathcal{J}(\mathbf{c}_t)$, where $\mathcal{J}(\mathbf{c}_t)$ is the set of unique values in $\mathbf{c}_t$, we have

$$U_{j,t} \overset{\text{iid}}{\sim} \mathbb{H}, \tag{8}$$
$$\mathbf{z}_{k,t}|U_{c_{k,t},t} \sim f(\cdot|U_{c_{k,t},t}).$$

For convenience, we label here the clusters by their order of appearance. We set $c_{1,1} = 1$, $K_1 = 1$ and $\mathbf{m}_1^1 = m_{1:K_1,1}^1$ a vector of size K_1. Then, at time $t = 1$, for $k = 2, \ldots, n$ the

following Pólya urn model describes the predictive distribution of a new allocation variable (Blackwell and MacQueen, 1973; Pitman, 1996)

w.p. $\frac{m_{i,1}^1 - \alpha}{k - 1 + \theta}$, $i \in \{1, \ldots, K_1\}$ set $c_{k,1}^1 = i$, $m_{i,1}^1 = m_{i,1}^1 + 1$,
w.p. $\frac{K_1\alpha + \theta}{k - 1 + \theta}$, set $K_1 = K_1 + 1$, $c_{k,1}^1 = K_1$, $m_{K_1,1}^1 = 1$,

where the abbreviation 'w.p.' stands for 'with probability'.

The sequence associated with $c_{1,t}, \ldots, c_{n,t}$ is exchangeable and induces a random partition of n, that is an unordered collection of $k \leq n$ positive integers with sum n or, equivalently, a random allocation of n unlabeled points into some random number of unlabeled clusters (materialized by a color for example); each cluster containing at least one point. A common way to represent a partition of n is by the number of terms of various sizes; that is the vector of counts $(a_1, \ldots, a_n)$ where $\sum_{j=1}^n a_j = k$ and $\sum_{j=1}^n j a_j = n$. Here a_1 is the number of terms of size 1, a_2 is the number of terms of size 2, etc. Following (Antoniak, 1974), we say that $\mathbf{c}_t \in C(a_{1:n})$ if there are a_1 distinct values of $\mathbf{c}_t$ that occur only once, a_2 that occur twice, etc. It can be shown that $\Pr(\mathbf{c}_t \in C(a_{1:n})) = P_n(a_{1:n})$ is given by the two-parameter Ewens sampling formula (Pitman, 1995)

$$P_n(a_1, \ldots, a_n) = \frac{n! \prod_{i=1}^{\sum_j a_j} (\theta + (i-1)\alpha)}{\prod_{i=1}^n (\theta + i - 1)} \prod_{i=1}^n \left(\frac{\prod_{j=1}^{i-1}(j - \alpha)}{i!} \right)^{a_i} \frac{1}{a_i!}. \tag{9}$$

In this paper, we introduce a class of statistical models with dependencies between the distributions $\{F_t\}$ and mixing distributions $\{\mathbb{G}_t\}$ while preserving (1) and (2) at any time t. This allows us to explicitly characterize the marginal model at each time step (more generally, at each value of the covariate) and have fine control over prior parameterization of the marginal model. Methods that preserve stationarity in this way have found use in other Bayesian nonparametric models (Teh et al., 2011; Griffin and Steel, 2011), and allow us to achieve good empirical performance in epidemiological and video applications in Section 5. We briefly review constructions in the literature for dependent processes below.

2.1 Literature review

Several authors have considered previously the problem of defining dependent nonparametric models, and in particular dependent Dirichlet processes for time series and spatial models, see e.g. Foti and Williamson (2013) for a recent review.

In an early contribution, Cifarelli and Regazzini (1978) introduced dependencies between distributions by defining a parametric model on the base distribution $\mathbb{H}_s$ dependent on a covariate s and $\mathbb{G}_s \sim \text{DP}(\theta, \mathbb{H}_s)$. This approach is different from ours as we follow here the setting introduced by MacEachern et al. (1999) and introduce dependencies directly on two successive mixing distributions while $\mathbb{H}$ is fixed.

The great majority of recent papers use the stick-breaking representation (5) to introduce dependencies. Under this representation, a realization of a Dirichlet process is represented by two (infinite dimensional) vectors of weights $V_{1:\infty,s}$ and cluster locations $U_{1:\infty,s}$. Dependency with respect to a covariate s is introduced on $V_{1:\infty,s}$ in (Griffin and Steel, 2006; Rodriguez and Dunson, 2011; Arbel et al., 2014), on $U_{1:\infty,s}$ in (MacEachern, 2000; Iorio et al., 2004; Gelfand et al., 2005; Caron et al., 2008) and on both cluster locations and weights in (Griffin and Steel, 2009).

An alternative approach consists of considering the mixing distribution to be a convex combination of independent random probability measures sampled from a Dirichlet process. The dependency is then introduced through some weighting coefficients; e.g. (Dunson et al., 2006; Dunson and Park, 2007; Müller et al., 2004). More recently, various classes of dependent Dirichlet processes were proposed by Griffin (2011), Rao and Teh (2009), Huggins and Wood (2014) and Lin et al. (2010) which rely on the representation of the DP as a normalized random measure.

Although these previous approaches have merits, we believe that it is possible to build more intuitive models in the time domain based on Pólya urn-type schemes. In (Zhu et al., 2005; Ahmed and Xing, 2008; Blei and Frazier, 2011), time-varying Pólya urn models were proposed but these models do not marginally preserve a Dirichlet process. The only model we know of which satisfies this property is presented in (Walker and Muliere, 2003). The authors define a joint distribution $p(\mathbb{G}_1, \mathbb{G}_2)$ such that $\mathbb{G}_1$ and $\mathbb{G}_2$ are marginally $\mathrm{DP}(\theta, \mathbb{H})$ by introducing m artificial auxiliary variables $\mathbf{w}_i \overset{\text{iid}}{\sim} \mathbb{G}_1$ and then $\mathbb{G}_2|\mathbf{w}_{1:m} \sim \mathrm{DP}(\theta+m, \frac{\theta\mathbb{H}+\sum_{i=1}^m \delta_{\mathbf{w}_i}}{\theta+m})$. An extension to time series is discussed in (Srebro and Roweis, 2005). An important drawback of this approach is that it requires introducing a very large number m of auxiliary variables to model strongly dependent distributions. When inference is performed, these auxiliary variables need to be inferred from the data and the resulting algorithm can be computationally intensive.

2.2 Contributions and Organization

The models developed here are based on a Pólya urn representation of the Pitman-Yor process but do not require introducing a large number of auxiliary variables to model strongly dependent distributions.

To obtain a first-order stationary Pitman-Yor process mixture using such an approach, we need to ensure that any time t

(A) the sequence $\mathbf{c}_t$ induces a random partition distributed according to the two-parameter Ewens sampling formula (9),

(B) for $j \in \mathbf{c}_t$, the $U_{j,t}$s are identically and independently distributed from $\mathbb{H}$.

The main contribution of this paper consists of defining models satisfying (**A**) using a generalized Pólya urn prediction rule based on the consistence properties under specific deletion procedures of the Ewens sampling formula (Kingman, 1978; Gnedin and Pitman, 2005). Ensuring (**B**) can be performed using standard methods from the time series literature; e.g. (Joe, 1997; Pitt and Walker, 2005).

Our models allow us to modify both the cluster locations and their weights. Furthermore, they rely on simple and intuitive birth and death procedures. By using a Pólya urn approach, the models are defined on the space of partitions; i.e. the labelling of the class to which each data belongs is irrelevant. From a computational point of view, it is usually easier to design efficient MCMC and SMC algorithms for inference based on this finite-dimensional representation than the infinite-dimensional stick-breaking representation, where slice sampling (Walker, 2007; Kalli et al., 2011) or retrospective sampling (Papaspiliopoulos and Roberts, 2008) techniques can be used.

3. Stationary Pitman-Yor Process Mixtures

We present here some models ensuring property (**A**) is satisfied. We then briefly discuss in Section 3.2 how to ensure (**B**).

3.1 Stationary Pitman-Yor Processes

The main idea behind our models consists at each time step t of

- deleting randomly a subset of the allocations variables sampled from time 1 to $t-1$ which had survived the previous $t-1$ deletion steps,

- sampling n new allocation variables corresponding to the n observations $\mathbf{z}_t$.

For any $t \geq 2$, we have generated the allocation variables $\mathbf{c}_{1:t-1}$ corresponding to $\mathbf{z}_{1:t-1}$ from time 1 to $t-1$. We denote by $\mathbf{c}^{t-1}_{1:t-1}$ the subset of $\mathbf{c}_{1:t-1}$ corresponding to variables having survived the deletion steps from time 1 to $t-1$, and we denote by $\mathbf{c}^{t}_{1:t-1}$ the subset corresponding to those having survived from time 1 to t. Let K_{t-1} be the number of clusters created from time 1 to $t-1$. We denote by $\mathbf{m}^{t-1}_{t-1}$ the vector of size K_{t-1} containing the size of the clusters associated to $\mathbf{c}^{t-1}_{1:t-1}$, and we denote by $\mathbf{m}^{t}_{t-1}$ the vector containing the size of clusters associated to $\mathbf{c}^{t}_{1:t-1}$. Hence, these vectors have zero entries corresponding to 'dead' clusters. The introduction of $\mathbf{m}^{t-1}_{t-1}$ and $\mathbf{m}^{t}_{t-1}$ simplifies the presentation of the procedure but note that, from a practical point of view, there is obviously no need to store these vectors of increasing dimension. It is only necessary to store the size of the non-empty clusters and their associated labels.

At time 1, we just generate $\mathbf{c}_1$ according to a standard Pólya urn described in the introduction. At time $t \geq 2$ we have $\mathbf{c}^{t-1}_{1:t-1} = \left(\mathbf{c}^{t-1}_{1:t-2}, \mathbf{c}_{t-1}\right)$ and we sample $\mathbf{c}^{t}_{1:t} = \left(\mathbf{c}^{t}_{1:t-1}, \mathbf{c}_{t}\right)$ as follows. We first obtain $\mathbf{c}^{t}_{1:t-1}$ by deleting a random number of allocation variables from $\mathbf{c}^{t-1}_{1:t-1}$ according to one of the following rules.

- **Uniform deletion**: delete each allocation variable in $\mathbf{c}^{t-1}_{1:t-1}$ with probability $1-\rho$ where $0 \leq \rho \leq 1$. This is statistically equivalent to sampling a number r from a binomial distribution $\mathcal{B}\mathrm{in}(\sum_k m^{t-1}_{k,t-1}, 1-\rho)$ and then removing r items uniformly from $\mathbf{c}^{t-1}_{1:t-1}$ to obtain $\mathbf{c}^{t}_{1:t-1}$. For $\rho = 0$, the partitions $\mathbf{c}_t$ and $\mathbf{c}_{t+1}$ are independent whereas for $\rho = 1$ we have a static Pitman-Yor process.

- **Deterministic deletion**: delete the allocation variables $\mathbf{c}_{t-r}$ from $\mathbf{c}^{t-1}_{1:t-1}$, where $r \in \mathbb{N}$ and $r < t$.

- **Cluster deletion** (for α, θ verifying condition (3) with $\theta \geq 0$): compute the following discrete probability distribution over the set of non-empty clusters

$$\pi_{k,t} = \frac{(\sum_\ell m^{t-1}_{\ell,t-1} - m^{t-1}_{k,t-1})\gamma + m^{t-1}_{k,t-1}(1-\gamma)}{(\sum_\ell m^{t-1}_{\ell,t-1})(1-\gamma+(K_{t-1}-1)\gamma)}$$

where $\sum_k \pi_{k,t} = 1$, $\gamma = \frac{\alpha}{\alpha+\theta}$ then sample an index from this distribution and delete the corresponding cluster to obtain $\mathbf{c}^{t}_{1:t-1}$. The cluster deletion allows us to model large potential 'jumps' in the distributions of the observations. Dependent on the value of γ, we

delete clusters with a probability independent of their size, proportional to their size or proportional to the size of the partition minus their size (Gnedin and Pitman, 2005). The introduction of the Pitman-Yor process gives us this extra modeling flexibility here. In particular, the following three cases are of special interest:

Size-biased deletion For $\alpha = 0$ and $\theta > 0$, and hence $\gamma = 0$, each cluster of size r is selected with probability proportional to r. This result is known as Kingman's characterization of the Ewens family of $(0, \theta)$ partition structures (Kingman, 1978).

Unbiased (uniform) deletion For $\gamma = \frac{\alpha}{\alpha+\theta} = \frac{1}{2}$, given that there are l clusters, each cluster is chosen with probability $\frac{1}{l}$ i.e., each cluster is deleted independently of its size. For $0 \leq \alpha \leq 1$, the (α, α) partition structures are the only partition structures invariant under uniform deletion.

Cosize-biased deletion For $\gamma = 1$ (hence $\theta = 0$), each cluster of size r is selected with probability proportional to the size $n - r$ of the remaining partition.

It is also possible to consider any mixture and composition of these deletion procedures. For example, we can pick w.p. ξ the uniform deletion strategy and w.p. $1 - \xi$ the cluster deletion strategy or perform one uniform deletion followed by one cluster deletion or a deterministic deletion etc. Finally, after these deletion steps, we sample the allocation variables $\mathbf{c}_t$ according to a standard Pólya urn scheme based on the surviving allocation variables $\mathbf{c}^t_{1:t-1}$.

To summarize, the generalized Pólya urn scheme proceeds as in Algorithm 1, where $\mathcal{I}\left(\mathbf{m}^t_t\right)$ and $\left|\mathcal{I}\left(\mathbf{m}^t_t\right)\right|$ denote respectively the indices corresponding to the non-zero entries of $\mathbf{m}^t_t$ and the number of non-zero entries.

Algorithm 1 Generalized Pólya Urn

At time $t = 1$

- Set $c^1_{1,1} = 1$, $m^1_{1,1} = 1$ and $K_1 = 1$.
- For $k = 2, ..., n$

w.p. $\frac{m^1_{i,1}-\alpha}{k-1+\theta}$, $i \in \{1, \ldots, K_1\}$ set $c^1_{k,1} = i$, $m^1_{i,1} = m^1_{i,1} + 1$,
w.p. $\frac{K_1\alpha+\theta}{k-1+\theta}$, set $K_1 = K_1 + 1$, $c^1_{k,1} = K_1$, $m^1_{K_1,1} = 1$.

At time $t \geq 2$

- Kill a subset of $\mathbf{c}^{t-1}_{1:t-1}$ using a mixture/composition of uniform, size-biased and deterministic deletions to obtain $\mathbf{c}^t_{1:t-1}$ (hence $\mathbf{m}^t_{t-1}$) and set $\mathbf{m}^t_t = \mathbf{m}^t_{t-1}$, $K_t = K_{t-1}$.
- For $k = 1, ..., n$

w.p. $\frac{m^t_{i,t}-\alpha}{\sum_i m^t_{i,t}+\theta}$, $i \in \mathcal{I}\left(\mathbf{m}^t_t\right)$ set $c^t_{k,t} = i$, $m^t_{i,t} = m^t_{i,t} + 1$,
w.p. $\frac{|\mathcal{I}(\mathbf{m}^t_t)|\alpha+\theta}{\sum_i m^t_{i,t}+\theta}$, set $K_t = K_t + 1$, $c^t_{k,t} = K_t$, $m^t_{K_t,t} = 1$.

The proof that $\mathbf{c}_t$ satisfies (**A**) is a direct consequence of the remarkable consistence properties under deletion of the Ewens sampling formula which have been first established in (Kingman, 1978) for the one-parameter case and then extended to the two-parameter case in (Gnedin and Pitman, 2005).

Proposition. At any time $t \geq 1$, $\mathbf{c}_t$ induces a random partition distributed according to the Ewens sampling formula.

Proof. We prove by induction a stronger result; that is $\mathbf{c}^t_{1:t}$ induces a random partition following (9). At time 1, this is trivially true as $\mathbf{c}^1_1 = \mathbf{c}_1$ is generated according to a standard Pólya urn. Assume it is true at time $t - 1$. For the deterministic and uniform deletion, exchangeability of the partition ensures that $\mathbf{c}^t_{1:t-1}$ also induces a random partition following (9). For the cluster deletion procedure, consistency follows from the results of (Kingman, 1978, pp. 3 and 5) and (Gnedin and Pitman, 2005, p. 4) on regenerative partitions. Finally, as $\mathbf{c}_t$ is sampled according to a standard Pólya urn scheme based on the surviving allocation variables $\mathbf{c}^t_{1:t-1}$ then $\mathbf{c}^t_{1:t}$ indeed induces by construction a random partition following (9). Thanks to exchangeability, it implies that $\mathbf{c}_t$ also induces a random partition distributed according to the Ewens sampling formula.■

So far, so as to simplify presentation, we have considered that the number of allocation variables at each time t is fixed to a value n corresponding to the number of observations received at each time instant. The value of n impacts on the number of alive allocation variables and the correlations between successive vectors $\mathbf{m}^t_{1:t}$. More precisely, the statistical model is not consistent in the Kolmogorov sense. For example, let $\pi_1(c_{1,1}, c_{1,2})$ and $\pi_2(c_{1,1}, c_{2,1}, c_{1,2}, c_{2,2})$ be two models defined for $n = 1$ and $n = 2$, then

$$\left(\sum_{c_{2,1}}\sum_{c_{2,2}} \pi_2(c_{1,1}, c_{2,1}, c_{1,2}, c_{2,2})\right) \neq \pi_1(c_{1,1}, c_{1,2}).$$

This lack of consistency is shared by other models based on the Pólya urn construction (Zhu et al., 2005; Ahmed and Xing, 2008; Blei and Frazier, 2011). Blei and Frazier (2011) provide a detailed discussion on this issue and describe cases where this property is relevant or not.

It is nonetheless possible to define a slightly modified version of our model that is consistent under marginalisation, at the expense of an additional set of latent variables. This is described in Appendix C.

3.2 Stationary Models for Cluster Locations

To ensure we obtain a first-order stationary Pitman-Yor process mixture model, we also need to satisfy (**B**). This can be easily achieved if for $k \in \mathcal{I}(\mathbf{m}^t_t)$

$$U_{k,t} \sim \begin{cases} p\left(\cdot|U_{k,t-1}\right) & \text{if } k \in \mathcal{I}(\mathbf{m}^t_{t-1}) \\ \mathbb{H} & \text{otherwise} \end{cases}$$

where $\mathbb{H}$ is the invariant distribution of the Markov transition kernel $p\left(\cdot|\cdot\right)$. In the time series literature, many approaches are available to build such transition kernels based on copulas (Joe, 1997) or Gibbs sampling techniques (Pitt and Walker, 2005).

Combining the stationary Pitman-Yor and cluster locations models, we can summarize the full model by the following Bayesian network in Figure 1. It can also be summarized using a Chinese restaurant metaphor (see Figure 2).

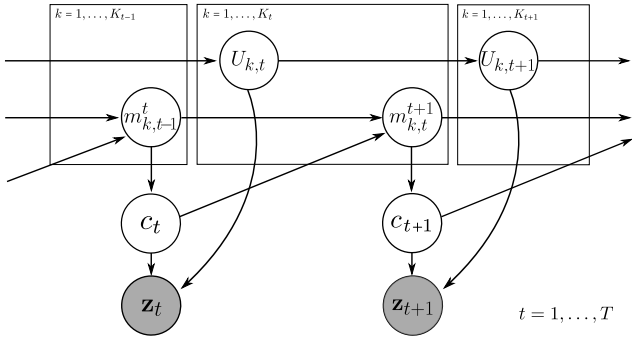


Figure 1: A representation of the time-varying Pitman-Yor process mixture as a directed graphical model, representing conditional independencies between variables. All assignment variables and observations at time t are denoted c_t and $\mathbf{z}_t$, respectively.

3.3 Properties of the Models

Under the uniform deletion model, the number $A_t = \sum_i m^t_{i,t-1}$ of alive allocation variables at time t can be written as

$$A_t = \sum_{j=1}^{t-1} \sum_{k=1}^{n} X_{j,k}$$

where $X_{j,i}$ are independently distributed from a Bernoulli of parameter ρ^j. A_t is therefore distributed from a Poisson binomial (also called Pólya Frequency) distribution (Pitman, 1997). The asymptotic mean and variance of the distribution of A_t are respectively $\frac{n\rho}{1-\rho}$ and $\frac{n\rho}{1-\rho^2}$, and the distribution $\Pr(A_t = k) \propto a_{k,t-1}$ where $a_{k,t}$, $k = 0, \ldots, nt$ satisfies the algebraic identity

$$\prod_{i=1}^{t} \left(x + \frac{1-\rho^i}{\rho^i} \right)^n = \sum_{k=0}^{nt} a_{k,t} x^k. \tag{10}$$

Its stationary distribution is obtained by taking the limit as $t \to \infty$.

Clearly the sequence $\{\mathbf{c}_t\}$ is not Markovian but $\{\mathbf{c}^t_{1:t}\}$ and $\{\mathbf{c}^t_{1:t-1}\}$ and the associated vectors $\{\mathbf{m}^t_{1:t}\}$ and $\{\mathbf{m}^t_{1:t-1}\}$ are for the uniform and cluster deletions. The transition probabilities for these processes are quite complex. However it can be shown easily for example that, for the uniform deletion model, we have for $k \in \mathcal{I}(\mathbf{m}^t_{t-1})$

$$\begin{aligned}
\mathbb{E}\left[m^{t+1}_{k,t} | \mathbf{m}^t_{t-1} \right] &= \mathbb{E}\left[\mathbb{E}\left[m^{t+1}_{k,t} \middle| m^t_{k,t} \right] | \mathbf{m}^t_{t-1} \right] \\
&= \rho \mathbb{E}\left[m^t_{k,t} | \mathbf{m}^t_{t-1} \right] \\
&= \rho \left(m^t_{k,t-1} + n \frac{m^t_{k,t-1} - \left| \mathcal{I}\left(\mathbf{m}^t_{t-1}\right) \right| \alpha}{\theta + \sum_k m^t_{k,t-1}} \right)
\end{aligned}$$

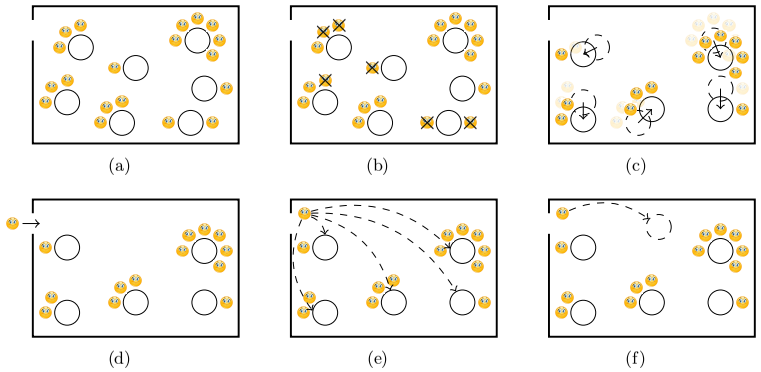


Figure 2: Illustration of the uniform deletion time-varying Pitman-Yor process mixture. Consider a restaurant with a countably infinite number of tables. (a) At time t, there are a certain number of customers in the restaurant, shared between several tables. Each customer remains seated at her/his table (w.p. ρ), or leaves the restaurant (w.p. $(1-\rho)$). (b) Once this choice has been made by each customer, empty tables are removed, and a certain number of customers remain in the restaurant. (c) Each table that is still occupied has its location evolved according to the transition kernel $p(\cdot|\cdot)$. (d) A new customer enters the restaurant and either (e) sits at a table with a probability proportional to the number of people at this table or (f) sits alone at a new table whose location has distribution $\mathbb{H}$. When $n-1$ other new customers enter the restaurant, repeat operations (d)-(f).

and

$$\mathbb{E}[\sum_{k \notin \mathcal{I}(\mathbf{m}^t_{t-1})} m^{t+1}_{k,t} | \mathbf{m}^t_{t-1}] = \frac{\rho n \left[\left| \mathcal{I}\left(\mathbf{m}^t_{t-1}\right) \right| \alpha + \theta \right]}{\theta + \sum_k m^t_{k,t-1}}.$$

For any deletion model, it can be additionally shown that we have, conditional on $\mathbf{m}^t_{t-1}$ (Pitman, 1996)

$$\mathbb{G}_t = \sum_{j \in \mathcal{I}(\mathbf{m}^t_{t-1})} P_j U_{j,t} + R_t \mathbb{G}^\star_t$$

where $\mathbb{G}^\star_t \sim \mathcal{PY}(\alpha, \theta + \alpha \left| \mathcal{I}\left(\mathbf{m}^t_{t-1}\right) \right|, \mathbb{H})$ and

$$\left(\{P_j | j \in \mathcal{I}(\mathbf{m}^t_{t-1})\}, R_t \right) | \mathbf{m}^t_{t-1} \sim \mathcal{D}(\{m^t_{j,t-1} - \alpha | j \in \mathcal{I}(\mathbf{m}^t_{t-1})\}, \left| \mathcal{I}(\mathbf{m}^t_{t-1}) \right| \alpha + \theta)$$

where $\mathcal{D}$ is the standard Dirichlet distribution. Moreover, $\mathbb{G}^\star_t$ and $\left(\{P_j | j \in \mathcal{I}(\mathbf{m}^t_{t-1})\}, R_t \right)$ are statistically independent. In particular, in the Dirichlet process case, we have

$$\mathbb{G}_t | \mathbf{c}^t_{1:t-1}, U_{j,t} \sim DP\left(\theta + \left| \mathcal{I}\left(\mathbf{m}^t_{t-1}\right) \right|, \frac{1}{\theta + \left| \mathcal{I}\left(\mathbf{m}^t_{t-1}\right) \right|} (\theta \mathbb{H} + \sum_k m^{t-1}_{k,t} \delta_{U_{k,t}}) \right).$$

We display a Monte Carlo estimate of $\mathrm{corr}\big(\int \mathbf{y}\mathbb{G}_t(d\mathbf{y}), \int \mathbf{y}\mathbb{G}_{t+\tau}(d\mathbf{y})\big)$ when $\mathbb{H}(\mathbf{y})$ is a standard normal distribution for different values of ρ, ξ and r resp. for the uniform, size-biased and deterministic deletions, and θ (with $\alpha = 0$) in Fig. 3. The correlations decrease faster as ρ, r and ξ decrease as expected.

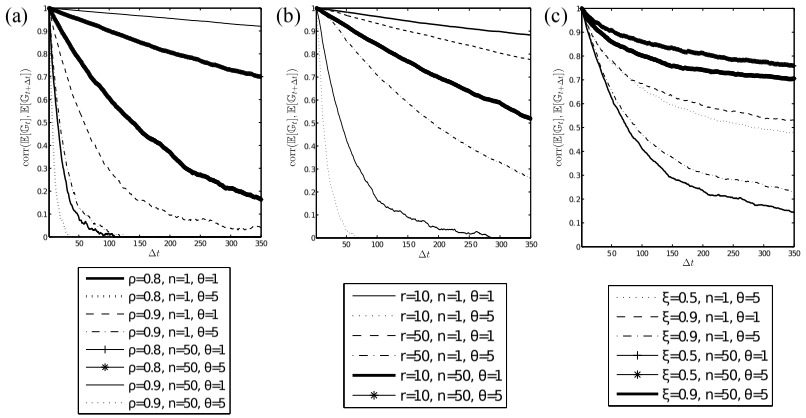


Figure 3: Plots of $\mathrm{corr}\big(\int \mathbf{y}\mathbb{G}_t(d\mathbf{y}), \int \mathbf{y}\mathbb{G}_{t+\Delta t}(d\mathbf{y})\big)$ approximated by Monte Carlo simulations as a function of the time difference Δt for (a) uniform (b) deterministic and (c) cluster deletions, for different values of θ and (a) ρ, (b) r and (c) ξ.

4. Bayesian Inference in Time-Varying PYPM

Bayesian inference is based on the posterior distribution of the cluster assignment variables $\mathbf{c}_t$, the vectors $\mathbf{m}_{t-1}^t$ and $U_{1:K_t}$ given by $p(\mathbf{c}_{1:t}, \mathbf{m}_{1:t-1}^{1:t}, U_{1:K_t}|\mathbf{z}_{1:t})$ at time t. We describe sequential Monte Carlo methods to fit our models.

To sample approximately from the sequence of distributions $p(\mathbf{c}_{1:t}, \mathbf{m}_{1:t-1}^{1:t}, U_{1:K_t}|\mathbf{z}_{1:t})$ as t increases, we propose an SMC method also known as particle filter. In this approach, the posterior distribution is approximated by a large collection of random samples—termed particles—which are propagated through time using Sequential Importance Sampling with resampling steps; see (Doucet et al., 2001) for a review of the literature. MacEachern et al. (1999) developed a sequential importance sampling method for Beta-Binomial Dirichlet process mixtures and Fearnhead (2004) proposed a SMC method for general conjugate Dirichlet process mixtures. In these papers, even in cases where the mixed distribution $\mathbb{G}$ can be integrated out analytically, the fact that $\mathbb{G}$ is a static parameter leads to an accumulation of the Monte Carlo errors over time (Kantas et al., 2009; Poyiadjis et al., 2011). This prevents using such techniques for large datasets with long time sequences. We deal here with time-varying models whose forgetting properties limit drastically this accumulation of errors.

We present in Algorithm 2 a general SMC algorithm. It relies on importance distributions denoted generically by $q(\cdot)$ and is initialized with weights $w_0^{(i)} = N^{-1}$ and cluster sizes $m_0^{0\ (i)} = 0$ for $i = 1, ..., N$. The simplest case of this algorithm consists of selecting the prior as an importance distribution. Note that we use $\overline{A}$ to denote the complementary of the set $A \subset \mathbb{N}$ in $\mathbb{N}$.

Algorithm 2 Sequential Monte Carlo for Time-Varying Pitman-Yor mixture processes

At time $t \geq 1$

- For each particle $i = 1, .., N$
  - Sample $\widetilde{\mathbf{m}}_{t-1}^{t\ (i)} | \mathbf{m}_{t-1}^{t-1\ (i)} \sim \Pr(\mathbf{m}_{t-1}^t | \mathbf{m}_{t-1}^{t-1\ (i)})$
  - Sample $\widetilde{\mathbf{c}}_t^{(i)} \sim q(\mathbf{c}_t | \widetilde{\mathbf{m}}_{t-1}^{t\ (i)}, U^{(i)}_{\mathcal{I}(\widetilde{\mathbf{m}}_t^{(i)}),t-1}, z_t)$
  - For $k \in \mathcal{J}(\widetilde{\mathbf{c}}_t^{(i)}) \cap \overline{\mathcal{I}(\widetilde{\mathbf{m}}_{t-1}^{t\ (i)})}$, sample $\widetilde{U}_{k,t}^{(i)} \sim q(U_{k,t} | \{\mathbf{z}_{j,t} | \widetilde{c}_{j,t}^{(i)} = k\})$
  - For $k \in \mathcal{I}(\widetilde{\mathbf{m}}_{t-1}^{t\ (i)})$, sample $\widetilde{U}_{k,t}^{(i)} \sim \begin{cases} q(U_{k,t} | U_{k,t-1}^{(i)}, \{\mathbf{z}_{j,t} | \widetilde{c}_{j,t}^{(i)} = k\}) & \text{if } k \in \mathcal{J}(\widetilde{\mathbf{c}}_t^{(i)}) \\ p(U_{k,t} | U_{k,t-1}^{(i)}) & \text{otherwise} \end{cases}$
- For $i = 1, .., N$, update the weights

$$\widetilde{w}_t^{(i)} \propto w_{t-1}^{(i)} \frac{p(\mathbf{z}_t | \widetilde{U}_t^{(i)}, \widetilde{\mathbf{c}}_t^{(i)}) \Pr(\widetilde{\mathbf{c}}_t^{(i)} | \widetilde{\mathbf{m}}_{t-1}^{t\ (i)})}{q(\widetilde{\mathbf{c}}_t^{(i)} | \widetilde{\mathbf{m}}_{t-1}^{t\ (i)}, U^{(i)}_{\mathcal{I}(\widetilde{\mathbf{m}}_t^{(i)}),t-1}, \mathbf{z}_t)} \times \prod_{k \in \mathcal{I}(\widetilde{\mathbf{m}}_{t-1}^{t\ (i)})} \frac{p(\widetilde{U}_{k,t}^{(i)} | U_{k,t-1}^{(i)})}{q(\widetilde{U}_{k,t}^{(i)} | U_{k,t-1}^{(i)}, \{\mathbf{z}_{j,t} | \widetilde{c}_{j,t}^{(i)} = k\})} \times \prod_{k \in \mathcal{J}(\widetilde{\mathbf{c}}_t^{(i)}) \cap \overline{\mathcal{I}(\widetilde{\mathbf{m}}_{t-1}^{t\ (i)})}} \frac{\mathbb{H}(\widetilde{U}_{k,t}^{(i)})}{q(\widetilde{U}_{k,t}^{(i)} | \{\mathbf{z}_{j,t} | \widetilde{c}_{j,t}^{(i)} = k\})} \tag{11}$$

  with $\sum_{i=1}^N \widetilde{w}_t^{(i)} = 1$.
- Resampling. Compute $N_{\text{eff}} = \left[\sum \left(\widetilde{w}_t^{(i)}\right)^2\right]^{-1}$. If $N_{\text{eff}} \leq N_T$, multiply the particles with large weights and remove the particles with small weights, resulting in a new set of particles denoted $\cdot_t^{(i)}$ with weights $w_t^{(i)} = 1/N$. Otherwise, rename the particles and weights by removing the $\widetilde{\cdot}$.

In this algorithm, N_T is a threshold triggering the resampling step; typically we set $N_T = N/2$. The posterior $p(\mathbf{c}_{1:t}, \mathbf{m}_{1:t-1}^{1:t}, U_{1:K_t}|\mathbf{z}_{1:t})$ is approximated using the set of weighted samples $\left\{w_t^{(i)}, \left(\mathbf{c}_{1:t}^{(i)}, \mathbf{m}_{1:t-1}^{1:t\ (i)}, U_{1:K_t}^{(i)}\right)\right\}$. In cases where $\mathbb{H}$ is a conjugate prior for f, we can integrate out the cluster locations and sample only the allocation variables.

We also develop a particle MCMC (PMCMC) inference algorithm (Andrieu et al., 2010) for this model. PMCMC is a method that uses SMC as an intermediate sampling step to move efficiently through high dimensional state spaces. It is an alternative to other common forms of MCMC, such as single-site Gibbs sampling, which we have found suffer from worse empirical performance due to quasi-ergodicity (i.e. poor mixing as the Markov chain becomes stuck in posterior modes). We implement the iterated-conditional SMC form of PMCMC (Andrieu et al., 2009, 2010), which involves iterating through the SMC algorithm described above while conditioning on a sampled particle at each iteration. While

this algorithm does not operate in an online fashion, as SMC does, we show in Section 5 that it yields improved performance in tasks such as video object tracking (shown in Section 5).

An alternative approach consists of modelling the observations as follows (we discuss here the deterministic deletion model but the uniform deletion can be defined similarly). In this model, the predictive distribution of $\mathbf{z}_t$ conditional on $c_t = k$ only depends on the observations assigned to the same cluster in the time interval $\{t-r+1, ..., t-1\}$ and

$$\begin{aligned} p(\mathbf{z}_t|c_t = k, \mathbf{z}_{1:t-1}, c_{1:t-1}) &= p(\mathbf{z}_t|\{\mathbf{z}_j : c_j = k \text{ for } j = t-r+1, \ldots, t-1\}) \\ &= \frac{\int_{\mathcal{Y}} f(\mathbf{z}_t|\mathbf{y}) \prod_{\{j\in\{t-r+1,\ldots t-1\}:c_j=k\}} f(\mathbf{z}_j|\mathbf{y}) d\mathbb{H}(\mathbf{y})}{\int_{\mathcal{Y}} \prod_{\{j\in\{t-r+1,\ldots t-1\}:c_j=k\}} f(\mathbf{z}_j|\mathbf{y}) d\mathbb{H}(\mathbf{y})}. \end{aligned} \tag{12}$$

This distribution can be computed in closed-form in the conjugate case. We propose the simple deterministic Algorithm 3 to approximate the posterior in the context of this model. The posterior $p(\mathbf{c}_{1:t}|\mathbf{z}_{1:t})$ is approximated using the set of weighted samples $\left\{w_t^{(i)}, \mathbf{c}_{1:t}^{(i)}\right\}$. This algorithm uses a deterministic selection step. Alternatively, we could have used the stochastic resampling procedure of Fearnhead (2004).

Algorithm 3 $N-$best algorithm for Deterministic Deletion

At time $t \geq 1$

- Set $w_{0,t}^{(i)} = w_{n,t-1}^{(i)}$
- For $k = 1, \ldots, n$
  - For each particle $i = 1, .., N$
    - For $j \in \mathcal{J}(\{c_{1:k-1,t}^{(i)}, c_{1:t-1}^{(i)}\}) \cup \{0\} \cup \{c_{new}\}$, let $\widetilde{c}_{k,t}^{(i,j)} = j$ and compute the weight

$$\widetilde{w}_{k,t}^{(i,j)} \propto w_{k-1,t}^{(i)} p\left(\mathbf{z}_{k,t}|\, \mathbf{z}_{1:k-1,t}, \mathbf{z}_{1:t-1}, c_{1:k-1,t}^{(i)}, c_{1:t-1}^{(i)}, \widetilde{c}_{k,t}^{(i,j)}\right) p\left(\widetilde{c}_{k,t}^{(i,j)} \middle| c_{1:k-1,t}^{(i)}, c_{1:t-1}^{(i)}\right) \tag{13}$$

  - Keep the N particles $\left(c_{1:k-1,t}^{(i)}, c_{1:t-1}^{(i)}, \widetilde{c}_{k,t}^{(i,j)}\right)$ with highest weights $\widetilde{w}_{k,t}^{(i,j)}$, rename them $\left(c_{1:k,t}^{(i)}, c_{1:t-1}^{(i)}\right)$ and denote $w_{k,t}^{(i)}$ the associated weights.

5. Applications

In this Section, we demonstrate the models and algorithms on simulated data, modeling of the spread of a disease, and multi-object tracking.

5.1 Sequential Time-Varying Density Estimation

We consider the synthetic problem of estimating sequentially time-varying densities F_t on the real line using observations $\mathbf{z}_t$. We assume the observations $\mathbf{z}_t$ (where $n = 1$) follow a time-varying Dirichlet process with both uniform and size-biased deletion, a Gaussian mixed density and normal-inverse Wishart base distribution, whose pdf is given by

$$p(\mu, \Sigma|\mu_0, \kappa_0, \nu_0, \Lambda_0) \propto |\Sigma|^{-\frac{\nu_0+p+1}{2}} \exp\left[-\frac{1}{2} tr(\Lambda_0\Sigma^{-1} - \frac{\kappa_0}{2}(\mu-\mu_0)^T\Sigma^{-1}(\mu-\mu_0))\right]$$

where $p = 1$ in this case. To keep the presentation simple, we assume here that the hyperparameters of the base distribution are assumed fixed and known $\mu_0 = 0$, $\kappa_0 = 0.1$, $\nu_0 = 2$ and $\Lambda_0 = 1$. Following Pitt et al. (2002), we introduce a time-varying model on the cluster location using r auxiliary variables $\omega_{j,t}$, $j = 1, \ldots, r$

$$\begin{aligned} \omega_{j,t} &\sim \mathcal{N}\left(\mu_t, \frac{\Sigma_t}{\tau}\right) \text{ for all } j \\ (\mu_{t+1}, \Sigma_{t+1})|\omega_{1:r,t} &\sim \mathcal{N}iW(\mu_0', \kappa_0', \nu_0', \Lambda_0') \end{aligned}$$

where $\tau = 0.5$ and $r = 4000$ are fixed parameters, $\kappa_0' = \kappa_0 + r\tau$, $\nu_0' = \nu_0 + r$, $\mu_0' = \frac{r\tau}{\kappa_0+r\tau}\overline{w} + \frac{\kappa_0}{\kappa_0+r\tau}\mu_0$ and $\Lambda_0' = \Lambda_0 + \frac{\kappa_0 r\tau}{\kappa_0+r\tau}(\overline{\omega}-\mu_0)(\overline{\omega}-\mu_0)^T + \tau\sum_{j=1}^n(\omega_{j,t}-\overline{\omega})(\omega_{j,t}-\overline{\omega})^T$ and $\overline{\omega} = \frac{1}{n}\sum_{j=1}^r \omega_{j,t}$. The DP scale parameter is $\theta = 3$. Instead of fixing ρ, we assume it is time-varying with $\rho_t|\rho_{t-1} \sim \mathcal{B}(a_\rho, a_\rho\frac{1-\rho_{t-1}}{\rho_{t-1}})$ where $a_\rho = 1000$, such that $\mathbb{E}[\rho_t|\rho_{t-1}] = \rho_{t-1}$ and $var(\rho_t|\rho_{t-1}) = \frac{\rho_{t-1}^2(1-\rho_{t-1})}{a_\rho+\rho_{t-1}}$. Note that the resulting model is still first-order stationary. We select a mixture of uniform and cluster deletions with $\xi = 0.98$. The observations $\mathbf{z}_t$ are generated for $t = 1, \ldots, 1000$ from a sequence of mixtures of normal distributions, see Figure 4. Abrupt changes occur at times $t = 301$ and $t = 601$ where modes of the true density appear/disappear whereas the mode moves smoothly from 0 to -1.5 between $t = 701$ and $t = 850$. For illustration purposes, we compute the average number of alive allocation variables $N_{t|t}$ as follows

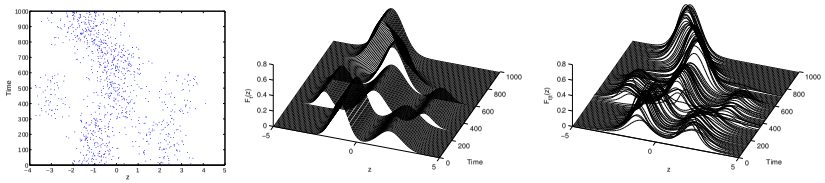
$$N_{t|t} = \mathbb{E}\left[\sum_{k=1}^{K_t} m_{k,t}^t \middle| \mathbf{z}_{1:t}\right]. \tag{14}$$

A SMC algorithm is implemented with 1000 particles (Doucet et al., 2001); this algorithm is a slight generalization of the Algorithm 2 where ρ_t is also sampled. Results are displayed in Figure 4 and the estimates of $N_{t|t}$ and $\rho_{t|t} = \mathbb{E}[\,\rho_t|\,\mathbf{z}_{1:t}]$ are plotted in Figure 5.

In Figure 4, we display the filtered density estimate $F_{t|t} = \mathbb{E}[F_t|\mathbf{z}_{1:t}]$ which manages to track the slow and abrupt changes of the true density. In Figure 5, we see that the model adapts to F_t quickly by also estimating ρ_t: ρ_t suddenly decreases at times $t = 300$ where the modes of the density suddenly change. $N_{t|t}$ follows a similar evolution: whenever F_t does not evolve, we use as many previously collected observations as possible to estimate the density by letting $N_{t|t}$ increases. When F_t changes abruptly, $N_{t|t}$ also decreases abruptly and the model quickly gets rid of the old clusters. Moreover, the cluster deletion procedure allows us to only delete irrelevant clusters. This is illustrated at $t = 600$ where the two minor modes disappear whereas the main mode is preserved.

5.2 Foot and Mouth Disease

Foot and Mouth disease is a highly transmissible viral infection which can spread rapidly among livestock. The epidemic which started in February 2001 and ended in October 2001 saw 2000 cases of death in farms throughout England. Although complex models have been designed in the epidemiology literature relying for example on the spatial distribution of farms (Keeling et al., 2001), we propose to use our model to estimate clusters of cases of Foot and Mouth disease. Let $\mathbf{z}_{k,t}$ be the two-dimensional geographical position of case k



(a) Data (b) True density (c) Estimated density

Figure 4: (a) Data (b) True density and (c) filtered density estimate. Abrupt changes occur at times $t = 301$ and $t = 601$. The mode of the density evolves smoothly between times $t = 700$ and $t = 850$.

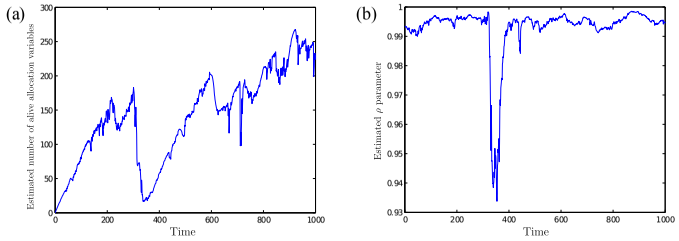
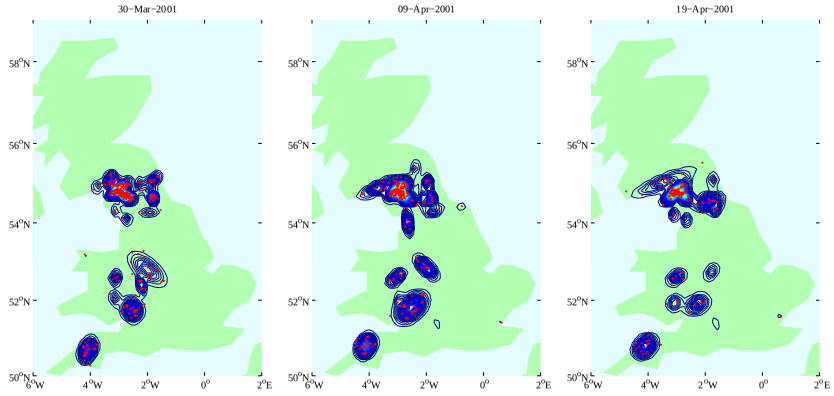


Figure 5: (a) Time evolution of the average number of alive clusters. (b) Time evolution of $\rho_{t|t}$.

at day number t. We assume that $\mathbf{z}_t$ follows a time-varying Dirichlet process mixture with a Gaussian mixed pdf whereas $\mathbb{H}$ is a Normal inverse Wishart distribution. We consider a deterministic deletion model where $r = 7$; seven days being an upper bound for the incubation of the disease. For the cluster locations, we use the model defined by Eq. (12). Inference is performed using the $N-$best algorithm 3 with $N = 1000$ particles. The marginal maximum a posteriori estimates of the cluster locations for some days during the epidemic are represented in Figure 6.

We computed the predictive cumulative density function $\Pr(\mathbf{z}_{k,t} \leq z | \mathbf{z}_{1:k-1,t}, \mathbf{z}_{1:t-1})$ for each disease case. The inverse Gaussian cumulative density function transform of it is plotted in Figure 7 for each of the two dimensions. As can be shown, our simple model does fit the data relatively well, considering that no prior information about farm locations is used here.



(a) (b) (c)

Figure 6: Evolution of the predictive density over time. Red dots represent foot and mouth cases over the last 7 days.

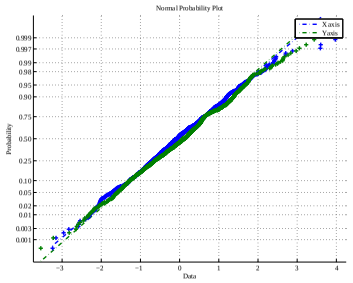


Figure 7: Normal qq-plot of the transformed predictive distribution for foot-and-mouth disease dataset for the two dimensions.

5.3 Object Tracking in Videos

The time-varying Pitman-Yor process can also be used as a prior in models used to find, track, and learn representations of arbitrary objects in a video without a predefined method

for object detection[2]. We present a model that localizes objects via unsupervised tracking while learning a representation of each object, avoiding the need for pre-built detectors. This model uses the Pitman-Yor prior to capture the uncertainty in the number and appearance of objects and requires only spatial and color video data that can be efficiently extracted via frame differencing. Nonparametric mixture models have been used in the past for a variety of computer vision tasks, including tracking (Topkaya et al., 2013; Neiswanger et al., 2014), trajectory clustering (Wang et al., 2011), and video retrieval (Li et al., 2008).

To find and track arbitrary video objects, we model spatial and color features that are extracted as objects travel within a video scene. The model isolates independently moving video objects and learns object models for each that capture their shape and appearance. The learned object models allow for tracking through occlusions and in crowded videos. The unifying framework is a time-varying Pitman-Yor process mixture, where each component is a (time-varying) object model. This setup allows us to estimate the number of objects in a video and track moving objects that may undergo changes in orientation, perspective, and appearance.

We describe the form of the extracted data, define the components of the model, and demonstrate this model on three real video datasets: a video of foraging ants, where we show improved performance over other detection-free methods; a human tracking benchmark video, where we show comparable performance against object-specific methods designed to detect humans; and a T cell tracking task where we demonstrate our method on a video with a large number of objects and show how our unsupervised method can be used to automatically train a supervised object detector.

Data. At each frame t, we assume we are given a set of N_t foreground pixels, extracted via some background subtraction method (such as those detailed in (Yilmaz et al., 2006)). These methods primarily segment foreground objects based on their motion relative to the video background. For example, an efficient method applicable for stationary videos is frame differencing: in each frame t, one finds the pixel values that have changed beyond some threshold, and records their positions $\mathbf{z}^s_{t,n} = (z^{s_1}_{t,n}, z^{s_2}_{t,n})$. In addition to the position of each foreground pixel, we extract color information. The spectrum of RGB color values is discretized into V bins, and the local color distribution around each pixel is described by counts of surrounding pixels (in an $m \times m$ grid) that fall into each color bin, denoted $\mathbf{z}^c_{t,n} = (z^{c_1}_{t,n}, \ldots, z^{c_V}_{t,n})$. Observations are therefore of the form

$$\mathbf{z}_{t,n} = (\mathbf{z}^s_{t,n}, \mathbf{z}^c_{t,n}) = (z^{s_1}_{t,n}, z^{s_2}_{t,n}, z^{c_1}_{t,n}, \ldots, z^{c_V}_{t,n}). \tag{15}$$

Examples of spatial pixel data extracted via frame differencing are shown in Figure 8 (a)-(g).

The time-varying Pitman-Yor process mixture of objects We define an object model, $\mathrm{F}(U_{k,t})$, which is a distribution over pixel data, where $U_{k,t}$ represents the parameters of the k^{th} object at time t. We wish to keep our object model general enough to be applied to arbitrary video objects, but specific enough to learn a representation that can aid in tracking. In this paper, we model each object with

$$\mathbf{z}_{t,n} \sim \mathrm{F}(U_{k,t}) = \mathcal{N}(\mathbf{z}^s_{t,n}|\mu_{k,t}, \Sigma_{k,t}) Mult(\mathbf{z}^c_{t,n}|\delta_{k,t}) \tag{16}$$

2. A preliminary version of this work has been presented as a conference paper (Neiswanger et al., 2014). Matlab code is available at `https://github.com/willieneis/DirichletProcessTracking`.

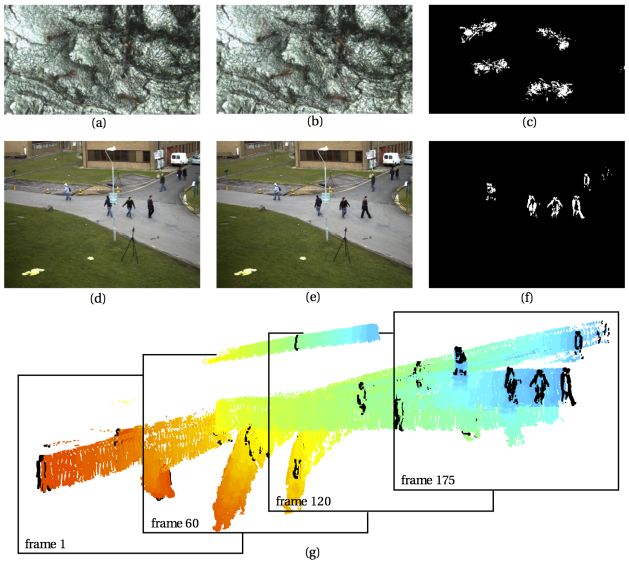


Figure 8: (a - f) Two pairs of consecutive frames and the spatial observations $\mathbf{z}^s_{t,n}$ extracted by taking the pixel-wise frame difference between each pair. (g) The results of frame differencing over a sequence of images (from the PETS2010 dataset).

where object parameters $U_{k,t} = \{\mu_{k,t}, \Sigma_{k,t}, \delta_{k,t}\}$, and $\sum_{j=1}^{V} \delta^j_{k,t} = 1$. The object model captures the objects' locus and extent with the multivariate Gaussian and color distribution with the multinomial. We demonstrate in the following experiments that this representation can capture the physical characteristics of a wide range of objects while allowing objects with different shapes, orientations, and appearances to remain isolated during tracking. We define $\mathbb{H}$, which represents the prior distribution over object parameters, to be

$$\mathbb{H}(\mu_{k,t}, \Sigma_{k,t}, \delta_{k,t}|\mu_0, \kappa_0, \nu_0, \Lambda_0, q_0) = \mathcal{N}iW(\mu_{k,t}, \Sigma_{k,t}|\mu_0, \kappa_0, \nu_0, \Lambda_0)\mathcal{D}(\delta_{k,t}|q_0). \tag{17}$$

To satisfy stationarity, we introduce a set of M auxiliary variables $\mathbf{a}^k_t = (a^k_{t,1}, \ldots, a^k_{t,M})$ for cluster k at time t (Pitt and Walker, 2005). Due to the form of the object parameter priors (i.e the base distribution of the Dirichlet process) and the object model, we can easily apply this auxiliary variable method. With this addition, object parameters do not directly depend on their values at a previous time, but are instead dependent through an intermediate sequence of variables. This allows the cluster parameters at each time step to be marginally distributed according to the base distribution $\mathbb{H}$ while maintaining simple time varying behavior. We can therefore sample from the transition kernel using

$U_{k,t} \sim T(U_{k,t-1}) = T_2 \circ T_1(U_{k,t-1})$, where

$$a^k_{t,1:M} \sim T_1(U_{k,t-1}) = \mathcal{N}(\mu_{k,t-1}, \frac{\Sigma_{k,t-1}}{\tau}) Mult(\delta_{k,t-1}) \quad (18)$$

$$\mu_{k,t}, \Sigma_{k,t}, \delta_{k,t} \sim T_2(a^k_{t,1:M}) = \mathcal{N}iW(\mu_M, \kappa_M, \nu_M, \Lambda_M)\mathcal{D}(q_M) \quad (19)$$

where we choose $\tau = 1$ for all experiments, and formulae for $\mu_M, \kappa_M, \nu_M, \Lambda_M$ and q_M are given in Appendix B.

We apply uniform deletion of allocation variables in this application, to maintain generality of the method over a variety of object types in videos. Additionally, we chose hyperparameter values by simulating from the prior and inspecting the simulated parameters for similarity with existing data. We found that the algorithm performance was not very sensitive to the hyperparameter values settings, and remained fairly consistent over a wide range of settings. In the following experiments we perform inference using both the SMC and PMCMC inference algorithms with $N = 100$ particles, and compare performance of both algorithms.

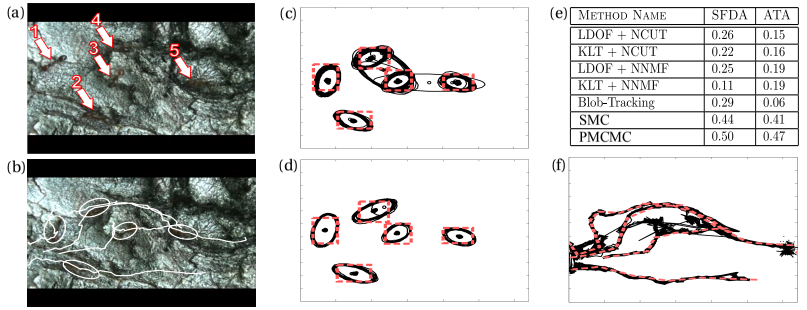
Detection-free comparison methods. Detection-free tracking strategies aim to find and track objects without any prior information about the objects' characteristics nor any manual initialization. One type of existing strategy uses optical flow or feature tracking algorithms to produce short tracklets, which are then clustered into full object tracks. We use implementations of Large Displacement Optical Flow (LDOF) (Brox and Malik, 2011) and the Kanade-Lucas-Tomasi (KLT) feature tracker (Tomasi and Kanade, 1991) to produce tracklets[3]. Full trajectories are then formed using the popular normalized-cut (NCUT) method (Shi and Malik, 2000) to cluster the tracklets or with a variant that uses non-negative matrix factorization (NNMF) to cluster motion using tracklet velocity information (Cheriyadat and Radke, 2009)[4]. We also compare against a detection-free blob-tracking method, where extracted foreground pixels are segmented into components in each frame (Stauffer and Grimson, 2000) and then associated with the nearest neighbor criterion (Yilmaz et al., 2006).

Performance metrics. For quantitative comparison, we report two commonly used performance metrics for object detection and tracking, known as the sequence frame detection accuracy (SFDA) and average tracking accuracy (ATA) (Kasturi et al., 2008). These metrics compare detection and tracking results against human-authored ground-truth, where SFDA$\in [0,1]$ corresponds to detection performance and ATA$\in [0,1]$ corresponds to tracking performance, the higher the better. We authored the ground-truth for all videos with the Video Performance Evaluation Resource (ViPER) tool (Doermann and Mihalcik, 2000).

Insect tracking. The video contains six ants with a similar texture and color distribution as the background. The ants are hard to discern, and it is unclear how a predefined detection criteria might be constructed. Further, the ants move erratically and the spatial data extracted via frame differencing does not yield a clear segmentation of the

3. The LDOF implementation can be found at `http://www.seas.upenn.edu/~katef/LDOF.html` and the KLT implementation at `http://www.ces.clemson.edu/~stb/klt/`.
4. The NCUT implementation can be found at `http://www.cis.upenn.edu/~jshi/software/` and the NNMF implementation at `http://www.ornl.gov/~czx/research.html`.

objects in individual frames. A still image from the video, with ant locations shown, is given in Figure 3(a).



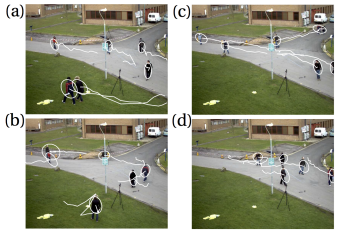
Method Name	SFDA	ATA
LDOF + NCUT	0.26	0.15
KLT + NCUT	0.22	0.16
LDOF + NNMF	0.25	0.19
KLT + NNMF	0.11	0.19
Blob-Tracking	0.29	0.06
SMC	0.44	0.41
PMCMC	0.50	0.47

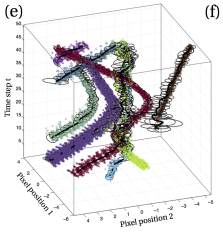
Figure 9: The ants in (a) are difficult to discern (positions labeled). We plot 100 samples from the inferred posterior over object parameters (using SMC (c) and PMCMC (d)) with ground-truth bounding boxes overlaid (dashed). PMCMC proves to give more accurate object parameter samples. We also plot samples over object tracks (sequences of mean parameters) using PMCMC in (f) , and its MAP sample in (b). We show the SFDA and ATA scores for all comparison methods in (e).

We compare the SMC and PMCMC inference algorithms, and find that PMCMC yields more accurate posterior samples (3(d)) than SMC (3(c)). Note that we run PMCMC as described in Section 4 for 100 iterations, where the first pass is equivalent to the SMC algorithm. Ground-truth bounding boxes (dashed) are overlaid on the posterior samples. The MAP PMCMC sample is shown in 3(b) and posterior samples of the object tracks are shown in 3(f), along with overlaid ground-truth tracks (dashed). SFDA and ATA performance metrics for all comparison methods are shown in 3(e). Our model yields higher metric values than all other detection-free comparison methods, with PMCMC inference scoring higher than SMC. The comparison methods seemed to suffer from two primary problems: very few tracklets could follow object positions for an extended sequence of frames, and clustering tracklets into full tracks sharply decreased in accuracy when the objects came into close contact with one another.

Comparisons with detector-based methods. Next, we aim to show that our general-purpose algorithm can compete against state of the art object-specific algorithms, even when it has no prior information about the objects. We use a benchmark human-tracking video from the International Workshop on Performance Evaluation of Tracking and Surveillance (PETS) 2009-2013 conferences (Ellis and Ferryman, 2010), due to its prominence in a number of studies (listed in Figure 10(f)). It consists of a monocular, stationary camera, 794 frame video sequence containing a number of walking humans. Due to the large number of frames and objects in this video, we perform inference with the SMC algorithm only.

Our model is compared against ten object-specific detector-based methods from the PETS conferences. These methods all either leverage assumptions about the orientation, position, or parts of humans, or explicitly use pre-trained human detectors. For example, out of the three top scoring comparison methods, (Breitenstein et al., 2009) uses a state of the art pedestrian detector, (Yang et al., 2009) performs head and feet detection, and (Conte et al., 2010) uses assumptions about human geometry and orientation to segment humans and remove shadows.





(f)

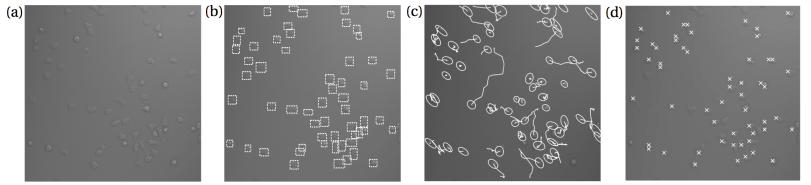
Method Name	SFDA	ATA
Breitenstein	0.57	0.30
Yang	0.55	0.45
Conte	0.53	0.06
SMC	0.51	0.30
Berclaz	0.48	0.15
Alahi 1	0.43	0.04
Alahi 2	0.42	0.05
Bolme 1	0.41	NA
Ge	0.38	0.04
Bolme 2	0.34	NA
Arsic	0.18	0.02

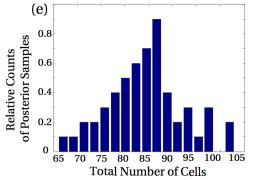
Figure 10: Results on the PETS human tracking benchmark dataset and comparison with object-detector-based methods. The MAP object parameter samples are overlaid on four still video frames (a-d). The MAP object parameter samples are also shown for a sequence of frames (a 50 time-step sequence) along with spatial pixel observations (e) (where the assignment variables $c_{t,n}$ for each pixel are represented by marker type and color). The SFDA and ATA performance metric results for our model and ten human-specific, detection-based tracking algorithms are shown in (f), demonstrating that the our model achieves comparable performance to these human-specific methods. Comparison results were provided by the authors of (Ellis and Ferryman, 2010).

In Figure 4(a-d), the MAP sample from the posterior distribution over the object parameters is overlayed on the extracted data over a sequence of frames. The first 50 frames from the video are shown in 4(e), where the assignment of each data point is represented by color and marker type. We show the SFDA and ATA values for all methods in 4(f), and can see that our model yields comparable results, receiving the fourth highest SFDA score and tying for the second highest ATA score.

Tracking populations of T cells. Automated tracking tools for cells are useful for cell biologists and immunologists studying cell behavior (Manolopoulou et al., 2012). We present results on a video containing T cells that are hard to detect using conventional methods due to their low contrast appearance against a background (Figure 5(a)). Furthermore, there are a large number of cells (roughly 60 per frame, 92 total). In this experiment, we aim to demonstrate the ability of our model to perform a tough detection task while scaling up to a large number of objects. Ground-truth bounding boxes for the cells at a single frame are shown in 5(b) and PMCMC inference results (where the MAP sample is

plotted) are shown in in 5(c). A histogram illustrating the inferred posterior over the total number of cells is shown in 5(e). It peaks around 87, near the true value of 92 cells.





(f)

Method Name	SFDA	ATA
LDOF + NCUT	0.17	0.02
KLT + NCUT	0.18	0.08
LDOF + NNMF	0.09	0.03
KLT + NNMF	0.1	0.09
Blob-Tracking	0.24	0.03
SMC	0.31	0.11
PMCMC	0.36	0.19

Figure 11: T cells are numerous, and hard to detect due to low contrast images (a). For a single frame, ground-truth bounding boxes are overlaid in (b), and inferred detection and tracking results are overlaid in (c). A histogram showing the posterior distribution over the total number of cells is shown in (e). The SFDA and ATA for the detection-free comparison methods are shown in (f). Inferred cell positions (unsupervised) were used to automatically train an SVM for supervised cell detection; SVM detected cell positions for a single frame are shown in (d).

Manually hand-labeling cell positions to train a detector is feasible but time consuming; we show how unsupervised detection results from our model can be used to automatically train a supervised cell detector (a linear SVM), which can then be applied (via a sliding window across each frame) as a secondary, speedy method of detection (Figure 5(d)). This type of strategy in conjunction with our model could allow for an ad-hoc way of constructing detectors for arbitrary objects on the fly, which could be taken and used in other vision applications, without needing an explicit predefined algorithm for object detection.

6. Discussion

In this article, we have presented a class of first-order stationary Pitman-Yor processes for time-varying density estimation and clustering. These models are based on a simple generalized Pólya urn sampling scheme whose validity follows from the consistence properties under specific deletion rules of the two-parameter Ewens sampling formula. We have proposed SMC and PMCMC methods to fit these models and have demonstrated them on several applications. The model proposed in the present paper has also been successfully

applied to dynamic spike sorting (Gasthaus et al., 2008) and to cell fluorescent microscopic imaging tracking (Ji and West, 2009).

There are numerous potential extensions to this work. We have focused on specifying directly the marginal distribution of the allocation variables, the underlying infinite-dimensional process $\{\mathbb{G}_t\}$ being integrated out. This has allowed us to develop intuitive models and simple algorithms to fit them. However, it would be interesting to explore whether it is possible to obtain an explicit representation for $\{\mathbb{G}_t\}$ and whether this can be related to the class of models described in (Griffin, 2011).

The uniform and deterministic deletion steps can be applied to any hierarchical model, as long as the predictive distribution is known. We could therefore develop time-varying versions of exchangeable models such as the Bernoulli trips introduced by Walker et al. (1999). The same methodology could also be extended to dynamic social networks. In this context, a classical model is Friend I, which assumes a finite Pólya Urn scheme as the reinforcement procedure for friends interactions (Skyrms and Pemantle, 2000). In order to take into account fading memory, a model called 'discounted from the past' has been proposed, where the weights of the Pólya Urn are discounted at each time step– an approach similar to that of Zhu et al. (2005). However this model has poor asymptotic properties, as each individual always ends up being friend with a single individual (Skyrms and Pemantle, 2000). We conjecture that uniform deletion rules would have nicer properties.

Finally in some applications, it would be interesting to develop models allowing clusters to merge and split over time. We believe that it is possible to build first-order stationary models including such merge-split mechanisms exploiting once more the remarkable properties of the Poisson-Dirichlet distributions under splitting and merging transformations; see e.g. (Pitman, 2002).

Acknowledgments

The authors thank the Action Editor and referees for their helpful comments and Dr Miles Thomas, DEFRA Food and Environment Research Agency, for providing the Foot-and-Mouth data. F. Caron acknowledges the support of the European Commission under the Marie Curie Intra-European Fellowship Programme[5]. A. Doucet's research is partly funded by EPSRC (EP/K000276/1 and EP/K009850/1). Part of this work has been supported by the BNPSI ANR project ANR-13-BS-03-0006-01.

Appendix A. Extensions

A.1 Species Sampling Priors

The consistence properties under size-biased deletion are remarkable properties specific to the two-parameter Ewens sampling formula (Gnedin and Pitman, 2005). However, the uniform and deterministic deletion steps can be extended to any model with a given prediction rule which ensures exchangeability over the data. In particular, the class of species sampling priors - which includes the two-parameter Ewens sampling formula - is of particular interest as it enjoys these two properties (Pitman, 1996; Lee, 2009). A sequence $\{\mathbf{y}_n\}$ is called a

5. The contents reflect only the authors views and not the views of the European Commission.

species sampling sequence if the following prediction rule holds

$$\begin{aligned}\mathbf{y}_1 &\sim \mathbb{G}_0 \\ \mathbf{y}_{n+1}|\mathbf{y}_1,\ldots,\mathbf{y}_n &\sim \sum_{j=1}^{k} \frac{p(n_1,\ldots,n_j+1,\ldots,n_k)}{p(n_1,\ldots,n_j,\ldots,n_k)}\delta_{U_j} + \frac{p(n_1,\ldots,n_j,\ldots,n_k,1)}{p(n_1,\ldots,n_j,\ldots,n_k)}\mathbb{G}_0 \end{aligned} \quad (20)$$

where $\mathbb{G}_0$ is a base distribution, U_j and n_j, $j = 1,\ldots k$, are respectively the set of distinct values within $\mathbf{y}_{1:n}$ and their relative occurrences. Here $p : \cup_{k=1}^{\infty}\mathbb{N}^k \to [0,1]$ is a *symmetric* function of k-tuples of non-negative integers with sum n called the *Exchangeable Partition Probability Function*. This function must satisfy

$$p(1) = 1 \text{ and } p(n_1,\ldots,n_k) = \sum_{j=1}^{k} p(n_1,\ldots,n_j+1,\ldots,n_k) + p(n_1,\ldots,,n_k,1).$$

To obtain first-order stationary species sampling process, we can apply the uniform and deterministic deletion steps followed by the predictive step (20).

A.2 Time-varying Hierarchical and Coloured Dirichlet Processes

Various hierarchical extensions of the Dirichlet process have been proposed such as the popular hierarchical Dirichlet process (Teh et al., 2006) and the coloured Dirichlet process (Green, 2010). We show here how time-varying stationary versions of these models can easily be designed.

The hierarchical Dirichlet process consists of embedded Dirichlet processes where we have at the top of the hierarchy

$$\mathbb{G} \sim \mathrm{DP}(\alpha, \mathbb{G}_0)$$

then for each group $k = 1,\ldots,d$

$$\mathbb{G}_k|\mathbb{G} \sim \mathrm{DP}(a, \mathbb{G})$$

and finally for each item $i = 1,\ldots,n_k$ within group k

$$\begin{aligned}\mathbf{y}_{i,k}|\mathbb{G}_k &\sim \mathbb{G}_k \\ \mathbf{z}_{i,k}|\mathbf{y}_{i,k} &\sim f(\cdot|\mathbf{y}_{i,k}).\end{aligned}$$

A popular application of this model is topic clustering. In this case, a group k represents a document and data $\mathbf{z}_{i,k}$ represent words within documents. The 'mother' mixing distribution $\mathbb{G}$ represents the overall mixture over topics, and each 'child' mixing distribution $\mathbb{G}_k$ represents the mixture associated to a document k, sampled from a Dirichlet process of base measure $\mathbb{G}$. This model can be straightforwardly extended to take into account a time-varying base measure $\mathbb{G}_t$. Similarly to (Teh et al., 2006), a 'Chinese restaurant franchise' formulation can be expressed based on the set of alive allocation variables at time t.

The coloured Dirichlet process consists of a finite mixture of Dirichlet process mixtures where

$$\pi_{1:K} \sim \mathcal{D}(a_1,\ldots,a_K).$$

Here $\mathcal{D}$ denotes the Dirichlet distribution. For each group $k = 1, \ldots, K$, we have

$$\mathbb{G}_k \sim \mathrm{DP}(\alpha_k, \mathbb{G}_{0,k}).$$

Finally for each item $i = 1, \ldots, n$, we have

$$\begin{aligned} c_i|\pi_{1:K} &\sim \pi_{1:K}, \\ \mathbf{y}_i|c_i &\sim \mathbb{G}_{c_i}, \\ \mathbf{z}_i|\mathbf{y}_i &\sim f(\cdot|\mathbf{y}_i). \end{aligned}$$

The main application of such models is for clustering data in presence of some background noise. Again, a time-varying version of this model, where the weights and cluster locations evolve over time, can straightforwardly be developed by deleting allocation variables as described in Section 3 and sampling them conditionally on the set of alive variables as described in (Green, 2010).

Appendix B. Auxiliary Variable Formulae

The parameters $\mu_M, \kappa_M, \nu_M, \Lambda_M$ and q_M in Eq. (19) can be written as

$$\kappa_M = \kappa_0 + M \tag{21}$$

$$\nu_M = \nu_0 + M \tag{22}$$

$$\mu_M = \frac{\kappa_0}{\kappa_0 + M}\mu_0 + \frac{M}{\kappa_0 + M}\overline{\mathbf{a}}^s \tag{23}$$

$$\Lambda_M = \Lambda_0 + S_{\mathbf{a}^s} \tag{24}$$

$$q_M = q_0 + \sum_{i=1}^{M} \mathbf{a}_i^c \tag{25}$$

where $\mathbf{a}^s$ and $\mathbf{a}^c$ respectively denote the spatial and color components of the auxiliary variables, and $\overline{\mathbf{a}}$ and $S_{\mathbf{a}}$ respectively denote the sample mean and sample covariance of the auxiliary variables.

Appendix C. Kolmogorov-Consistent Model

As mentioned in Section 3, the model is not Kolmogorov-consistent. Consider a sequence of distributions $\pi_n(\mathbf{c}_{1:n,1:t})$, $n = 1, 2, \ldots$, where there are n allocation variables at each time t, then, except in the trivial cases $\rho = 0$ or $\rho = 1$

$$\sum_{\mathbf{c}_{n,1:t}} \pi_n(\mathbf{c}_{1:n,1:t}) \neq \pi_{n-1}(\mathbf{c}_{1:n-1,1:t}). \tag{26}$$

It is possible to construct a slightly different model that satisfies the above equality. Consider that there are p latent variables $\widetilde{c}_{k,t}$ which evolve according to the generalized Pólya urn defined in Algorithm 1. We denote by $\widetilde{\mathbf{c}}_{1:t-1}^{t-1}$ the subset of $\widetilde{\mathbf{c}}_{1:t-1}$ corresponding to variables having survived the deletion steps from time 1 to $t-1$, and we denote by $\widetilde{\mathbf{c}}_{1:t-1}^{t}$ the subset corresponding to those having survived from time 1 to t. Let $\widetilde{K}_{t-1}$ be the number of

clusters created from time 1 to $t-1$. We denote by $\widetilde{\mathbf{m}}_{t-1}^{t-1}$ the vector of size $\widetilde{K}_{t-1}$ containing the size of the clusters associated to $\widetilde{\mathbf{c}}_{1:t-1}^{t-1}$, and we denote by $\widetilde{\mathbf{m}}_{t-1}^{t}$ the vector containing the size of clusters associated to $\widetilde{\mathbf{c}}_{1:t-1}^{t}$.

Conditional on $(\widetilde{\mathbf{c}}_{1:t-1}^{t}, \widetilde{c}_t)$ (or equivalently $(\widetilde{\mathbf{m}}_{t-1}^{t}, , \widetilde{c}_t)$), the allocation variables $c_{k,t}$ of the n observations are simply drawn from a standard Pólya urn. The whole model is represented in Figure 12. As the Markov process is on the latent variables and not on the allocation variables $c_{k,t}$ associated to observations, this model is Kolmogorov-consistent:

$$\sum_{\mathbf{c}_{n,1:t}} \pi_n(\mathbf{c}_{1:n,1:t}) = \pi_{n-1}(\mathbf{c}_{1:n-1,1:t}). \tag{27}$$

Note that we have in this case an additional tuning parameter p. Larger values of p induce higher correlations.

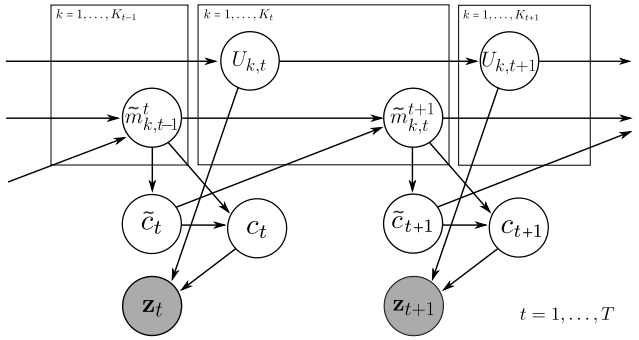


Figure 12: A representation of the time-varying Pitman-Yor process mixture as a directed graphical model, representing conditional independencies between variables. All assignment variables and observations at time t are denoted c_t and $\mathbf{z}_t$, respectively.

References

A. Ahmed and E.P. Xing. Dynamic non-parametric mixture models and the recurrent Chinese restaurant process. In *Proceedings of the Eighth SIAM International Conference on Data Mining*, 2008.

A. Alahi, L. Jacques, Y. Boursier, and P. Vandergheynst. Sparsity-driven people localization algorithm: Evaluation in crowded scenes environments. In *Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS-Winter)*, pages 1–8. IEEE, 2009.

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo for efficient numerical simulation. In *Monte Carlo and Quasi-Monte Carlo Methods*, pages 45–60. Springer, 2009.

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods (with discussion). *Journal of the Royal Statistical Society B*, 72:269–342, 2010.

C. E. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2:1152–1174, 1974.

J. Arbel, K. Mengersen, and J. Rousseau. Bayesian nonparametric dependent model for the study of diversity for species data. Technical report, arXiv preprint arXiv:1402.3093, 2014.

D. Arsic, A. Lyutskanov, G. Rigoll, and B. Kwolek. Multi camera person tracking applying a graph-cuts based foreground segmentation in a homography framework. In *Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS-Winter)*, pages 1–8. IEEE, 2009.

N. Bartlett, D. Pfau, and F. Wood. Forgetting counts: Constant memory inference for a dependent hierarchical Pitman-Yor process. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 63–70, 2010.

J. Berclaz, F. Fleuret, and P. Fua. Multiple object tracking using flow linear programming. In *Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS-Winter)*, pages 1–8. IEEE, 2009.

D. Blackwell and J.B. MacQueen. Ferguson distributions via Pólya urn schemes. *The Annals of Statistics*, 1:353–355, 1973.

D. Blei and P. Frazier. Distance dependent Chinese restaurant processes. *Journal of Machine Learning Research*, 12:2383–2410, 2011.

D.S. Bolme, Y.M. Lui, BA Draper, and JR Beveridge. Simple real-time human detection using a single correlation filter. In *Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS-Winter)*, pages 1–8. IEEE, 2009.

M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Markovian tracking-by-detection from a single, uncalibrated camera. In *Int. IEEE CVPR Workshop on Performance Evaluation of Tracking and Surveillance (PETS'09)*, 2009.

T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, 2011.

F. Caron, M. Davy, and A. Doucet. Generalized Polya urn for time-varying Dirichlet process mixtures. In *23rd Conference on Uncertainty in Artificial Intelligence*, 2007.

F. Caron, M. Davy, A. Doucet, E. Duflos, and P. Vanheeghe. Bayesian inference for linear dynamic models with Dirichlet process mixtures. *IEEE Transactions on Signal Processing*, 56(1):71–84, 2008.

A. M. Cheriyadat and R. J. Radke. Non-negative matrix factorization of partial track data for motion segmentation. In *IEEE 12th International Conference on Computer Vision*, pages 865–872. IEEE, 2009.

D.M. Cifarelli and E. Regazzini. Nonparametric statistical problems under partial exchangeability. *Annali del'Instituto di Matematica Finianziara dell'Universit di Torino*, 12:1–36, 1978.

D. Conte, P. Foggia, G. Percannella, and M. Vento. Performance evaluation of a people tracking system on pets2009 database. In *Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 119–126. IEEE, 2010.

D. Doermann and D. Mihalcik. Tools and techniques for video performance evaluation. In *15th International Conference on Pattern Recognition*, volume 4, pages 167–170. IEEE, 2000.

A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in practice*. Springer-Verlag, 2001.

D. B. Dunson and J.-H. Park. Kernel stick-breaking processes. *Biometrika*, 95:307–323, 2007.

D. B. Dunson, N. Pillai, and J.-H. Park. Bayesian density regression. *Journal of the Royal Statistical Society B*, 69(2):163–183, 2006.

A. Ellis and J. Ferryman. PETS2010 and PETS2009 evaluation of results using individual ground truthed single views. In *Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 135–142. IEEE, 2010.

M.D. Escobar and M. West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90:577–588, 1995.

P. Fearnhead. Particle filters for mixture models with an unknown number of components. *Statistics and Computing*, 14:11–21, 2004.

T.S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1: 209–230, 1973.

N. Foti and S. Williamson. A survey of non-exchangeable priors for Bayesian nonparametric models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, to appear, 2013.

J. Gasthaus, F. Wood, D. Görür, and Y.W. Teh. Dependent Dirichlet process spike sorting. In *NIPS*, 2008.

W. Ge and R.T. Collins. Evaluation of sampling-based pedestrian detection for crowd counting. In *2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS-Winter)*, pages 1–7. IEEE, 2009.

A.E. Gelfand, A. Kottas, and S.N. MacEachern. Bayesian nonparametric spatial modeling with Dirichlet process mixing. *Journal of the American Statistical Association*, 100(471):1021–1035, 2005. doi: 10.1198/016214504000002078.

A. Gnedin and J. Pitman. Regenerative partition structures. *Electronic Journal of Combinatorics*, 11:1–21, 2005.

P.J. Green. Colouring and breaking sticks: random distributions and heterogeneous clustering. In *Probability and Mathematical Genetics: Papers in Honour of Sir John Kingman*. Cambridge University Press, 2010.

P.J. Green and S. Richardson. Modelling heterogeneity with and without the Dirichlet process. *Scandinavian Journal of Statistics*, 28(2):355–375, 2001.

J. E. Griffin. The Ornstein–Uhlenbeck Dirichlet process and other time-varying processes for Bayesian nonparametric inference. *Journal of Statistical Planning and Inference*, 141(11):3648–3664, 2011.

J.E. Griffin and M.F.J. Steel. Order-based dependent Dirichlet processes. *Journal of the American Statistical Association*, 101(473):179–194, 2006.

J.E. Griffin and M.F.J. Steel. Time-dependent stick-breaking processes. Technical report, University of Kent, 2009.

J.E. Griffin and M.F.J. Steel. Stick-breaking autoregressive processes. *Journal of Econometrics*, 162 (2):383–396, 2011.

P. Hall, H. Müller, and P. Wu. Time-dynamic density and mode estimation with applications to fast mode tracking. *Journal of Computational and Graphical Statistics*, 15:82–100, 2006.

J. H. Huggins and F. Wood. Infinite structured hidden semi-Markov models. *arXiv preprint arXiv:1407.0044*, 2014.

M. De Iorio, P. Müller, G. L. Rosner, and S.N. MacEachern. An ANOVA model for dependent random measures. *Journal of the American Statistical Association*, 99:205–215, 2004.

N. Jaoua, F. Septier, E. Duflos, and P. Vanheeghe. State and impulsive time-varying measurement noise density estimation in nonlinear dynamic systems using Dirichlet process mixtures. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1–5, Florence, Italy, 2014.

C. Ji and M. West. Dynamic spatial mixture modeling and its application in Bayesian tracking for cell fluorescent microscopic imaging. In *Joint Statistical Meeting*, 2009.

H. Joe. *Multivariate Models and Dependence Concepts*. Chapman & Hall, 1997.

M. Kalli, J.E. Griffin, and S.G. Walker. Slice sampling mixture models. *Statistics and Computing*, 21(1):93–105, 2011.

N. Kantas, A. Doucet, S. Singh, and J. M. Maciejowski. An overview of sequential Monte Carlo methods for parameter estimation in general state-space models. In *15th IFAC Symposium on System Identification (SYSID), Saint-Malo, France.*, volume 102, page 117, 2009.

R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova, and J. Zhang. Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 319–336, 2008.

M.J. Keeling, M.E.J. Woolhouse, D.J. Shaw, L. Matthews, M. Chase-Topping, D.T. Haydon, S.J. Cornell, J. Kappey, J. Wilesmith, and B.T. Grenfell. Dynamics of the 2001 UK foot and mouth epidemic: stochastic dispersal in a heterogeneous landscape. *Science*, 294:813–817, 2001.

J.F.C. Kingman. Random partitions in population genetics. *Proceedings of the Royal Society of London*, 361:1–20, 1978.

J. Lee. Species sampling model and its application to Bayesian statistics. Technical report, Seoul National University, 2009.

X. Li, W. Hu, Z. Zhang, X. Zhang, and G. Luo. Trajectory-based video retrieval using Dirichlet process mixture models. In *BMVC*, pages 1–10, 2008.

D. Lin, E. Grimson, and J. W. Fisher III. Construction of dependent Dirichlet processes based on Poisson processes. In *NIPS*. Neural Information Processing Systems Foundation (NIPS), 2010.

S.N. MacEachern. Dependent Dirichlet processes. Technical report, Dept. of Statistics, Ohio State university, 2000.

S.N. MacEachern, M. Clyde, and J.S. Liu. Sequential importance sampling for nonparametric Bayes models: the next generation. *Canadian Journal of Statistics*, 27(2):251–267, 1999.

I. Manolopoulou, M. P. Matheu, M. D. Cahalan, M. West, and T. B. Kepler. Bayesian spatio-dynamic modeling in cell motility studies: Learning nonlinear taxic fields guiding the immune response. *Journal of the American Statistical Association*, 107(499):855–865, 2012.

P. Müller and F. Quintana. Nonparametric Bayesian data analysis. *Statistical Science*, pages 95–110, 2004.

P. Müller, F. Quintana, and G. Rosner. A method for combining inference across related nonparametric Bayesian models. *Journal of the Royal Statistical Society B*, 66:735–749, 2004.

W. Neiswanger, F. Wood, and E. Xing. The dependent Dirichlet process mixture of objects for detection-free tracking and object modeling. In *Artificial Intelligence and Statistics*, 2014.

E. Ozkan, I. Y. Ozbek, and M. Demirekler. Dynamic speech spectrum representation and tracking variable number of vocal tract resonance frequencies with time-varying Dirichlet process mixture models. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(8):1518–1532, 2009.

O. Papaspiliopoulos and G.O. Roberts. Retrospective Markov chain Monte Carlo methods for Dirichlet process hierarchical models. *Biometrika*, 95(1):169–186, 2008.

J. Pitman. Exchangeable and partially exchangeable random partitions. *Probability Theory and Related Fields*, 102:145–158, 1995.

J. Pitman. Some developments of the Blackwell-MacQueen urn scheme. In *Statistics, Probability and Game Theory; Papers in Honor of David Blackwell*. Institute of Mathematical Statistics, California, 1996.

J. Pitman. Probabilistic bounds on the coefficients of polynomials with only real zeros. *Journal of Combinatorial Theory*, 77:279–303, 1997.

J. Pitman. Poisson-Dirichlet and GEM invariant distributions for split-and-merge transformations of an interval partition. *Combinatorics, Probability and Computing*, 11:501–514, 2002.

J. Pitman and M. Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, 25:855–900, 1997.

M.K. Pitt and S.G. Walker. Constructing stationary time series models using auxiliary variables with applications. *Journal of the American Statistical Association*, 100(470):554–564, 2005.

M.K. Pitt, C. Chatfield, and S.G. Walker. Constructing first order stationary autoregressive models via latent processes. *Scandinavian Journal of Statistics*, 29:657–663, 2002.

G. Poyiadjis, A. Doucet, and S. Singh. Particle approximations of the score and observed information matrix in state space models with application to parameter estimation. *Biometrika*, 98(1):65–80, 2011.

V. Rao and Y.W. Teh. Spatial normalized gamma processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

A. Rodriguez and D. B. Dunson. Nonparametric Bayesian models through probit stick-breaking processes. *Bayesian Analysis*, 6(1), 2011.

J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

B. Skyrms and R. Pemantle. A dynamic model of social network formation. *Proceedings of the National Academy of Sciences*, 97:9340–9346, 2000.

N. Srebro and S. Roweis. Time-varying topic models using dependent Dirichlet processes. Technical report, Department of Computer Science, University of Toronto, 2005.

C. Stauffer and E. L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–757, 2000.

Y. W. Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, 2006.

Y. W. Teh, C. Blundell, and L. Elliott. Modelling genetic variations using fragmentation-coagulation processes. In *Advances in Neural Information Processing Systems (NIPS)*, pages 819–827, 2011.

Y.W. Teh and M.I. Jordan. Hierarchical Bayesian nonparametric models with applications. *Bayesian Nonparametrics*, pages 158–207, 2010.

Y.W. Teh, M.I. Jordan, M.J. Beal, and D.M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101:1566–1581, 2006.

C. Tomasi and T. Kanade. *Detection and tracking of point features.* School of Computer Science, Carnegie Mellon Univ., 1991.

I. S. Topkaya, H. Erdogan, and F. Porikli. Detecting and tracking unknown number of objects with Dirichlet process mixture models and Markov random fields. In *Advances in Visual Computing*, pages 178–188. Springer, 2013.

S.G. Walker. Sampling the Dirichlet mixture model with slices. *Communications in Statistics - Simulation and Computation*, 36(1):45–54, January 2007.

S.G. Walker and P. Muliere. A bivariate Dirichlet process. *Statistics and probability letters*, 64:1–7, 2003.

S.G. Walker, P. Damien, P.W. Laud, and A.F.M. Smith. Bayesian nonparametric inference for random distributions and related functions. *Journal of the Royal Statistical Society B*, 61(3): 485–527, 1999.

X. Wang, K. T. Ma, G. Ng, and E. L. Grimson. Trajectory analysis and semantic region modeling using nonparametric hierarchical bayesian models. *International Journal of Computer Vision*, 95 (3):287–312, 2011.

J. Yang, PA Vela, Z. Shi, and J. Teizer. Probabilistic multiple people tracking through complex situations. In *11th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2009.

A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys (CSUR)*, 38(4):13, 2006.

X. Zhu, Z. Ghahramani, and J. Lafferty. Time-sensitive Dirichlet process mixture models. Technical report, Carnegie Mellon University, Pittsburgh, PA, 2005.

Journal of Machine Learning Research 18 (2017) 1-39 Submitted 7/15; Revised 2/17; Published 4/17

Particle Gibbs Split-Merge Sampling for Bayesian Inference in Mixture Models

Alexandre Bouchard-Côté BOUCHARD@STAT.UBC.CA
Department of Statistics
University of British Columbia
Corresponding address: 3182 Earth Sciences Building, 2207 Main Mall, Vancouver, BC, Canada V6T 1Z4

Arnaud Doucet DOUCET@STATS.OX.AC.UK
Department of Statistics
University of Oxford, United Kingdom

Andrew Roth ANDREW.ROTH@LUDWIG.OX.AC.UK
Department of Statistics and Ludwig Institute for Cancer Research
University of Oxford, United Kingdom

Editor: Zhihua Zhang

Abstract

This paper presents an original Markov chain Monte Carlo method to sample from the posterior distribution of conjugate mixture models. This algorithm relies on a flexible split-merge procedure built using the particle Gibbs sampler introduced in Andrieu et al. (2009, 2010). The resulting so-called Particle Gibbs Split-Merge sampler does not require the computation of a complex acceptance ratio and can be implemented using existing sequential Monte Carlo libraries. We investigate its performance experimentally on synthetic problems as well as on geolocation data. Our results show that for a given computational budget, the Particle Gibbs Split-Merge sampler empirically outperforms existing split merge methods. The code and instructions allowing to reproduce the experiments is available at `https://github.com/aroth85/pgsm`.

Keywords: Dirichlet process mixture models; Gibbs sampler; Particle Gibbs sampler; Sequential Monte Carlo.

1. Introduction

Mixture models are very commonly used to perform clustering and density estimation, and they have consequently found numerous applications in a wide range of scientific fields. Since the introduction of Markov chain Monte Carlo (MCMC) methods in statistics over twenty five years ago, the Bayesian approach to mixture models has become very popular (Marin et al., 2005; Richardson and Green, 1997). However, sampling from the posterior distribution of mixture models remains a challenging computational problem.

When conjugate priors are used, it is possible to analytically integrate out the mixing proportions and the parameters of the components. This is the scenario we will focus on in this article. In this case, we aim to sample from the posterior distribution of the latent indicator variables associated with the observations, each latent variable indicating which component of the mixture generates a given data point. A simple Gibbs sampler can be used which updates the latent indicator variables one-at-a-time (MacEachern, 1994; Escobar and West, 1995) but this algorithm is inefficient when the number of observed data points T is large. First, the simulated Markov chain would have to visit a long chain of lower probability configurations in order to split and merge large clusters. As a result, it is prone to getting trapped in severe local modes. Second, it is non-trivial to parallelize due to the inherently sequential nature of the updates.

The limitations of the simple Gibbs sampler has motivated a rich literature on MCMC algorithms for Bayesian mixture models which partially address these issues; see, e.g., Ishwaran and James (2001); Liang et al. (2007); Walker (2007); Kalli et al. (2011). In particular, procedures proposing to split and merge existing clusters in one single step have become prominent as they generally perform better than the simple Gibbs sampler (Richardson and Green, 1997; Neal, 2000; Dahl, 2003; Jain and Neal, 2004).

While designing an efficient merge proposal is simple, designing an efficient split proposal is a more complicated task. When the mixing proportions and parameters are not integrated out, split-merge moves were first proposed in Richardson and Green (1997). The proposals were built to ensure the conservation of some moments and accepted/rejected using Metropolis-Hastings steps. However, it is difficult to design efficient proposals in this context.

When the mixing proportions and parameters are integrated out, split-merge moves on the latent indicator variables were first proposed in Jain and Neal (2004). Assume one is interested in splitting a block/cluster of points $b \subset \{1, \ldots, T\}$ into two blocks. We select two points in b, which will be in distinct blocks after the split. There are $2^{|b|-2}$ possible ways to split the original block b, hence any efficient proposal needs to be informed by the observations corresponding to the indices in b. In Jain and Neal (2004), one selects two points at random which are used as anchors. When the two anchors are in separate clusters, a merging of the two clusters is proposed. When the two anchors are in the same cluster, a split is proposed as follows: first, the two anchor points seed a pair of new clusters, and second, several restricted Gibbs scans are performed to reallocate the remaining points originally clustered with the anchors to the two new clusters. All clusters which do not contain the anchors are not altered, leading to a restricted Gibbs move. After either a split or a merge is proposed, the Metropolis-Hastings ratio is computed to accept or reject the move. The number of Gibbs scans in the split move is a free tuning parameter for this sampler. In Dahl (2003), an alternative approach is proposed for split moves. The restricted Gibbs scans are replaced by a sequential allocation step whereby the anchors define two new clusters and all points which were originally clustered with these points are sequentially allocated to one of the anchor clusters.

These split-merge algorithms have become popular as they provide state-of-the-art performance but they are relatively difficult to implement due to their complex Metropolis-Hastings acceptance ratios.

In the present work, we propose a novel split-merge sampler based on the conditional Sequential Monte Carlo (SMC) algorithm appearing in the Particle Gibbs (PG) sampler (Andrieu et al., 2009, 2010), which we call the Particle Gibbs Split Merge (PGSM) sampler. Most of the complexity inherent to split-merge operators is encapsulated into the well-understood PG sampling procedure (Chopin and Singh, 2015), and no acceptance ratio

©2017 Bouchard-Côté, Doucet and Roth.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v18/15-397.html`.

needs to be computed. Moreover, as the PGSM sampler relies on SMC methods, it benefits from advanced simulation methods from the SMC literature, such as adaptation schemes (Lee, 2011) and methods for parallel and distributed inference (Lee et al., 2010; Jun et al., 2012; Lee and Whiteley, 2016), as well as from efficient SMC software libraries (Johansen, 2009; Murray, 2015). The PGSM sampler does not make any topological assumption on the observation space in contrast to the posterior simulation techniques described in Dahl (2009) and Liang et al. (2007). This methodology complements the maximum *a posteriori* inference techniques developed in Daumé III (2007); Wang and Dunson (2011).

There has been previous work on applying sequential importance sampling and SMC methods for posterior simulation of Dirichlet processes and related mixture models. However, to the best of our knowledge, SMC methods have never been previously used to design split-merge moves. Indeed, the methods proposed in MacEachern et al. (1999); Fearnhead (2004); Fearnhead and Meligkotsidou (2007); Mansinghka et al. (2007); Caron and Doucet (2009); Carvalho et al. (2010) directly apply a single pass SMC algorithm to the entire clustering problem. Empirical results in Kantas et al. (2015) suggest that such methods may require a number of particle which scales at least quadratically with respect to the number of datapoints. The work of Ülker et al. (2010) uses SMC within the context of the SMC Samplers methodology (Del Moral et al., 2006), which makes it closer in spirit to existing MCMC methods. Our contribution is to provide a principled approach for breaking down the clustering problem into smaller sub-problems more amenable to the use of SMC techniques.

Finally, other lines of work are devoted to parallelization and distribution of MCMC methods for mixture models (Chang and Fisher III, 2013; Williamson et al., 2013; Gal and Ghahramani, 2014; Ge et al., 2015). As alluded to earlier, our method can potentially be parallelized and distributed using existing approaches from the SMC literature (Lee et al., 2010; Jun et al., 2012; Lee and Whiteley, 2016). Like the other available split-merge procedures, it is also possible to consider different split-merge moves simultaneously when the prior clustering distribution restricted to the clusters being updated does not depend on the number of clusters for the whole dataset. However, we do not focus on these aspects here.

The rest of this article is organized as follows. Section 2 introduces our notation for the types of Bayesian mixture models that we consider. Section 3 details the PGSM sampler. Section 4 applies the method to synthetic datasets, as well as real data from a geolocation application. We conclude with some directions for future work and discussion in Section 5.

2. Mixture models and Bayesian inference

In this section we first layout notation and then describe Bayesian mixture models. We focus on the case where the component base measure is conjugate to the data likelihood, so that the posterior distribution of any clustering can be evaluated analytically up to a normalizing constant.

2.1 Notation

We use bold letters for (random) vectors, and normal fonts for (random) scalars, sets, and matrices. For quantities such as an individual observation y_i, or a parameter θ, which can

be either scalars or vectors without affecting our methodology, we consider them as scalars without loss of generality. Given a vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, and $i \leq j$, we use $\mathbf{x}_{i:j}$ to denote the sub-vector $\mathbf{x}_{i:j} = (x_i, x_{i+1}, \ldots, x_j)$. To simplify notation, we do not distinguish random variables from their realization. We define discrete probability distributions with their probability mass functions, and continuous probability distributions with their density functions with respect to the Lebesgue measure. A list of symbols is available in the Appendix.

2.2 Bayesian mixture model

Consider T observations $\mathbf{y} \coloneqq (y_1, \ldots, y_T)$. A mixture model assumes that the observations indices $[T] \coloneqq \{1, \ldots, T\}$ are partitioned into subsets. This partition is called a clustering, $c \coloneqq \{b_1, \ldots, b_{|c|} : b_k \subseteq [T]\}$ where $|c|$ denotes the cardinality of the set c and each block b in the partition is referred to as a cluster. Given the clustering c, we define the following likelihood for the data

$$\pi(\mathbf{y}|c) \coloneqq \prod_{b \in c} L(\mathbf{y}_b), \tag{1}$$

where $L(\mathbf{y}_b)$ is the likelihood of the observations in cluster b

$$L(\mathbf{y}_b) \coloneqq \int \left(\prod_{i \in b} L(y_i|\theta) \right) H(\,\mathrm{d}\theta). \tag{2}$$

In this expression, $L(y_i|\theta)$ is a probability density function parametrized by θ and $H(\,\mathrm{d}\theta)$ a prior measure over this parameter.

The clustering c is unknown and is viewed as a random variable. Let $\tau(c)$ denote its prior probability, defined over the space of partitions of $[T]$ and assumed to factorize as

$$\tau(c) \propto \tau_1(|c|) \prod_{b \in c} \tau_2(|b|), \tag{3}$$

where $\tau_1 : \mathbb{N} \to \mathbb{R}^+$ and $\tau_2 : \mathbb{N} \to \mathbb{R}^+$ are arbitrary functions.

This assumption on the prior clustering distribution is not restrictive and includes several popular priors, such as:

Dirichlet process prior (Ferguson, 1973) with parameter $\alpha_0 > 0$: $\tau_1(j) \propto {\alpha_0}^j$ and $\tau_2(j) \propto (j-1)!$.

Pitman-Yor process prior (Pitman and Yor, 1997; Ishwaran and James, 2003) with parameters α_0, d ($\alpha_0 > -d$, $0 \leq d < 1$): $\tau_1(j) \propto \prod_{j'=1}^{j} \{\alpha_0 + d\,(j'-1)\}$ and $\tau_2(j) \propto \Gamma(j-d)$, where $\Gamma(\cdot)$ is the Gamma function.

Finite Dirichlet mixture with parameter $\alpha_{\mathrm{Dir}} > 0$ and k_0 components and symmetric concentration $(\alpha_{\mathrm{Dir}}, \ldots, \alpha_{\mathrm{Dir}})$: $\tau_1(j) \propto \mathbf{1}\,[j \leq k_0]$ and $\tau_2(j) \propto \Gamma(j + \alpha_{\mathrm{Dir}})$.

The likelihood (1)-(2) and prior (3) define the following target posterior distribution

$$\pi(c) \coloneqq \pi(c \mid \mathbf{y}) \propto \tau(c) \prod_{b \in c} L(\mathbf{y}_b). \tag{4}$$

Since we view the observations as fixed, we drop the dependency on $\mathbf{y}$ from the notation throughout the paper. We detail in the following sections an original approach to sample from this posterior distribution.

3. Methodology

We organize the description of our method into two main parts. First, we define a generic construction for decomposing the problem of sampling from the posterior (Equation (4)) with arbitrary numbers of clusters into *split-merge* sub-problems. Second, we show how the PG methodology can be used to address these sub-problems.

3.1 Decomposing the clustering problem into split-merge subproblems

Algorithm 1 allows us to break down the problem of sampling from the posterior into split-merge subproblems. We refer the reader to Figure 1 for an illustrative example of the notation used throughout this description.

Algorithm 1

function SplitMerge($c, h(s), K_{c,s}(\overline{c}'|\overline{c})$)
  $s \sim h(\cdot)$ ▷ $s = \{i_1, i_2\}$
  $\overline{c} \leftarrow \{b \in c : b \cap \overline{s} \neq \emptyset\}$ ▷ Clustering restricted to the anchors
  $\overline{s} \leftarrow \bigcup_{b \in \overline{c}} b$ ▷ Closure of the anchors with respect to the clustering $\overline{c}$
  $\overline{c}' \sim K_{c,s}(\cdot|\overline{c})$
  $c' \leftarrow \overline{c}' \cup (c \backslash \overline{c})$
  return c'
end function

The algorithm requires three inputs:

1. c: the current clustering,
2. $h(s)$: a distribution for proposing an unordered pair of *anchors* $s = \{i_1, i_2\} \subset [T]$,
3. $K_{c,s}(\overline{c}'|\overline{c})$: a Markov transition kernel over the space of partitions of $\overline{s}$. This kernel is assumed to be invariant with respect to the following target distribution:

$$\overline{\pi}_{c,s}(\overline{c}') \propto \overline{\tau}_1(|\overline{c}'|) \left(\prod_{b \in \overline{c}'} \tau_2(|b|) L(\mathbf{y}_b) \mathbf{1}\left[b \cap s \neq \emptyset\right] \right), \qquad (5)$$
$$\overline{\tau}_1(j) \coloneqq \tau_1(j + |c| - |\overline{c}|).$$

In the following, we drop the subscripts from the kernel K and target $\overline{\pi}$ for simplicity.

The distribution $\overline{\pi}$ has a form similar to the posterior distribution defined in Equation (4) with two modifications. First, $\tau_1(|c|)$ is replaced by $\overline{\tau}_1(|\overline{c}'|)$. Second, the support of the distribution is restricted so that each block in $\overline{c}'$ must contain at least one anchor point. This also implicitly enforces the constraint that $|\overline{c}'| \leq |s| = 2$.

Algorithm 1 returns an updated clustering where only the allocation of points in $\overline{s}$ have changed; i.e. the updated clustering c' only potentially differs from c at points which were

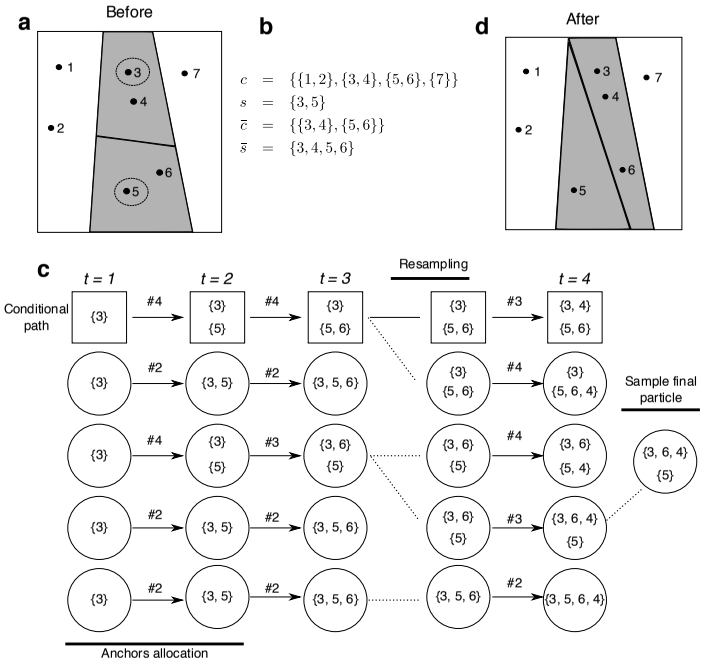


Figure 1: (**a**) Example illustrating the setup of a split merge and (**b**) the notation used in Section 3.1. The seven points denote the observation indices. While we show points in a two-dimensional space for visualization, the methodology does not rely on any topological properties of the observation space. The dashed circles denote the anchors s, and the shaded region, the closure of the anchors. (**c**) Example of a single PGSM step with $\boldsymbol{\sigma} = (3, 5, 6, 4)$. Square boxes represent particles corresponding to the fixed conditional path. Circles represent regular particles. Each particle has an associated clustering denoted by the set or pair of sets written inside each particle. Arrows indicate proposal draws labelled by the proposed state (see Figure 2). Dashed lines indicate when resampling occurs. (**d**) Configuration after the PGSM update illustrated in **c**.

initially clustered with the anchor points. The anchor proposal distribution h obviously impacts the performance of this procedure. We empirically compare the performance of three anchor proposal distributions h in Section 4.3.

This scheme has the following property:

Proposition 1 *If $c \sim \pi$, where π is given by Equation (4), then the output of Algorithm 1, c', satisfies $c' \sim \pi$. That is, the Markov kernel $K(c'|c)$ induced by Algorithm 1 is π-invariant.*

3.2 Overview of the particle Gibbs algorithm

Ideally, we would like to sample independently from $\overline{\pi}$ in Algorithm 1, that is we would like to have $K(\overline{c}'|\overline{c}) = \overline{\pi}(\overline{c}')$, but this is too computationally expensive if $|\overline{s}|$ is large. Our primary contribution is an original way to address this issue using SMC-based methods.

In principle, it would be possible to use SMC methods to obtain a sample approximately distributed according to $\overline{\pi}$ (Fearnhead, 2004). However, if we were to use this sample within Algorithm 1, the resulting invariant distribution would not be π. For this reason, we consider Particle MCMC (PMCMC) methods (Andrieu et al., 2009, 2010).

PMCMC methods allow us to use SMC ideas in a principled way within MCMC schemes. We will focus here on the PG sampler and show how one can use this methodology to obtain an efficient MCMC kernel K targeting the distribution $\overline{\pi}$ given in Equation (5). The outcome of the PG sampling steps will be either to cluster all the points in $\overline{s}$ into one block or to break $\overline{s}$ into two clusters, with the restriction that each of the two blocks should contain one anchor. Interestingly, the form of the PG algorithm is the same no matter if the two anchors were previously together or apart before its execution. This contrasts with previous split-merge algorithms such as Jain and Neal (2004), which require a different treatment for split and merge moves.

To sample from $\overline{\pi}$, PG breaks the sampling of $\overline{c}'$ into a sequence of $n := |\overline{s}|$ simpler sampling problems. In this scenario, contrary to most applications of PG, there is no intrinsic time ordering of the observations. We randomize the order in which the points are included by introducing, conditionally on s and $\overline{s}$, a random permutation $\boldsymbol{\sigma} := (\sigma_1, \ldots, \sigma_n)$. This permutation is sampled using Algorithm 2.

Algorithm 2

1: **function** SamplePermutation$(s, \overline{s})$
2: $\quad\sigma_1 \sim \text{Uniform}(s)$
3: $\quad\sigma_2 \leftarrow s \backslash \{\sigma_1\}$
4: $\quad(\sigma_3, \sigma_4, \ldots, \sigma_n) \leftarrow \text{UniformPermutation}(\overline{s} \backslash s)$
5: $\quad$**return** $\boldsymbol{\sigma}$
6: **end function**

In other words, $\boldsymbol{\sigma}$ is uniform over the permutations of the observation indices in $\overline{s}$ such that the members of s appear in the first two entries. The variable σ_t specifies the index of the observation y_{σ_t} introduced into the PG algorithm at SMC iteration ("algorithmic" time) t and x_t is the corresponding allocation decision. A particle $\mathbf{x}_t$ is defined as a sequence of allocation decisions, $\mathbf{x}_t := (x_1, \ldots, x_t)$, where $x_t \in \mathcal{X}$, $t \in \{1, \ldots, n\}$.

Given $\boldsymbol{\sigma}$, we denote the SMC proposals used within PG by $q_t^{\boldsymbol{\sigma}}(x_t|\mathbf{x}_{t-1})$, and the intermediate unnormalized target distributions, by $\gamma_t^{\boldsymbol{\sigma}}(\mathbf{x}_t)$. We remind the reader that both $\gamma_t^{\boldsymbol{\sigma}}$ and $q_t^{\boldsymbol{\sigma}}$ are allowed to depend on arbitrary subsets of the observations $\mathbf{y}$; see, e.g., (Del Moral et al., 2006). However, we omit this dependency for notational simplicity. Our methodology is flexible with respect to the choice of the proposals and the choice of the intermediate

unnormalized target distributions. For our methodology to provide consistent estimates, only the following weak assumptions have to be satisfied.

Assumption 1 *For all $t \in \{1, \ldots, n\}$, we assume* $\text{support}(\gamma_t^{\boldsymbol{\sigma}}) \subseteq \text{support}(q_t^{\boldsymbol{\sigma}})$ *where* $q_t^{\boldsymbol{\sigma}}(\mathbf{x}_t) := q_1^{\boldsymbol{\sigma}}(x_1) \prod_{k=2}^{t} q_k^{\boldsymbol{\sigma}}(x_k|\mathbf{x}_{k-1})$ *for $t \geq 2$.*

Assumption 2 *We assume that there exists a bijection $\phi^{\boldsymbol{\sigma}}$ taking a particle as input, and outputting a clustering of $\overline{s}$. More precisely, $\phi^{\boldsymbol{\sigma}}$ is a bijection between the support of the proposal, and the support of the split-merge target distribution, $\phi^{\boldsymbol{\sigma}} : \text{support}(q_n^{\boldsymbol{\sigma}}) \to \text{support}(\overline{\pi})$.*

Assumption 3 *We assume that $\gamma_n^{\boldsymbol{\sigma}}(\mathbf{x}_n) \propto \overline{\pi}(\phi^{\boldsymbol{\sigma}}(\mathbf{x}_n))$.*

Assumption 1 ensures that all the importance weights appearing in the SMC method are well-defined. Assumption 2 is a simple condition ensuring that we can consistently relabel the particles. Assumption 3 ensures that we target the desired distribution at algorithmic time n. Note that Assumption 3 only restricts the choice of γ_t for the final SMC iteration, $t = n$. We use this flexibility in Section 3.4. We show in the next section how to design $q^{\boldsymbol{\sigma}}$, $\gamma^{\boldsymbol{\sigma}}$, $\phi^{\boldsymbol{\sigma}}$ and $\mathcal{X}$ that satisfy these assumptions.

PG proceeds in a way similar to standard SMC algorithms, with the important difference that one of the N particle paths is fixed. In our setup, this path is obtained using the inverse of the bijection described in Assumption 2, applied to the state of the restricted clustering $\overline{c}$ prior to the current PG step. As discussed in Chopin and Singh (2015), we can without loss of generality set the genealogy of the conditioning path $\overline{c}$ to $(1, ..., 1)$, i.e. we use the particle index $p = 1$ for this conditioning path: $\mathbf{x}_n^1 := (\phi^{\boldsymbol{\sigma}})^{-1}(\overline{c})$. This defines a path by taking a prefix of length t of the vector $\mathbf{x}_n^1$ for $\mathbf{x}_t^1$, i.e. $\mathbf{x}_t^1 = \left(\mathbf{x}_n^1\right)_{1:t}$.

The final ingredient required to describe the PG algorithm is a *conditional resampling* distribution $r(\boldsymbol{a} \mid \boldsymbol{w})$, where $\boldsymbol{a} := (a_2, \ldots, a_N)$ denotes the resampling ancestors, $a_p \in \{1, \ldots, N\}$, and $\boldsymbol{w} := (w^1, \ldots, w^N)$ denotes a vector of probabilities. We limit ourselves to multinomial resampling:

$$r(\boldsymbol{a} \mid \boldsymbol{w}) = \prod_{p=2}^{N} r(a_p \mid \boldsymbol{w}) = \prod_{p=2}^{N} w^{a_p}. \tag{6}$$

More elaborate schemes can be used, see Andrieu et al. (2009, 2010). Instead of resampling at each time step as in vanilla SMC algorithms, we only resample when the relative Effective Sampling Size (ESS) criterion, which takes values between 0 and 1, is below a pre-specified threshold β, $\beta \in [0, 1]$. The adaptive resampling procedure was proposed by Liu and Chen (1995) for standard particle methods and the correctness of this procedure for PG has been established in Lee (2011). The resulting procedure is described in Algorithm 3.

Most of Algorithm 3 is concerned with the creation of temporary auxiliary variables (lines 1–34). These auxiliary variables can all be discarded after the algorithm returns $\overline{c}'$, as they can be resampled from scratch every time Algorithm 3 is run. The part of the algorithm that performs the actual split or merge is in lines 35 and 36. At this point of the execution of the algorithm, the particle population at SMC generation n can be interpreted (via $\phi^{\boldsymbol{\sigma}}$) as a distribution over clusterings of $\overline{s}$, with some particles corresponding to merging

Algorithm 3

function ParticleGibbsSplitMerge($s, \overline{s}, \overline{c}, \overline{\pi}$) ▷ Inputs coming from Algorithm 1
  $\boldsymbol{\sigma} \leftarrow$ SamplePermutation$(s, \overline{s})$
  $\mathbf{x}_n^1 \leftarrow (\phi^{\boldsymbol{\sigma}})^{-1}(\overline{c})$ ▷ Compute the conditional path
  for $t \in \{1, \ldots, n-1\}$ **do**
    $\mathbf{x}_t^1 \leftarrow (\mathbf{x}_n^1)_{1:t}$ ▷ First particle of each generation matches the conditional path
  end for
  for $p \in \{2, \ldots, N\}$ **do** ▷ Initialize particles
    $x_1^p \sim q_1^{\boldsymbol{\sigma}}(\cdot)$
    $\mathbf{x}_1^p \leftarrow (x_1^p)$
  end for
  for $p \in \{1, \ldots, N\}$ **do** ▷ Initialize incremental importance weights
    $\tilde{w}_1^p \leftarrow \frac{\gamma_1^{\boldsymbol{\sigma}}(\mathbf{x}_1^p)}{q_1^{\boldsymbol{\sigma}}(\mathbf{x}_1^p)}$
  end for
  for $p \in \{1, \ldots, N\}$ **do**
    $w_1^p \leftarrow \frac{\tilde{w}_1^p}{(\sum_{p'=1}^N \tilde{w}_1^{p'})}$ ▷ Compute normalized weights
  end for
  for $t \in \{2, \ldots, n\}$ **do**
    if $(N \sum_{p=1}^N (w_{t-1}^p)^2)^{-1} < \beta$ **then** ▷ Resample only if relative ESS is too low
      $\boldsymbol{a} \sim r(\cdot \mid \boldsymbol{w}_{t-1})$ ▷ Perform the conditional resampling step
      $\tilde{\boldsymbol{w}}_{t-1} \leftarrow (1, 1, 1, \ldots, 1)$ ▷ Reset the weights
    else
      $\boldsymbol{a} \leftarrow (1, 2, 3, \ldots, N)$ ▷ Resampling is skipped: set $\boldsymbol{a}$ to the identity map
    end if
    for $p \in \{2, \ldots, N\}$ **do**
      $x_t^p \sim q_t^{\boldsymbol{\sigma}}(\cdot \mid \mathbf{x}_{t-1}^{a_p})$ ▷ Propose new block allocation for y_{σ_t}
      $\mathbf{x}_t^p \leftarrow (\mathbf{x}_{t-1}^{a_p}, x_t^p)$ ▷ Concatenate new block allocation to path
    end for
    for $p \in \{1, \ldots, N\}$ **do**
      $\tilde{w}_t^p \leftarrow \tilde{w}_{t-1}^p \cdot w(\mathbf{x}_{t-1}^{a_p}, x_t^p)$ ▷ Update weights (see Equation (9))
    end for
    for $p \in \{1, \ldots, N\}$ **do**
      $w_t^p \leftarrow \frac{\tilde{w}_t^p}{(\sum_{p'=1}^N \tilde{w}_t^{p'})}$ ▷ Compute normalized weights
    end for
  end for
  $\mathbf{x}'_n \sim \sum_{p=1}^N w_n^p \delta_{\mathbf{x}_n^p}(\cdot)$ ▷ Sample particle representing new state
  $\overline{c}' \leftarrow \phi^{\boldsymbol{\sigma}}(\mathbf{x}'_n)$ ▷ Compute updated partition
  return $\overline{c}'$
end function

all points in $\overline{s}$ into one block (i.e. when $|\phi^{\boldsymbol{\sigma}}(\mathbf{x}'_n)| = 1$), and others, to various way of splitting $\overline{s}$ into two blocks (when $|\phi^{\boldsymbol{\sigma}}(\mathbf{x}'_n)| = 2$).

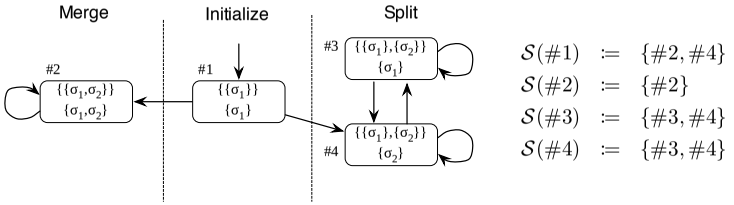


Figure 2: Left: State space $\mathcal{X}$ and allowed transitions $\mathcal{S}(\cdot)$ for the local allocation decisions. Right: allowed transitions between the states.

Correctness of this procedure follows straightforwardly from the original PG argument (see Appendix B for details):

Proposition 2 *Under Assumptions 1, 2, and 3, the output of Algorithm 3, $\overline{c}'$, satisfies $\overline{c}' \sim \overline{\pi}$ if $\overline{c} \sim \overline{\pi}$, for any $N \geq 2$, i.e. the Markov kernel $\overline{K}(\overline{c}'|\overline{c})$ induced by Algorithm 3 is $\overline{\pi}$-invariant.*

3.3 Intermediate target distributions and proposals construction

We detail here the construction of a set of proposal distributions $q^{\boldsymbol{\sigma}}$, unnormalized target distributions $\gamma^{\boldsymbol{\sigma}}$, and mappings $\phi^{\boldsymbol{\sigma}}$ satisfying Assumptions 1 to 3. We denote the space of possible allocation decisions at a given PG iteration by $\mathcal{X}$. Our construction is based on an encoding where the space $\mathcal{X}$ consists in the rectangles shown in Figure 2. We call the rectangles *states* for short. These states are used to build particles: recall that a particle $\mathbf{x}_t$ is defined as a list of local decisions, $\mathbf{x}_t \coloneqq (x_1, \ldots, x_t)$, $x_{t'} \in \mathcal{X}$.

The state appended to a particle at time t represents (a) the clustering restricted to the anchors (shown in the first line of each rectangle in Figure 2), and (b), the cluster joined by y_{σ_t} (encoded by the anchor(s) contained in the joined cluster, second line in the same figure). As shown in Figure 2, the "merge state" (left) is an absorbing state, encoding the fact that following this local decision, all children particles are forced to join the unique block in the restricted clustering. The two "split states" (right), on the other hand, both have two outgoing transitions, encoding the fact that for each index in $\overline{s} \backslash s$, the corresponding observation needs to be allocated to one of the two blocks.

There is a bijection between the support of $\overline{\pi}$, and particles respecting the transition constraints defined by the arrows in Figure 2. More precisely, for each state $x \in \mathcal{X}$, we let $\mathcal{S}(x)$ denote the set of allowed transitions from x. We write $\mathbf{x}_t \in \mathcal{S}_t$ if (a) $x_1 = \#1$, and (b) for all $t' \in \{2, \ldots, t\}$, $x_{t'} \in \mathcal{S}(x_{t'-1})$. From this definition, we obtain the following result whose proof is given in Appendix C.

Proposition 3 *For any permutation $\boldsymbol{\sigma}$ satisfying $\{\sigma_1, \sigma_2\} = s$, there is a bijective map $\phi^{\boldsymbol{\sigma}}$ from the space of particles respecting the transition constraints, $\mathcal{S}_n$, to the support of the restricted target,* $\text{support}(\overline{\pi})$.

We use this bijection to define a sequence of intermediate target and proposal distributions. The intermediate target at time t of support $\mathcal{S}_t$ is given by:

$$\gamma_t^{\boldsymbol{\sigma}}(\mathbf{x}_t) \coloneqq \overline{\tau}_1(\overline{c}_t) \left(\prod_{b \in \overline{c}_t} \tau_2(|b|) L(\mathbf{y}_b) \right), \tag{7}$$

where $\overline{c}_t = \phi^{\boldsymbol{\sigma}_{1:t}}(\mathbf{x}_t)$. By construction, we have that for $t = n$, $\gamma_n^{\boldsymbol{\sigma}}(\mathbf{x}_n) \propto \overline{\pi}(\phi^{\boldsymbol{\sigma}}(\mathbf{x}_n))$ so Assumption 3 is satisfied.

We define as proposals:

$$\begin{aligned} q_1^{\boldsymbol{\sigma}}(x_1) &\coloneqq \delta_{\#1}(x_1), \\ q_t^{\boldsymbol{\sigma}}(x_t \mid \mathbf{x}_{t-1}) &\coloneqq \frac{\gamma_t^{\boldsymbol{\sigma}}(\mathbf{x}_t)}{\sum_{x_t' \in \mathcal{S}(x_{t-1})} \gamma_t^{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}, x_t')}, \end{aligned} \tag{8}$$

where $(\mathbf{x}_{t-1}, x_t')$ denotes the concatenation of x_t' to the vector $\mathbf{x}_{t-1}$, and $\mathbf{x}_t = (\mathbf{x}_{t-1}, x_t)$. These definitions satisfy Assumption 1, and yield the following weight updates:

$$\begin{aligned} w_t(\mathbf{x}_{t-1}, x_t) &\coloneqq \frac{\gamma_t^{\boldsymbol{\sigma}}(\mathbf{x}_t)}{\gamma_{t-1}^{\boldsymbol{\sigma}}(\mathbf{x}_{t-1})} \frac{1}{q_t^{\boldsymbol{\sigma}}(x_t \mid \mathbf{x}_{t-1})} \\ &= \frac{\sum_{x_t' \in \mathcal{S}(x_{t-1})} \gamma_t^{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}, x_t')}{\gamma_{t-1}^{\boldsymbol{\sigma}}(\mathbf{x}_{t-1})} \\ &= \sum_{x_t' \in \mathcal{S}(x_{t-1})} \frac{\gamma_t^{\boldsymbol{\sigma}}(\mathbf{x}_{t-1}, x_t')}{\gamma_{t-1}^{\boldsymbol{\sigma}}(\mathbf{x}_{t-1})}. \end{aligned} \tag{9}$$

If $t > |s|$ then Equation (10) simplifies as follows

$$\frac{\gamma_t^{\boldsymbol{\sigma}}(\mathbf{x}_t)}{\gamma_{t-1}^{\boldsymbol{\sigma}}(\mathbf{x}_{t-1})} = \frac{\tau_2(|b_t^+|)}{\tau_2(|b_t^-|)} L\left(\mathbf{y}_{b_t^+} \mid \mathbf{y}_{b_t^-}\right), \tag{10}$$

where

$$L\left(\mathbf{y}_{b_t^+} \mid \mathbf{y}_{b_t^-}\right) \coloneqq \frac{L\left(\mathbf{y}_{b_t^+}\right)}{L\left(\mathbf{y}_{b_t^-}\right)}. \tag{11}$$

Here b_t^- and b_t^+ encode the block in which a point is added to when transitioning from $\mathbf{x}_{t-1}$ to $\mathbf{x}_t$, the first being the block before the addition, and the second, the same block after the addition:

$$b_t^- \coloneqq \overline{c}_{t-1} \backslash \overline{c}_t, \;\; b_t^+ \coloneqq b^- \cup \{\sigma_t\}. \tag{12}$$

Depending on the form of the partition prior and likelihood it may be possible to simplify these quantities into more computationally efficient forms.

3.4 An improved sequence of intermediate target distributions

We now describe an improvement over the basic intermediate and proposal distributions presented in the previous section. This improvement addresses a "greediness" problem of the (conditional) SMC procedure. Consider a case where the ratio $\overline{\tau}_1(1)/\overline{\tau}_1(2)$ between a merge and a split is large. This can occur for example when the Dirichlet process concentration parameter α_0 is small. In this case, the proposal in the first non-trivial step, $q_2^{\boldsymbol{\sigma}}$, will assign most of its mass to the transition from state #1 to state #2 (see Figure 2). However, the likelihood might overcome this prior when $|\overline{s}|$ is large. But proposing such split has low probability under the definitions given in the previous section, as #2 is an absorbing state.

To overcome this issue, we build a new sequence of intermediate distributions, which delay the incorporation of the prior:

$$\widehat{\gamma_t^{\boldsymbol{\sigma}}}(\mathbf{x}_t) \coloneqq \begin{cases} \mathbf{1}[\mathbf{x}_t \in \mathcal{S}_t], & \text{if } t \in \{1, 2\}, \\ (\gamma_2^{\boldsymbol{\sigma}}(\mathbf{x}_{1:2}))^{\zeta_t} \frac{\gamma_t^{\boldsymbol{\sigma}}(\mathbf{x}_t)}{\gamma_2^{\boldsymbol{\sigma}}(\mathbf{x}_{1:2})}, & \text{otherwise.} \end{cases} \tag{13}$$

where ζ_t is a positive increasing annealing schedule such that $\zeta_n = 1$. We use the following proposal based on these new intermediate distributions:

$$\begin{aligned} \widehat{q_1^{\boldsymbol{\sigma}}}(x_1) &\coloneqq \delta_{\#1}(x_1), \\ \widehat{q_t^{\boldsymbol{\sigma}}}(x_t \mid \mathbf{x}_{t-1}) &\coloneqq \frac{\widehat{\gamma_t^{\boldsymbol{\sigma}}}(\mathbf{x}_t)}{\sum_{x_t' \in \mathcal{S}(x_{t-1})} \widehat{\gamma_t^{\boldsymbol{\sigma}}}(\mathbf{x}_{t-1}, x_{t-1}')}. \end{aligned} \tag{14}$$

This yields the weight updates:

$$\widehat{w_t}(\mathbf{x}_{t-1}, x_t) \coloneqq \sum_{x_t' \in \mathcal{S}(x_{t-1})} \frac{\widehat{\gamma_t^{\boldsymbol{\sigma}}}(\mathbf{x}_{t-1}, x_t')}{\widehat{\gamma_{t-1}^{\boldsymbol{\sigma}}}(\mathbf{x}_{t-1})}. \tag{15}$$

For simplicity, we pick $\zeta_t = \frac{t-2}{n-2}$. This choice simplifies ratios of intermediate distributions to:

$$\frac{\widehat{\gamma_t^{\boldsymbol{\sigma}}}(\mathbf{x}_t)}{\widehat{\gamma_{t-1}^{\boldsymbol{\sigma}}}(\mathbf{x}_{t-1})} = \begin{cases} 1 & \text{if } t = 2, \\ (\gamma_2^{\boldsymbol{\sigma}}(\mathbf{x}_{1:2}))^{\Delta\zeta} \frac{\gamma_t^{\boldsymbol{\sigma}}(\mathbf{x}_t)}{\gamma_{t-1}^{\boldsymbol{\sigma}}(\mathbf{x}_{t-1})} & \text{if } t > 2, \end{cases} \tag{16}$$

where $\Delta\zeta \coloneqq (n-2)^{-1}$.

3.5 Runtime analysis

To simplify the analysis of the running time, we make a few assumptions.

Assumption 4 *The parametric likelihood model has the following properties:*

1. *let $\psi \coloneqq \psi(\mathbf{y}_b)$ denote a sufficient statistic, and define, with a slight abuse of notation, $L(\psi) \coloneqq L(\mathbf{y}_b)$. For a given sufficient statistic value ψ, the likelihood $L(\psi)$ can be computed in time $O(l)$,*
2. *the sufficient statistic for b^+, $\psi^+ \coloneqq \psi(\mathbf{y}_{b^+})$, can be updated in time $O(u)$ from the sufficient statistic for b^-, $\psi^- \coloneqq \psi(\mathbf{y}_{b^-})$.*

The next assumption holds for all the clustering priors reviewed in Section 2.

Assumption 5 *The ratio* $\frac{\tau_2(j+1)}{\tau_2(j)}$ *can be computed in constant time.*

For example, with a Dirichlet process, this ratio is equal to $j!/(j-1)! = j$. Since $|b^+| = |b^-| + 1$, Assumption 5 implies that the ratio $\frac{\tau_2(|b^+|)}{\tau_2(|b^-|)}$ in Equation (10) can be computed in constant time.

Proposition 4 *Under Assumptions 4 and 5, one weight computation, Equation (15), takes time* $O(u+l)$. *The storage cost per particle is* $O(1)$. *Moreover, the running time per weight computation is independent of the number of clusters.*

The running time result follows directly from the fact that $|\mathcal{S}(\cdot)| \leq 2$, and hence the sum in Equation (15) has a constant number of terms. The constant storage cost follows from the finite dimensionality of the sufficient statistics (see Assumption 4), and from $|\mathcal{X}| = 4$.

We also remind the reader that for most resampling schemes, including the one in Equation (6), the computational cost as a function of the number of particles and SMC iterations is $O(Nn) = O(N|\overline{s}|)$ (Doucet and Johansen, 2011).

3.6 Generalization

For simplicity, we have assumed so far that $|s| = 2$, and hence, according to the auxiliary variable analysis of Appendix B, $|\overline{c}| \leq 2$. In fact, the same auxiliary variables with more than two anchor points can be used to construct novel sampling algorithms. Details are given in Appendix D.

This generalization loses some interpretability compared to the split-merge case ($|s| = 2$), but can be useful in finite clustering models. In this case, it may only be possible to split a cluster if a merge is performed simultaneously. For this reason, we use $|s| = 3$ in the finite Dirichlet mixture model examples in Section 4.3. For the Dirichlet Process, we did not observe notable improvements by going from $|s| = 2$ to $|s| = 3$, so we use the former setting for the non-parametric models.

4. Applications

In this section, we demonstrate the performance of our methodology and compare it to standard alternatives. We use a series of synthetic datasets covering a large spectrum of cluster separateness, as well as real data coming from a geolocation application.

4.1 Implementation and evaluation

We have implemented the following three Dirichlet Process (DP) clustering samplers in the same Python codebase: the PGSM method described in this work, the efficient Sequentially-Allocated Merge Split (SAMS) method of Dahl (2003), as well as the standard Gibbs sampler. The code and instructions allowing to reproduce the experiments is available at `https://github.com/aroth85/pgsm`. The implementation of the likelihood computations are the same for all samplers, thus the running times are comparable. We have tested

the correctness of our computer implementations by computing the true posterior distribution on small examples via combinatorial enumeration, and verified that the Monte Carlo estimates converged to this distribution for all three methods.

Unless we state otherwise, we initialized the samplers with the single-cluster configuration. In datasets much smaller than those studied in this work, initializing the Gibbs sampler to the fully disconnected clustering is advantageous (Sudderth, 2006), but in larger datasets, the quadratic burn-in cost involved with this initialization is not scalable. However, we verified that after a long burn-in period the Gibbs method initialized to the fully disconnected clustering eventually reaches the same likelihood values in the synthetic examples. We also investigate the high cost of the fully disconnected initialization in the results shown in Figure 9.

To evaluate the performance of the samplers, we held-out a random but fixed 10% of each dataset. We collected samples and computed the predictive likelihood and V-measure (Rosenberg and Hirschberg, 2007) every 100 iterations. All experiments are replicated 10 times, and smoothed using a moving average with a window size of 20 for plotting.

4.2 Likelihoods and priors

In six of the synthetic experiments and the geolocation experiments discussed further in Section 4.3 and Section 4.4, we used a Normal-Inverse-Wishart conjugate likelihood model. In two of the synthetic experiments in Section 4.3 we used Bernoulli mixture models with 50 dimensions. Each dimension is an independent draw from a Bernoulli random variable with cluster specific parameters. We set a proportion of the dimensions to be uninformative as follows. Values for uninformative dimensions were drawn from Bernoulli variables with parameter 0.5 regardless of the cluster membership. Values for the remaining dimensions were drawn from cluster specific Bernoulli variables with parameters sampled from the Uniform distribution. For the cancer data discussed in Section 4.5, we use the application-specific PyClone likelihood model (Roth et al., 2014). The PyClone model uses genomic sequence data from tumours to identify mutations which co-occur in cells and estimates the proportion of cells harbouring the mutations. The model is not conjugate, so we apply a discretization that allows us to treat the model as conjugate. Complete details for each model are provided in Appendix E.

We use a DP prior with base measure given by the conjugate prior of the corresponding likelihood model in all experiments. We use a Gamma$(1, 0.1)$ prior and resample the value of the concentration parameters α_0 using a standard auxiliary variable method (Escobar and West, 1995). The value of α_0 is initialized to 1.0.

4.3 Artificial datasets

We used four sources of synthetic data. First, the four datasets from Franti and Virmajoki (2006) denoted S1–4. Each of the four datasets consists in 5000 points generated from 15 bivariate Normal distributions with increasing amount of overlap between the clusters. Second, we created another synthetic dataset, which we call C1, shown in Figure 5, right. Third, we simulated two datasets with 5000 points from a Bernoulli mixture model with 50 dimensions and 16 clusters, where we set 25% (Ber-0.25) and 50% (Ber-0.5) of dimensions

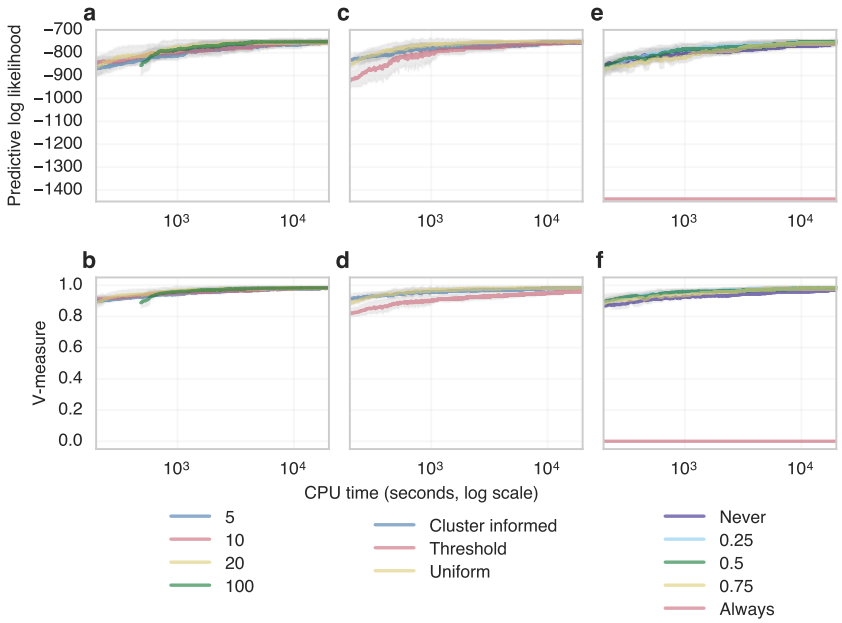


Figure 3: Effect on the predictive performance and clustering accuracy as a function of CPU time in log scale. **a**) and **b**) Varying the number of particles with the cluster informed proposal and a relative ESS resampling threshold of 0.5. **c**) and **d**) Varying the distribution for proposing pairs of anchor points, h, with 20 particles and an ESS resampling threshold of 0.5. **e**) and **f**) Varying the relative ESS resampling threshold with the cluster informed anchor proposal and 20 particles.

to be uninformative. Finally, we used 64 (DIM064) and 128 (DIM128) dimensional Normal datasets from Fränti et al. (2006) with 1024 data points and 16 clusters.

We started with a series of pilot experiments on S1 only, designed to assess the effect of various tuning parameters on the performance of PGSM. For all pilot experiments we use the unmixed PGSM sampler to isolate the effect of each tuning parameter. In practice it is usually better to alternate between one iteration of the PGSM sampler and one iteration of the Gibbs sampler. The effect of this mixing is explored later in this section.

We first explore the performance as we vary the number of particles used for each PGSM iteration (Figures 3 **a** and **b**). The curves with more particles take more time per iteration to run, however seem to achieve slightly better V-measure and predictive likelihood after

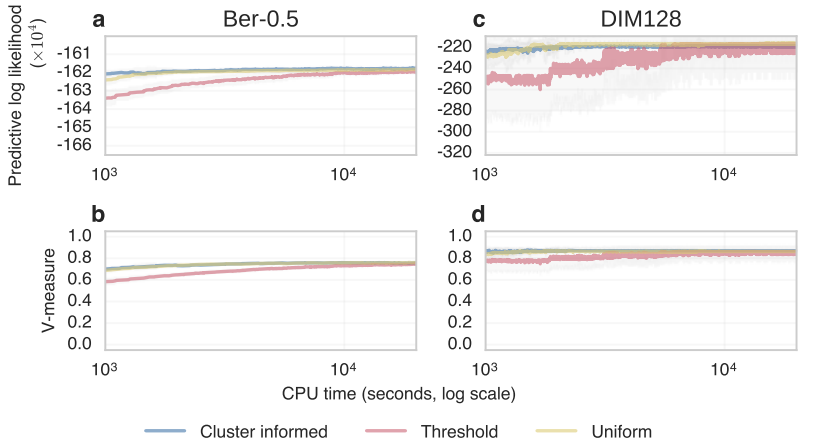


Figure 4: Effect on the predictive performance and clustering accuracy as a function of CPU time in log scale with different distribution h for proposing pairs of anchor points. **a**) and **b**) Comparison using a 50 dimensional Bernoulli dataset with 50% of the dimensions being uninformative. **c**) and **d**) Comparison using a 128 dimensional Normal dataset.

the initial iterations. The performance difference are negligible and the PGSM sampler generally seems insensitive to the number of particles for this dataset. For subsequent experiments we used 20 particles.

Next we compare the performance of different proposal distributions for the anchor auxiliary variables (Figures 3 **c** and **d**). For this experiment we kept the number of particles fixed at 20 and the resampling threshold at 0.5. We consider three proposal distributions.

Uniform: Sample the anchors uniformly at random from the $\binom{T}{2}$ possibilities.

Cluster informed: Sample the first anchor uniformly at random. Sample a cluster to draw the second anchor from with probability $\frac{1}{|c-1|}$ for the cluster containing the first anchor; otherwise proportional to

$$\frac{L(y_{\bar{b}\cup b})}{L(y_{\bar{b}})L(y_b)}$$

where $\bar{b}$ is the cluster containing the first anchor and b is the candidate cluster. Sample the second anchor uniformly from the chosen cluster.

Threshold informed: Sample the first anchor uniformly at random and the second anchor from clusters that have Chinese restaurant attachment probabilities greater than a threshold of 0.01.

To ensure the adaptation of the informed proposals stops, and does not perturb the invariant distribution of the sampler, we only update the proposal distributions when the number of clusters instantiated breaks the previous record. Adaptation is guaranteed to terminate in finite time given there are only a finite number of points. This would usually take a long time, so in practice it is advantageous to stop adaptation after a fixed period of time. Detailed implementations of the cluster informed and threshold informed proposals are given in Appendix F. Our results suggest that performance is not strongly affected by the anchor proposal distribution h. We only saw a small advantage when using the informed proposal distributions for the auxiliary anchor variables in s. We also explored the effect of h in higher dimensional datasets (Figure 4). Again we found the results are not sensitive to the choice of h. With the exception of the circle dataset, where we used the uniform proposal, we used the cluster informed prior for both the PGSM and SAMS samplers in subsequent experiments.

The frequency of the resampling step had a larger effect. A critical implementation point in order for the PGSM method to work is that resampling should be done adaptively by monitoring the ESS of the particle approximations in Algorithm 3 (Liu and Chen, 1995; Lee, 2011). We varied the relative ESS resampling threshold β from 0 (never resample) to 1 (always resample) (Figures 3 **e** and **f**). We observed that the performance is markedly degraded if resampling is performed after each SMC iteration, but similar for all other resampling thresholds. We used a threshold of 0.5 in all other experiments.

Next, we used the dataset C1 to investigate the effectiveness of our method with the finite clustering model introduced in Section 2, with the number of clusters fixed to $k_0 = 5$. In this case, standard split-merge methods such as SAMS are less helpful since only merging can be performed when the maximum number of clusters has been allocated. The PGSM sampler does not have this restriction and naturally allows simultaneously splitting and merging while preserving the total number of clusters. Furthermore, the PGSM sampler can use more than two anchors, potentially allowing for large changes in configuration without altering the number of clusters. We compared the PGSM with two ($|s| = 2$) and three ($|s| = 3$) anchors to the Gibbs sampler. The PGSM method outperformed the Gibbs sampler, though increasing the number of anchors did not improve the performance (Figure 5 **a**). We plot the predictive densities (Figure 5 **b**, **d**, **f**) and cluster allocations (Figure 5 **c**, **e**, **g**) after running each sampler for 1000 seconds. At this point the PGSM sampler used a single cluster to model the points in the middle, while the Gibbs samplers used two clusters to model the central cluster.

In Figures 6 and 7 we show a series of experiments on the four datasets S1–4 describe in the previous section. We compare the PGSM to standard Gibbs and the SAMS method of Dahl (2003). We first compared pure kernels, where the split-merge samplers are not mixed with standard Gibbs moves (Figure 6). The pure PGSM kernel outperformed both Gibbs and SAMS on datasets S1 and S2. The Gibbs kernel and PGSM perform similarly for datasets S3 and S4, and both outperformed SAMS. When the split-merge moves are mixed with standard Gibbs moves, the split-merge methods outperformed Gibbs on datasets S1 and S2, with all methods showing similar performance on datasets S3 and S4 (Figure 7).

Finally, we explored the performance of the methods on four high dimensional datasets. The mixed PGSM and Gibbs samplers performed the best on the Bernoulli datasets, while the unmixed PGSM sampler is slower to reach the same predictive likelihood and V-measure

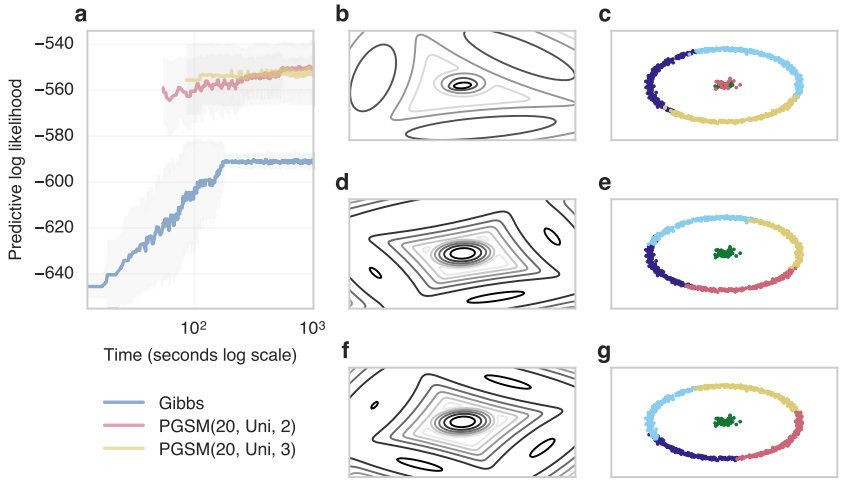


Figure 5: Comparison of Gibbs and PGSM for finite Dirichlet prior with $k_0 = 5$. **a**) Predictive log likelihood comparison of Gibbs and PGSM using two (PGSM(20, Uni, 2)) or three (PGSM(20, Uni, 3)) anchors. Predictive density after 1000 seconds for **b**) Gibbs; **d**) PGSM(20, Uni, 2); **f**) PGSM(20, Uni, 3). Cluster assignment after 1000 seconds for **c**) Gibbs; **e**) PGSM(20, Uni, 2); **g**) PGSM(20, Uni, 3).

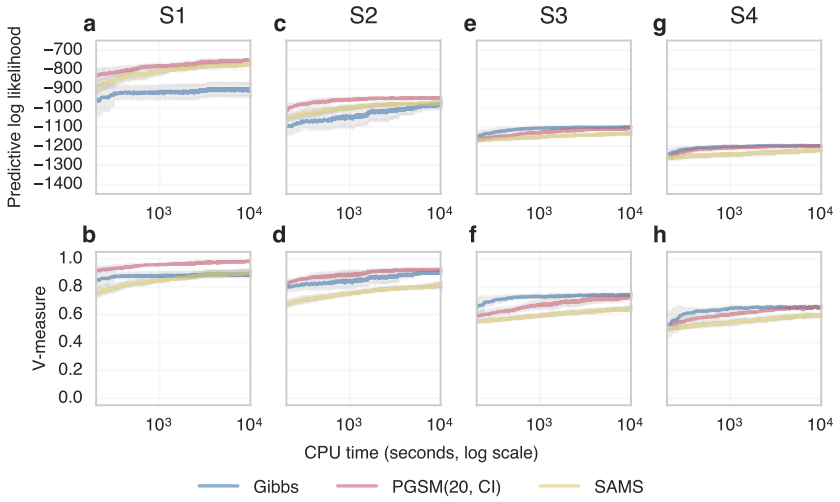


Figure 6: Comparison of MCMC algorithm using only a single kernel at a time (pure kernels) on 2D Normal datasets. Predictive log likelihood for datasets **a**) S1; **c**) S2; **e**) S3; **g**) S4. V-measure for datasets **b**) S1; **d**) S2; **f**) S3; **h**) S4.

(Figures 8 **a-d**). The mixed SAMS sampler failed to reach the same predictive likelihood as the PGSM and Gibbs methods, oscillating around lower values. The unmixed SAMS sampler appears to be trapped in a local mode, corresponding to poor predictive likelihood and V-measure. For the Normal datasets, the Gibbs sampler was trapped in a local mode and had markedly worse performance than other methods (Figures 8 **e-h**). The unmixed samplers outperformed the mixed equivalents on the 64 dimensional data. Furthermore, the unmixed PGSM method had a large performance advantage over all other methods on the 128 dimensional dataset.

4.4 Geolocation data

We compared the performance of the three sampling methods on a geolocation dataset. The dataset, described in more detail in Fränti et al. (2010), consists of a subset of data collected by MOPSI, a Finnish mobile application where users can post their current geographic location via their mobile device. The subset we used consists of a list of 13,467 locations (latitude-longitude pairs) from users located in Finland until 2012. The data is freely accessible from `http://cs.joensuu.fi/sipu/datasets/`.

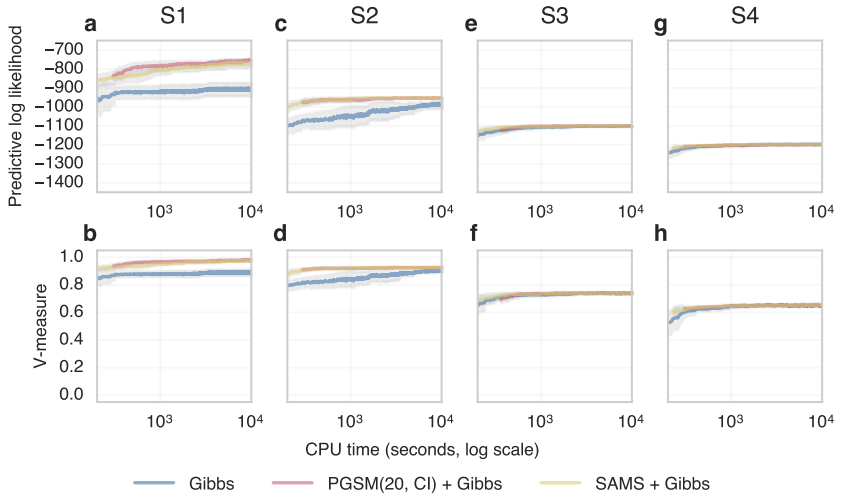


Figure 7: Comparison of MCMC algorithm using split-merge moves combined with standard Gibbs moves (mixed kernels) on 2D Normal datasets. Predictive log likelihood for datasets **a**) S1; **c**) S2; **e**) S3; **g**) S4. V-measure for datasets **b**) S1; **d**) S2; **f**) S3; **h**) S4.

We use this data as a proxy for the estimation of mobile device user density. The DP mixture of Normal-Inverse-Wishart distributions provides a natural way to obtain a parsimonious estimate of population density, where the flexibility on the shape and number of clusters can accommodate a broad range of density variability factors ranging from densely populated cities to vast low-density rural areas.

We summarize the results in Figure 9. In Figures 9 **a-c**, we display quantitative results as measured by held-out predictive likelihood performance. In Figure 9 **a**, we show that mixed PGSM, mixed SAMS and Gibbs samplers perform similarly. In Figure 9 **b**, we show that the performance of SAMS is considerably degraded if SAMS is not mixed with a Gibbs kernel. In Figure 9 **c**, we show that the performance of PGSM is less degraded if not mixed with a Gibbs kernel.

In Figures 9 **e-j**, we visualize the posterior predictive density approximated using MCMC samples. We also show the raw data in Figure 9 **d** for reference. The following three pairs of density plots are included to illustrate the high computational cost of initializing a standard Gibbs sampler at a fully disconnected configuration. From left to right: the first pair shows the predictive density after one round of Gibbs sampling initialized at the fully disconnected configuration (Figures **e** and **f**); the second pair, after sampling with PGSM

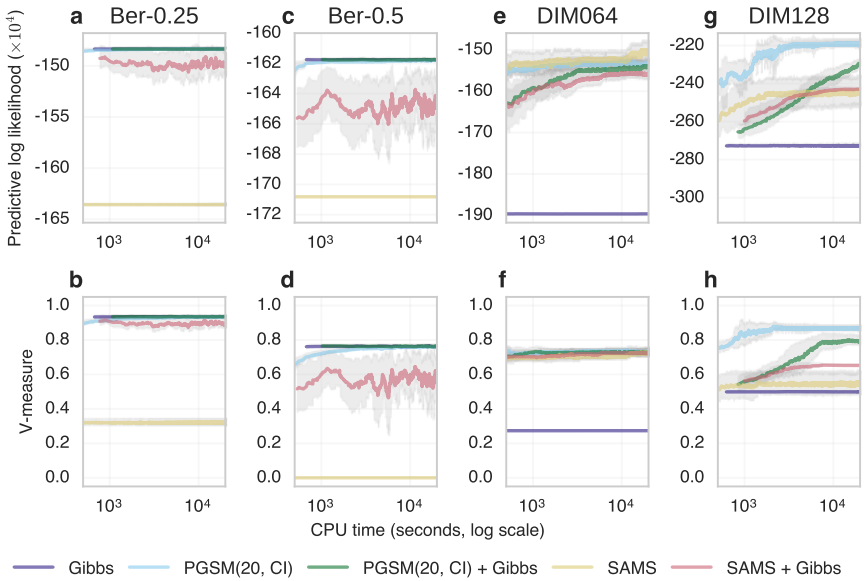


Figure 8: Comparison of MCMC algorithm using split-merge moves with combined with standard Gibbs moves (mixed kernels) on 50 dimensional Bernoulli data with 25% (Ber-0.25) and 50% (Ber-0.5) and Normal data 64 (DIM064) and 128 (DIM128) dimensions. Predictive log likelihood for datasets **a**) Ber-0.25; **c**) Ber-0.5; **e**) DIM064; **g**) DIM128. V-measure for datasets **b**) Ber-0.25; **d**) Ber-0.5; **f**) DIM064; **h**) DIM128.

initialized at the fully connected configuration for the same time (Figures **g** and **h**); the third, after running the Gibbs sampler for 10^5 seconds (Figures **i** and **j**). This demonstrates that our method can produce accurate and compact density estimates without relying on an expensive initialization phase.

4.5 Inferring population structure in heterogeneous tumours

The PyClone model (Roth et al., 2014) is designed to infer the proportion of cancer cells in a tumour sample which contain a mutation, which we refer to as the cellular prevalence of the mutation. The input data consists of a set of digital measurements of allelic abundance which is assumed to be proportional to the true abundance of the allele in the sample. The key factors which need to be deconvolved to convert this measurement to an estimate of

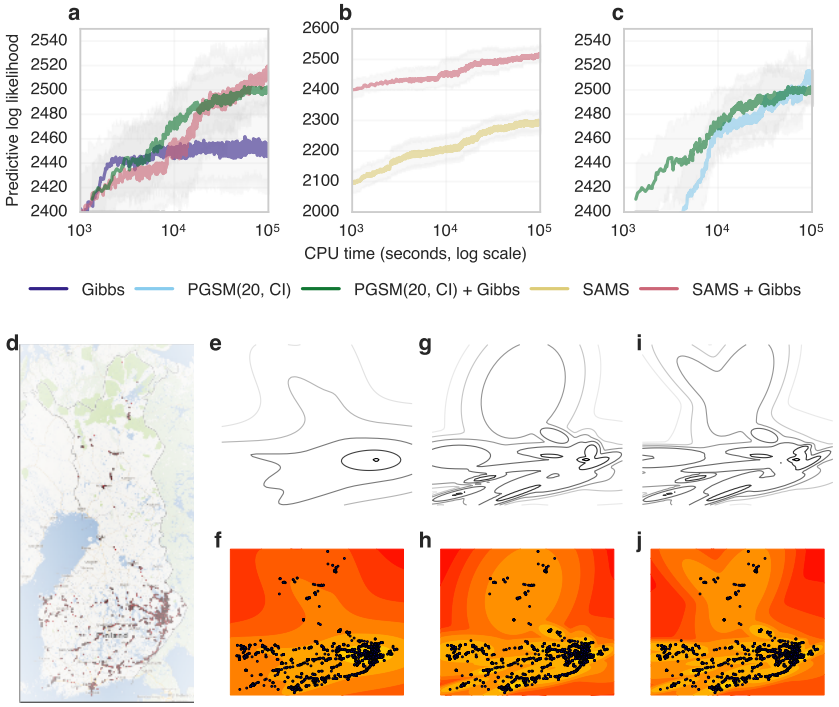


Figure 9: Geolocation dataset. Comparison of predictive likelihoods of **a**) mixed kernels; **b**) PGSM pure and mixed kernels; **c**) SAMS pure and mixed kernels. **d**) MOPSI geolocation dataset. **e**) Predictive density of Gibbs initialized from the disconnected configurations after one iteration; **d**) with point data points overlayed. **g**) and **h**) Predictive density of mixed PGSM kernel after an equivalent amount of time. **i**) and **j**) Predictive density of disconnected Gibbs after 10^5 seconds.

cellular prevalence are that some cells derive from healthy (normal) tissue and the genomes of cancer cells contain multiple copies of a locus. The model assumes that mutations will group by cellular prevalence due to the expansion of populations of genetically identical cells. The number of populations is unknown, thus the PyClone model uses a DP prior with a Uniform($[0, 1]$) base measure. The component parameters are interpreted as the cellular prevalence of the mutations associated with the component.

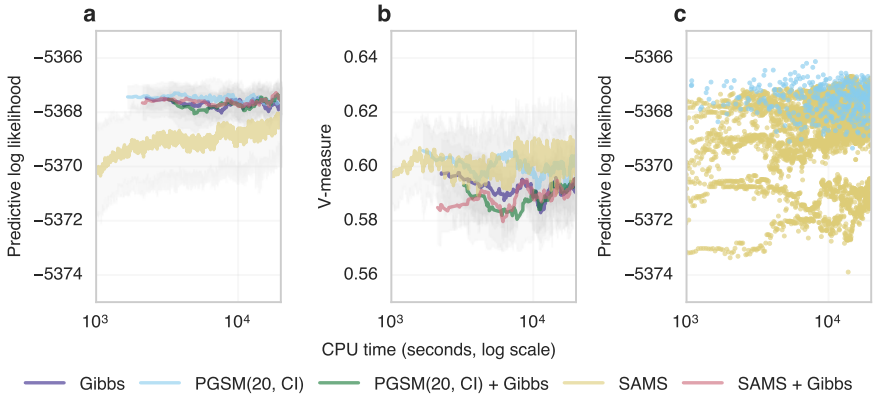


Figure 10: Clustering of cancer mutations from synthetic next-generation sequencing data. **a**) Comparison of predictive likelihood. **b**) Trace of log predictive likelihood for PGSM and SAMS pure kernels from 10 random restarts.

We show results on a dataset with 10,000 synthetic mutations in Figure 4.5. All methods except the pure SAMS kernel performed similarly in terms of predictive likelihood, while the pure SAMS kernel performed significantly worse (Figure 4.5 **a**). The pure PGSM and SAMS kernels outperformed the other methods in terms of V-measure, though the difference were small (Figures 4.5 **b**).

As observed in the other domains, the performance of SAMS critically depends on mixing the kernel with GIBBS moves. We show the data points for each replication of the pure split-merge kernels further supporting this point (Figure 4.5 **c**).

5. Discussion

We have proposed a new methodology to design efficient split-merge moves for Bayesian mixture models. The method also generalizes to new types of moves useful for finite clustering models when $|s| > 2$. We have shown empirically that the proposed method is competitive in a range of clustering and likelihood models, including synthetic and real datasets from geolocation and genomics applications.

Our method, being based on the established PMCMC framework, opens up many directions for future improvements. This includes applying recent advances in parallel implementations of SMC, for example via graphical processing units (Lee et al., 2010), or modifications of the SMC algorithm itself (Jun et al., 2012; Murray et al., 2016; Lee and Whiteley, 2016).

Another area of improvement comes from the development of resampling schemes tailored to discrete latent variables. In Algorithm 3, the number of possible distinct successors for each given particle is a small finite number (at most two if $|s| = 2$ for example). The complexity of the problem comes from the fact that a potentially long sequence of such decisions need to be made in order to split a cluster. In these specific scenarios, custom PMCMC methods based on the early work of Fearnhead and Clifford (2003) have been developed in Whiteley et al. (2010) and would provide futher improvement.

The fact that the state transitions $\mathcal{S}(\cdot)$ have an absorbing state has both advantages and disadvantages. On the one hand it may cause Algorithm 3 to be greedy, as explained in Section 3.4. We have described in the same section a choice of intermediate and proposal distributions tailored to alleviate this issue. A potential alternative consists in designing a resampling distribution r, which conditions on the survival of at least one representative of both a merge and a split. None of the existing resampling schemes have this property. On the other hand, having an absorbing state has the advantage that if all particles simulated by Algorithm 3 at some iteration t are equal to the merge absorbing state (i.e. $x_t^p = \#2$ for all particle index $p \in \{1, \ldots, N\}$), then there is no need to continue the computation of the particle filter for $t' > t$.

In standard applications of the PG algorithm, coalescence of the particle genealogy may cause slow mixing as noted in Andrieu et al. (2010). The issue is that the particles $\mathbf{x}_n^1, \mathbf{x}_n^2, \ldots, \mathbf{x}_n^N$ appearing in Algorithm 3 have components at time t for $t \ll n$ which coincide with high probability with the components of the conditioning path. This can be resolved using more sophisticated MCMC moves on the PG auxiliary variables (Whiteley, 2010; Whiteley et al., 2010; Lindsten et al., 2014). In our non-standard setup, this issue is partially mitigated by the fact that the order $\boldsymbol{\sigma}$ at which the particles are introduced is itself random. Nonetheless, it would be interesting to implement these more advanced schemes to the problem at hand.

We have shown in Section 4.5 a simple and effective method for handling models where each cluster component is governed by a non-conjugate model with a low-dimensional parameterization. We leave for future work the extension of our method to higher dimensional non-conjugate likelihood models. This problem can be approached, for example, by combining our method with the auxiliary variables described in Neal (2000).

Acknowledgements

Alexandre Bouchard-Côté's research was funded by an NSERC Discovery Grant. Arnaud Doucet's research is partially supported by EPSRC grants EP/K000276/1 and EP/K009850/1. Andrew Roth was partially supported by a CIHR CGS scholarship. Computing was supported by WestGrid and Compute Canada.

Appendix A. Correctness of the decomposition into split-merge subproblems

We present in this section the proof of correctness of the decomposition of the clustering into split-merge sub-problems. The main tool used to prove this result is an auxiliary variable construction. The auxiliary variable consists of a pair (s, c_-), where s is the set of anchors,

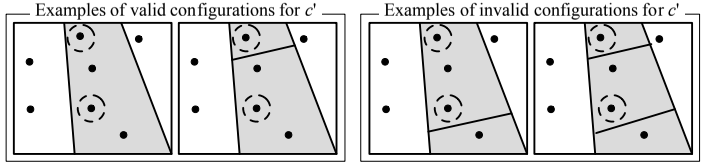


Figure 11: Assuming that the values of c, c_- and s are as in Figure 1, this illustrates some examples of configurations c' that are part of (left), or excluded from (right), the support of $c'|s, c_-$.

and c_- consists of the blocks of the partition c that do not contain anchor points:

$$c_- \coloneqq \{b \in c : b \cap s = \emptyset\}, \tag{17}$$

These intuitively correspond to the blocks of the partition that are forced to stay unchanged in this split-merge step. We will view the split-merge step as a Gibbs step conditioning on c_-, s.

A slight subtlety is that conditioning on the auxiliary variables not only forces the blocks in c_- to stay constant; it also forces the other blocks to each contain at least one of the anchors. See Figure 11 for an example. This leads to condition 2 in Lemma 5.

Lemma 5 *Let c, c' denote two partitions of $[T]$. Let $s \subseteq [T]$. Define $c_- \coloneqq \{b \in c : s \cap b = \emptyset\}$ and $c_-{}' \coloneqq \{b \in c' : s \cap b = \emptyset\}$. Then $c_- = c_-{}'$ if and only if the following two conditions hold:*

1. *$c \cap c_- = c' \cap c_-$, and,*
2. *$b \in c' \backslash c_- \implies b \cap s \neq \emptyset$.*

Proof ($\implies$) Condition 1 holds trivially. For condition 2, suppose (a) $c_- = c_-{}'$, (b), $b \in c' \backslash c_-$, but (c) $b \cap s = \emptyset$. By (b), $b \in c'$ and $b \notin c_-$. This and (c) implies that $b \in c_-{}'$. But this contradicts (a), so condition 2 holds as well.

($\impliedby$) First, suppose $b \in c_-$. By condition 1, $b \in c_-{}'$. Therefore, $c_- \subseteq c_-{}'$.

Second, suppose $b \in c_-{}'$. By the contrapositive of condition 2, $b \notin c' \backslash c_-$. This point and $c_-{}' \subseteq c'$ implies that $b \in c_-$. ■

We can now turn to the proof of Proposition 1. We copy its statement here for convenience:

Proposition 6 *If $c \sim \pi$, then the output of Algorithm 1, c', satisfies $c' \sim \pi$; i.e. the Markov kernel $K(c'|c)$ induced by Algorithm 1 is π-invariant.*

Proof Consider the model augmented with the auxiliary variables s and c_- (see Figure 12(a)), defined formally using the following auxiliary distribution:

$$\tilde{\pi}(s, c_-, c) \coloneqq \pi(c) h(s) \mathbf{1}[c_- = c_-(s, c)], \tag{18}$$

where $c_-(s, c) \coloneqq \{b \in c : b \cap s = \emptyset\}$. Note that this auxiliary distribution admits the target distribution as a marginal:

$$\begin{aligned} \sum_s \sum_{c_-} \tilde{\pi}(s, c_-, c) &= \pi(c) \sum_s h(s) \sum_{c_-} \mathbf{1}[c_- = c_-(s, c)] \\ &= \pi(c) \sum_s h(s) = \pi(c), \end{aligned} \tag{19}$$

where the sum over c_- is over all sets of subsets of $[T]$, and the sum over s is over all subsets of $[T]$. We used the fact that only one c_- satisfies $c_-(s, c) = c_-$, and that h is a probability mass function.

Next, we introduce three kernels with inputs and outputs denoted by:

$$c \overset{K_1}{\longmapsto} (s, c_-, c) \overset{K_2}{\longmapsto} (s, c_-, c') \overset{K_3}{\longmapsto} c'. \tag{20}$$

These kernels play the following roles:

- K_1 samples the auxiliary variables according to $\tilde{\pi}(s, c_- \mid c)$, while keeping c fixed,
- K_2 performs a Metropolis-within-Gibbs step on c targeting the auxiliary distribution $\tilde{\pi}$,
- K_3 deterministically projects the triplet back to the original space, retaining only the clustering c.

Formally:

$$\begin{aligned} K_1(s', c_-{}', c' \mid c) &\coloneqq h(s') \mathbf{1}[c = c'] \mathbf{1}[c_-{}' = c_-(s, c)], \\ K_2(s', c_-{}', c' \mid s, c_-, c) &\coloneqq \tilde{\pi}(c'|s, c_-) \mathbf{1}[s' = s] \mathbf{1}[c_-{}' = c_-], \\ K_3(c'|s, c_-, c) &\coloneqq \mathbf{1}[c' = c]. \end{aligned} \tag{21}$$

Since $\tilde{\pi}$ admits π as a marginal, the composition of K_1, K_2, and K_3 is clearly π-invariant. It is therefore enough to show that when $c_- = c_-(s, c)$ where c is a valid partition of $[T]$, sampling from K_2 is equivalent to sampling from the Markov kernel $K(c'|c)$ induced by Algorithm 1:

$$\begin{aligned} \tilde{\pi}(c' \mid s, c_-) &\propto \pi(c') \mathbf{1}[c_- = c_-(s, c')] \\ &= \pi(c') \mathbf{1}[c_-(s, c) = c_-(s, c')] \\ &= \tau_1(|c'|) \left(\prod_{b \in c'} \tau_2(|b|) L(\mathbf{y}_b) \right) \mathbf{1}[c_-(s, c) = c_-(s, c')]. \end{aligned} \tag{22}$$

Using Lemma 5, we now rewrite the support as follows:

$$\mathbf{1}[c_-(s, c') = c_-(s, c)] = \mathbf{1}[c \cap c_- = c' \cap c_-] \mathbf{1}[b \in c' \backslash c_- \implies b \cap s \neq \emptyset]. \tag{23}$$



Figure 12: Graphical models of the auxiliary variables used in the correctness proofs. The structure of the dependencies give an intuitive justification that the original model can be recovered as a marginal in both cases, as there are not directed path from the auxiliary variables to the original variables. (a) In Appendix A, the auxiliary variables are s and c_-, and the original variable is c. (b) In Appendix B, the auxiliary variables are $\boldsymbol{\sigma}$ and g, and the original variable is $\overline{c}$.

Let now $\overline{c}' = c' \backslash c_-$. Plugging in the last line of Equation (22), we obtain:

$$\begin{aligned} \tilde{\pi}(c' \mid s, c_-) &= \overline{\tau}_1(|\overline{c}'|) \left(\prod_{b \in \overline{c}'} \tau_2(|b|) L(\mathbf{y}_b) \right) \mathbf{1}[c \cap c_- = c' \cap c_-] \mathbf{1}[b \cap s \neq \emptyset] \qquad (24) \\ &= \overline{\pi}(\overline{c}') \mathbf{1}[c \cap c_- = c' \cap c_-], \end{aligned}$$

where $\overline{\pi}(\overline{c}')$ is defined in Equation (5). Since Algorithm 1 does not change the clustering of points outside of $\overline{s}$ (line 6 of Algorithm 1), it follows that the indicator function in the last line of Equation (24) is equal to one. ■

Appendix B. Correctness of particle Gibbs for split merge

We provide here the proof of Proposition 2. The main steps in the proof follow a structure similar to the proof of Proposition 1.

Proposition 7 *Under Assumption 1, 2, and 3, and if $\overline{c} \sim \overline{\pi}$, then the output of Algorithm 3, $\overline{c}'$, satisfies $\overline{c}' \sim \overline{\pi}$ for any $N \geq 2$; i.e. the Markov kernel $\overline{K}(\overline{c}'|\overline{c})$ induced by Algorithm 3 is $\overline{\pi}$-invariant.*

Proof We augment the model $\overline{c}$ with the auxiliary variables $\boldsymbol{\sigma}$ and g (see Figure 12(b)), defined as:

1. $\boldsymbol{\sigma}$ is distributed according to the output of Algorithm 2, defined in Section 3.2.
2. Given $\boldsymbol{\sigma}$ and $\overline{c}$, the variables $g = (\boldsymbol{a}_{2:n}, \mathbf{x}_{1:n}^{1:N}, k)$ are distributed according to the specification of Algorithm 3, with the exception that all particle indices are shuffled according to an independent permutation of $\{1, \ldots, N\}$ at each generation. Here k is the index of the particle sampled at iteration n (on line (35)).

Next, we introduce three kernels with inputs and outputs denoted by:

$$\overline{c} \overset{\overline{K}_1}{\longmapsto} (\boldsymbol{\sigma}, \overline{c}) \overset{\overline{K}_2}{\longmapsto} (\boldsymbol{\sigma}, g, \overline{c}') \overset{\overline{K}_3}{\longmapsto} \overline{c}'. \qquad (25)$$

These kernels play the following roles:

- $\overline{K}_1$ samples the permutation $\boldsymbol{\sigma}$ while keeping the auxiliary variables $\overline{c}$ fixed,
- $\overline{K}_2$ samples g using the PG step then sets $\overline{c}'$ to $\phi^{\boldsymbol{\sigma}}(\mathbf{x}_n^k)$,
- $\overline{K}_3$ deterministically projects the triplet back to the original space, retaining only the restricted clustering $\overline{c}'$.

The kernel $\overline{K}_2$ is equivalent to a standard PG algorithm. Assumption 1, 3, and Theorem 5(a) of Andrieu et al. (2010) imply that $\overline{K}_2$ is $\overline{\pi}$-invariant (and in fact, irreducible). Assumption 2 ensures that the computation of the conditioned path is well-defined. ■

Appendix C. Construction of the bijection

We provide here the proof of Proposition 3:

Proposition 8 *For any permutation $\boldsymbol{\sigma}$ satisfying $\{\sigma_1, \sigma_2\} = s$, there is a bijective map $\phi^{\boldsymbol{\sigma}}$ from the space of particles respecting the transition constraints, $\mathcal{S}_n$, to the support of the restricted target,* $\mathrm{support}(\overline{\pi})$.

Proof Consider the following mapping:

$$\phi^{\boldsymbol{\sigma}}(\mathbf{x}_t) \coloneqq \begin{cases} \{\{\sigma_1, \ldots, \sigma_t\}\} & \text{if } x_t \in \{\#1, \#2\}, \\ \{\overline{\sigma}_1(\mathbf{x}_t), \overline{\sigma}_2(\mathbf{x}_t)\} & \text{otherwise,} \end{cases} \qquad (26)$$

where $\overline{\sigma}_i(\mathbf{x}_t) \coloneqq \{\sigma_{t'} : x_{t'} = \#(2+i), 1 \leq t' \leq t\}$. It is easy to check that it has an inverse given by:

$$\left((\phi^{\boldsymbol{\sigma}})^{-1} (\overline{c}) \right)_t \coloneqq \begin{cases} \#1 & \text{if } t = 1, \\ \#2 & \text{if } t > 1, |\overline{c}| = 1, \\ \#3 & \text{if } t > 1, |\overline{c}| > 1, \sigma_1 \sim_{\overline{c}} \sigma_t, \\ \#4 & \text{if } t > 1, |\overline{c}| > 1, \sigma_2 \sim_{\overline{c}} \sigma_t, \end{cases} \qquad (27)$$

where $\sigma_i \sim_{\overline{c}} \sigma_j$ means that y_{σ_i} is in the same block as y_{σ_j} for the clustering $\overline{c}$. By the construction of the support of $\overline{\pi}$, exactly one of the four cases above holds when $\overline{c} \in \mathrm{support}(\overline{\pi})$.

■

Appendix D. Generalization to $|s| > 2$

We describe here the algorithmic implications of increasing the number of anchor points, $|s|$, to some constant greater than two. This constant should be selected so that the number of partitions of $|s|$ points is much lower than the number of particles.

The algorithm is generally unchanged, with the following exceptions:

1. Algorithm 2 is modified to sample $(\sigma_1, \sigma_2, \ldots, \sigma_{|s|})$ uniformly over the permutations of s, and $(\sigma_{|s|+1}, \ldots, \sigma_n)$, over the permutations of $\overline{s} \backslash s$,

2. as before, the local allocation state space $\mathcal{X}$ can be viewed as a pair each containing a partition and a block in this partition (see Figure 2). In the case where $|s| = 2$, the partitions are taken from the union of the set of partitions of a set of size one with the set of partitions of a set of size two. When $|s| > 2$, we add more states, corresponding to partitions of a set of size three, etc. until we add states corresponding to partitions of a set of size $|s|$. The support of the transition $\mathcal{S}$ consists in (a) edges $x \to x'$ linking a state x' such that removing one element from one of its blocks yields x, and (b) edges $x \to x'$ where x and x' correspond to the same partition of a set of size $|s|$. This is a generalization of the case $|s| = 2$ shown in Figure 2. The mapping $\phi^{\boldsymbol{\sigma}}$ is generalized in the obvious way,

3. in Section 3.4 the following equations are substituted,

   (a) $t \in \{1, 2\} \to t \in \{1, 2, \ldots, |s|\}$,
   (b) $t = 2 \to t \in \{2, \ldots, |s|\}$,
   (c) $t > 2 \to t > |s|$,
   (d) $\Delta\zeta \coloneqq (n-2)^{-1} \to \Delta\zeta \coloneqq (n - |s|)^{-1}$.

Appendix E. Models

E.1 Multivariate normal

The first likelihood we use is the multivariate normal (MVN) with density denoted $\mathcal{N}(y|\mu, \Sigma)$. We specify a normal inverse Wishart (NIW) prior for the mean and covariance parameters with density denoted $\mathcal{NIW}(\mu, \Sigma|\nu, r, u, S)$. The densities are given by

$$\begin{aligned}
\mathcal{NIW}(\mu, \Sigma|\nu, r, u, S) &= \mathcal{N}\left(\mu|u, \frac{1}{r}\Sigma\right)\mathcal{IW}(\Sigma|\nu, S), \qquad (28)\\
\mathcal{N}(y|\mu, \Sigma) &= \frac{1}{(2\pi)^{\frac{D}{2}}|\Sigma|^{\frac{1}{2}}}\exp\left(-\frac{1}{2}(y-\mu)^T\Sigma^{-1}(y-\mu)\right),\\
\mathcal{IW}(\Sigma|\nu, S) &= \frac{|S|^{\frac{\nu}{2}}}{2^{\frac{\nu p}{2}}\Gamma_D\left(\frac{\nu}{2}\right)}|\Sigma|^{-\frac{\nu+p+1}{2}}\exp\left(-\frac{1}{2}\mathrm{tr}(S\Sigma^{-1})\right),
\end{aligned}$$

where $\Gamma_D(x) = \pi^{\frac{D(D-1)}{4}}\prod_{d=1}^{D}\Gamma\left(x + \frac{d-1}{2}\right)$.

We use the following priors for all experiments $(\nu, r, u, S) = (\nu_0, r_0, u_0, S_0) = (2 + D, 1, \mathbf{0}, \boldsymbol{I})$, where $\mathbf{0}$ is the D dimensional vector of zeros, and $\boldsymbol{I}$ is the D dimensional identity matrix. The posterior distribution of μ, Σ given $\boldsymbol{y} = (y_1, \ldots, y_m)$ is $\mathcal{NIW}(\mu, \Sigma|\nu_m, r_m, u_m, S_m)$ where

$$\begin{aligned}
\nu_m &= \nu_0 + m, \qquad (29)\\
r_m &= r_0 + m,\\
u_m &= \frac{r_0 u_0 + \sum_{i=1}^{m} y_i}{r_m},\\
S_m &= S_0 + \sum_{i=1}^{m} y_i y_i^T + r_0 u_0 u_0^T - r_m u_m u_m^T.
\end{aligned}$$

For computational efficiency it is convenient to express these updates iteratively using the following equations:

$$\begin{aligned}
\nu_m &= \nu_{m-1} + 1, \qquad (30)\\
r_m &= r_{m-1} + 1,\\
u_m &= \frac{r_{m-1}u_{m-1} + y_m}{r_m},\\
S_m &= S_{m-1} + \frac{r_m}{r_{m-1}}(y_m - u_m)(y_m - u_m)^T.
\end{aligned}$$

Using these equations the Cholesky decomposition of S_0 can be performed once using $O(D^3)$ operations and cached. This decomposition can then be updated using m rank one updates, each requiring $O(D^2)$ operations, to obtain S_m. This allows for efficient evaluation of the marginal and predictive likelihoods as $|S_m|$ can be evaluated using $O(D)$ operations using the Cholesky decomposition, instead of the standard $O(D^3)$ operations.

The marginal likelihood for the MVN-NIW congugate pair is

$$\begin{aligned}
L(\boldsymbol{y}) &= \int\prod_{i=1}^{m} L(y_i|\theta)H(\mathrm{d}\theta) \qquad (31)\\
&= \int\prod_{i=1}^{m}\mathcal{N}(y_i|\mu, \Sigma)\mathcal{NIW}(\mu, \Sigma|\nu, r, u, S)\,\mathrm{d}\mu\,\mathrm{d}\Sigma\\
&= \frac{1}{\pi^{\frac{mD}{2}}}\frac{r_0^{\frac{D}{2}}}{r_m^{\frac{D}{2}}}\frac{|S_0|^{\frac{\nu_0}{2}}}{|S_m|^{\frac{\nu_m}{2}}}\frac{\prod_{d=1}^{D}\Gamma\left(\frac{\nu_m+d-1}{2}\right)}{\prod_{d=1}^{D}\Gamma\left(\frac{\nu_0+d-1}{2}\right)}.
\end{aligned}$$

The predictive likelihood is given by

$$\begin{aligned}
L(\boldsymbol{y}^+|\boldsymbol{y}^-) &= \frac{L(y_1, \ldots, y_m)}{L(y_1, \ldots, y_{m-1})} \qquad (32)\\
&= \frac{1}{\pi^{\frac{D}{2}}}\frac{r_{m-1}^{\frac{D}{2}}}{r_m^{\frac{D}{2}}}\frac{|S_{m-1}|^{\frac{\nu_{m-1}}{2}}}{|S_m|^{\frac{\nu_m}{2}}}\frac{\prod_{d=1}^{D}\Gamma\left(\frac{\nu_m+d-1}{2}\right)}{\prod_{d=1}^{D}\Gamma\left(\frac{\nu_{m-1}+d-1}{2}\right)}.
\end{aligned}$$

E.2 Bernoulli

We use a Bernoulli likelihood, $\mathrm{Bernoulli}(x|\theta)$, with a Beta prior distribution, $\mathrm{Beta}(\theta|\alpha, \beta)$. We use the following priors $(\alpha, \beta) = (\alpha_0, \beta_0) = (1, 1)$ for all experiments. The densities are

$$\begin{aligned}
\mathrm{Bernoulli}(x|\theta) &= \theta^x(1-\theta)^{1-x}, \qquad (33)\\
\mathrm{Beta}(\theta|\alpha, \beta) &= \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}\theta^{\alpha-1}\theta^{\beta-1}.
\end{aligned}$$

The posterior density of θ given $\boldsymbol{y} = (y_1, \ldots, y_m)$ is $\text{Beta}(\alpha_m, \beta_m)$ where $\alpha_m = \alpha_0 + \sum_{i=1}^m y_i$ and $\beta_m = \beta_0 + \sum_{i=1}^m (1 - y_i)$. The marginal likelihood is

$$\begin{aligned} L(\boldsymbol{y}) &= \int \prod_{i=1}^m L(y_i|\theta) H(\mathrm{d}\theta) \\ &= \int \prod_{i=1}^m \text{Bernoulli}(y_i|\theta)\text{Beta}(\theta|\alpha_0, \beta_0)\,\mathrm{d}\theta \\ &= \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha_m)\Gamma(\beta_m)} \frac{\Gamma(\alpha_m + \beta_m)}{\Gamma(\alpha_0 + \beta_0)}, \end{aligned} \tag{34}$$

and the predictive log likelihood is

$$\begin{aligned} L(\boldsymbol{y}^+|\boldsymbol{y}^-) &= \frac{L(y_1, \ldots, y_m)}{L(y_1, \ldots, y_{m-1})} \\ &= \frac{\Gamma(\alpha_{m-1})\Gamma(\beta_{m-1})}{\Gamma(\alpha_m)\Gamma(\beta_m)} \frac{\Gamma(\alpha_m + \beta_m)}{\Gamma(\alpha_{m-1} + \beta_{m-1})}. \end{aligned} \tag{35}$$

E.3 PyClone

For the cancer genomics data we use the application-specific PyClone likelihood model over clonal prevalences, genotypes, and observed read counts. The key variables in the model are as follows (see Roth et al. (2014) for a more detailed description of the model):

$$\begin{aligned} \phi_i &: \text{proportion of cancer cells with mutation } i,\ \phi_i \in [0,1], \\ t &: \text{proportion of cancer cells in a sample (treated as known)},\ t \in [0,1], \\ \psi_i &: \text{genotype of normal, non-mutated cancer and mutated cancer cells},\ \psi_i \in (g_N, g_R, g_V), \\ g_x &\in \mathcal{G} = \{\text{A}, \text{B}, \text{AA}, \text{AB}, \ldots\}, \\ \pi_{i,\psi_i} &: \text{probability that mutation } i \text{ has genotype } \psi_i \text{ (elicited from auxillary data)}, \\ c(g_x) &= \#\text{A}(g_x) + \#\text{B}(g_x), \\ \mu(g_x) &= \frac{\#\text{A}(g_x)}{c(g_x)}, \\ \xi(\psi, \phi, t) &: \text{probability of sampling a B from the population of cells in the sample, i.e.:} \\ &= \frac{(1-t)c(g_N)\mu(g_N) + t(1-\phi)c(g_R)\mu(g_R) + t\phi c(g_V)\mu(g_V)}{(1-t)c(g_N) + t(1-\phi)c(g_R) + t\phi c(g_V)}, \\ y_i &: \text{number of sequence reads with a B and total number of reads covering mutation } i, \text{ i.e.:} \\ &= (y_{i,b}, y_{i,d}) \in \mathbb{N}^2. \end{aligned}$$

The generative model is specified as follow:

$$\begin{aligned} H_0 &= \text{Uniform}([0,1]), \\ \alpha_0 &\sim \text{Gamma}(\alpha_0|a, b), \\ H|\alpha_0, H_0 &\sim \text{DP}(H|\alpha_0, H_0), \\ \phi_i|H &\sim H, \\ y_i|\psi_i, \phi_i, t &\sim \text{Binomial}(y_{i,b}|y_{i,d}, \xi(\psi_i, \phi, t)). \end{aligned} \tag{36}$$

This model is not conjugate. However, if we let $x \in \{x_0, \ldots, x_M\} = \left\{0, \frac{1}{M-1}, \ldots, \frac{M-2}{M-1}, 1\right\}$ be a discretization of the interval $[0,1]$ and replace the continuous uniform base measure, $H_0 = \text{Uniform}([0,1])$, with the discrete uniform measure, $H_0 = \text{Uniform}(\{x_0, \ldots, x_M\})$, then we can approximate the model. Using this approximation, we can now treat the model as if it were conjugate. The marginal likelihood for data $(y_1, ..., y_m)$ is given by

$$\begin{aligned} \int \prod_{i=1}^m L(y_i|\theta) H(\mathrm{d}\theta) &= \int \prod_{i=1}^m \sum_{\psi_i \in \mathcal{G}^3} \pi_{i,\psi_i} \text{Binomial}(y_{i,b}|y_{i,d}, \xi(\psi_i, \phi, t)) H(\mathrm{d}\phi) \\ &= \sum_{k=0}^M \prod_{i=1}^m \sum_{\psi_i \in \mathcal{G}^3} \pi_{i,\psi_i} \text{Binomial}(y_{i,b}|y_{i,d}, \xi(\psi_i, x_k, t)) \frac{1}{M} \\ &= \sum_{k=0}^M \prod_{i=1}^m \exp\left(\underbrace{\log \sum_{\psi_i \in \mathcal{G}^3} \pi_{i,\psi_i} \text{Binomial}(y_{i,b}|y_{i,d}, \xi(\psi_i, x_k, t))}_{\Xi_k(y_i)} \right) \frac{1}{M} \\ &= \sum_{k=0}^M \exp\left(\sum_{i=1}^m \Xi_k(y_i) \right) \frac{1}{M}, \end{aligned} \tag{37}$$

where we have the sufficient statistics

$$\boldsymbol{\Xi}(y_i) = (\Xi_0(y_i), \ldots, \Xi_M(y_i)). \tag{38}$$

Remark 9 *The possibly infinite sum $\sum_{\psi_i \in \mathcal{G}^3}$ is truncated to a finite sum over biologically plausible states.*

Appendix F. Anchor proposal distribution

The anchor proposal distribution, h, is a free tuning parameter for the PGSM sampler. In principle, proposals which are informed by the current clustering state of the chain or by the topology of the space may improve the performance of the sampler.

We consider two informed proposal distributions. While bespoke proposals for each model may perform better, we restrict attention here to proposals which can be applied generically to any class of model for which the PGSM sampler is applicable. In particular, we do not assume a distance metric is available. Both proposals we discuss are only applicable when two anchor points are used.

Remark 10 *If the any of the sets that we sample uniformly from are empty, we return two anchors sampled uniformly at random.*

Algorithm 4 Cluster informed (CI) proposal

$i_1 \sim \text{Uniform}([T])$
$\bar{b} \leftarrow b \in c$ s.t. $i_1 \in b$
$c' \leftarrow c \setminus \{\bar{b}\}$
for $b \in c'$ **do**
  $s_b \leftarrow \frac{L(y_{\bar{b} \setminus b})}{L(y_{\bar{b}})L(y_b)}$
end for
$s_{\bar{b}} \leftarrow \frac{\sum_{b \in c'} s_b}{|c|-1}$ ▷ Merge probability is set to $\frac{1}{|c|-1}$
for $b \in c$ **do**
  $p_b \leftarrow \frac{s_b}{\sum_{b \in c} s_b}$
end for
$b' \sim \text{Discrete}(c, p_b)$ ▷ Sample a block b' in c with probability p_b
$i_2 \sim \text{Uniform}(b' \setminus i_1)$
return i_1, i_2

Algorithm 5 Threshold informed (TI) proposal

$i_1 \sim \text{Uniform}([T])$
for $b \in c$ **do**
  if $i_1 \in b$ **then**
    $b \leftarrow b \setminus i_1$
  end if
  $s_b \leftarrow \tau_2(b) L(y_{i_1} | b)$ ▷ CRP attachment probability where $L(\cdot|b)$ is the predictive distribution
end for
for $b \in c$ **do**
  $p_b \leftarrow \frac{s_b}{\sum_{b \in c} s_b}$
end for
$b' \sim \text{Uniform}(\{b : p_b \geq t\})$ ▷ t is a pre-specified threshold, set to 0.01 in the experiments
$i_2 \sim \text{Uniform}(b' \setminus i_1)$
return i_1, i_2

List of Symbols

α_{Dir}	Symmetric concentration parameters for the finite Dirichlet model
β	Relative ESS threshold used for adaptive resampling
$\mathbf{x}$	Particle
$\mathbf{y}$	All observations
$\boldsymbol{\sigma}$	Permutation auxiliary variable vector
$\boldsymbol{a}$	Resampling ancestry vector
$\boldsymbol{w}$	Weight vector
k_0	Maximum number of clusters in the finite Dirichlet model
$\mathcal{S}$	Support defining valid sequences of allocation decisions
$\mathcal{X}$	State space containing the possible allocation decisions
$\overline{\pi}$	Split merge target distribution
$\overline{\tau}_1$	Split merge modification of the prior factor depending on the number of clusters
$\overline{c}$	Restriction of the clustering to the closure subset
$\overline{K}$	Markov kernel used to conceptualized a particle Gibbs step
$\overline{s}$	Closure of a subset of datapoint indices relative to a clustering
α_0	Concentration parameter used in clustering priors such as the Dirichlet process
τ_1	Factor in the clustering prior depending on the number of clusters
τ_2	Factor in the clustering prior depending on the size of each block
ψ	Sufficient statistics for the likelihood
γ	Intermediate un-normalized distributions
ϕ	Bijection from particles to restricted clusterings
π	Target density of interest
σ	Permutation auxiliary variable component
τ	Prior over clusterings
θ	Parameter of the parametric family
$\sim$	A binary relationship indicating whether two datapoint indices are in the same block (relative to some clustering specified as a subscript)
$\tilde{\pi}$	Auxiliary target distribution
ζ	Annealing schedule used to delay the introduction of the clustering prior
a	Resampling ancestry variable
b	One block in a clustering
c_-	Blocks of the partition outside of the closure
c	Partition or clustering of the observation indices
d	Discount parameter used in the Pitman-Yor process
g	Particle Gibbs auxiliary variables

H	Prior for the parametric family, and base measure of the process
h	Distribution of the anchors
i	Observation index
K	Markov kernel used to conceptualized a whole split-merge step
k	Index of a block (cluster)
L	Likelihood of the parametric family
N	Number of particles used in the particle Gibbs algorithms
n	Number of points to allocate, also coincides with the number of particle Gibbs generations (iterations)
p	Particle index
q	Proposal distribution for the particle Gibbs algorithm
r	Conditional resampling distribution
s	Subset of observation indices; in particular, set of anchor points
T	Number of datapoints
t	SMC generation (iteration) index
w	Weight variable
x	Allocation decision (a particle is made of a vector of such allocations)
y	One observed datapoint

References

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo for efficient numerical simulation. In *Monte Carlo and Quasi Monte Carlo Methods Conference 2008, Lecture Notes in Statistics*, pages 45–60. Springer-Verlag, 2009.

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B*, 72(3):269–342, 2010.

F. Caron and A. Doucet. Bayesian nonparametric models on decomposable graphs. In *Advances in Neural Information Processing Systems*, pages 225–233. 2009.

C. Carvalho, H. Lopes, N. Polson, and M. Taddy. Particle learning for general mixtures. *Bayesian Analysis*, 5(4):709–740, 12 2010.

J. Chang and J. W. Fisher III. Parallel sampling of Dirichlet process mixutre models using sub-clusters splits. In *Advances in Neural Information Processing Systems*, pages 620–628, 2013.

N. Chopin and S. S. Singh. On particle Gibbs sampling. *Bernoulli*, 21(3):1855–1883, 2015.

D. B. Dahl. An improved merge-split sampler for conjugate Dirichlet process mixture models. Technical Report 1086, University of Wisconsin–Madison, 2003.

D. B. Dahl. Modal clustering in a univariate class of product partition models. *Bayesian Analysis*, 4(2):243–264, 2009.

H. Daumé III. Fast search for Dirichlet process mixture models. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2007.

P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B*, 68(3):411–436, 2006.

A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. In *Oxford Handbook of Nonlinear Filtering*. Oxford University Press, 2011.

M. D. Escobar and M. West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430):577–588, 1995.

P. Fearnhead. Particle filters for mixture models with an unknown number of components. *Statistics and Computing*, 14:11–21, 2004.

P. Fearnhead and P. Clifford. On-line inference for hidden Markov models via particle filters. *Journal of the Royal Statistical Society: Series B*, 65(4):887–899, 2003.

P. Fearnhead and L. Meligkotsidou. Filtering methods for mixture models. *Journal of Computational and Graphical Statistics*, 16(3):586–607, 2007.

T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

P. Franti and O. Virmajoki. Iterative shrinking method for clustering problems. *Pattern Recognition*, 39(5):761–765, 2006.

P. Fränti, O. Virmajoki, and V. Hautamäki. Fast agglomerative clustering using a k-nearest neighbor graph. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 28(11):1875–1881, 2006.

P. Fränti, J. Kuittinen, A. Tabarcea, and L. Sakala. MOPSI location-based search engine: concept, architecture and prototype. In *ACM Symposium on Applied Computing*, volume 10, pages 872–873, 2010.

Y. Gal and Z. Ghahramani. Pitfalls in the use of parallel inference for the Dirichlet process. In *Proceedings of the 31st International Conference on Machine Learning*, pages 208–216. JMLR Workshop and Conference Proceedings, 2014.

H. Ge, Y. Chen, M. Wan, and Z. Ghahramani. Distributed inference for Dirichlet process mixture models. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2276–2284, 2015.

H. Ishwaran and L. F. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 96:161–173, 2001.

H. Ishwaran and L. F. James. Generalized weighted Chinese restaurant processes for species sampling mixture models. *Statistica Sinica*, 13:1211–1235, 2003.

S. Jain and R. M. Neal. A split-merge Markov chain Monte Carlo procedure for the Dirichlet Process mixture model. *Journal of Computational and Graphical Statistics*, 13:158–182, 2004.

A. M. Johansen. SMCTC: Sequential Monte Carlo in C++. *Journal of Statistical Software*, 30(6):1–41, 2009.

S. Jun, L. Wang, and A. Bouchard-Côté. Entangled Monte Carlo. In *Advances in Neural Information Processing Systems*, volume 25, pages 2735–2743, 2012.

M. Kalli, J. E. Griffin, and S. G. Walker. Slice sampling mixture models. *Statistics and Computing*, 21(1):93–105, 2011.

N. Kantas, A. Doucet, S. S. Singh, J. M. Maciejowski, and N. Chopin. On particle methods for parameter estimation in state-space models. *Statistical Science*, 30(3):328–351, 2015.

A. Lee. *On auxiliary variables and many-core architectures in computational statistics*. PhD thesis, Department of Statistics, Oxford University, 2011.

A. Lee and N. Whiteley. Forest resampling for distributed sequential Monte Carlo. *Statistical Analysis and Data Mining*, 9(4):230–248, 2016.

A. Lee, C. Yau, M. Giles, A. Doucet, and C. Holmes. On the utility of graphics cards to perform massively parallel simulation of advanced Monte Carlo methods. *Journal of Computational and Graphical Statistics*, 19(4):769–789, 2010.

P. Liang, M. I. Jordan, and B. Taskar. A permutation-augmented sampler for Dirichlet process mixture models. *Proceedings of the International Conference on Machine learning*, 24:545–552, 2007.

F. Lindsten, M. I. Jordan, and T. B. Schön. Particle Gibbs with ancestor sampling. *Journal of Machine Learning Research*, 15:2145–2184, 2014.

J.S. Liu and R. Chen. Blind deconvolution via sequential imputation. *Journal of the American Statistical Association*, 90:567–576, 1995.

S. N. MacEachern. Estimating normal means with a conjugate style Dirichlet process prior. *Communications in Statistics: Simulation and Computation*, 23:727–741, 1994.

S. N. MacEachern, M. Clyde, and J. S. Liu. Sequential importance sampling for nonparametric Bayes models: The next generation. *Canadian Journal of Statistics*, 27(2): 251–267, 1999.

V. K. Mansinghka, D. M. Roy, R. Rifkin, and J. Tenenbaum. AClass: An online algorithm for generative classification. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, volume 2, pages 315–322, 2007.

J. Marin, K. L. Mengersen, and C. Robert. Bayesian modelling and inference on mixtures of distributions. In D. Dey and C.R. Rao, editors, *Handbook of Statistics: Volume 25*. Elsevier, 2005.

L. Murray, A. Lee, and P. Jacob. Parallel resampling in the particle filter. *Journal of Computational and Graphical Statistics*, 25(3):789–805, 2016.

L. M. Murray. Bayesian state-space modelling on high-performance hardware using LibBi. *Journal of Statistical Software*, 67(10):1–36, 2015.

R. M. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9:249–265, 2000.

J. Pitman and M. Yor. The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *Annals of Probability*, 25(2):855–900, 1997.

S. Richardson and P. J. Green. On Bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society: Series B*, 59:731–792, 1997.

Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, volume 7, pages 410–420, 2007.

A. Roth, J. Khattra, D. Yap, A. Wan, E. Laks, J. Biele, G. Ha, S. Aparicio, A. Bouchard-Côté, and S. P. Shah. PyClone: statistical inference of clonal population structure in cancer. *Nature methods*, 11(4):396–398, 2014.

E. B. Sudderth. *Graphical Models for Visual Object Recognition and Tracking*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2006.

Y. Ülker, B. Günsel, and A. T. Cemgil. Sequential Monte Carlo samplers for Dirichlet process mixtures. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, volume 9, pages 876–883, 2010.

S. G. Walker. Sampling the Dirichlet mixture model with slices. *Communications in Statistics - Simulation and Computation*, 36(1):45–54, 2007.

L. Wang and D. B. Dunson. Fast Bayesian inference in Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 20(1):196–216, 2011.

N. Whiteley. Discussion on Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B*, 72:306–307, 2010.

N. Whiteley, C. Andrieu, and A. Doucet. Efficient Bayesian inference for switching state space models using discrete particle Markov chain Monte Carlo methods. Technical Report 10:04, Bristol Statistics Research Report, 2010.

S. Williamson, A. Dubey, and E. Xing. Parallel Markov chain Monte Carlo for nonparametric mixture models. In *Proceedings of the 30th International Conference on Machine Learning*, pages 98–106, 2013.

Journal of Machine Learning Research 18 (2017) 1-53 Submitted 11/15; Revised 2/17; Published 4/17

Certifiably Optimal Low Rank Factor Analysis

Dimitris Bertsimas DBERTSIM@MIT.EDU
Sloan School of Management
MIT
Cambridge, MA 02139, USA

Martin S. Copenhaver MCOPEN@MIT.EDU
Operations Research Center
MIT
Cambridge, MA 02139, USA

Rahul Mazumder RAHULMAZ@MIT.EDU
Sloan School of Management
MIT
Cambridge, MA 02139, USA

Editor: Benjamin Recht

Abstract

Factor Analysis (FA) is a technique of fundamental importance that is widely used in classical and modern multivariate statistics, psychometrics, and econometrics. In this paper, we revisit the classical rank-constrained FA problem which seeks to approximate an observed covariance matrix ($\mathbf{\Sigma}$) by the sum of a Positive Semidefinite (PSD) low-rank component ($\mathbf{\Theta}$) and a diagonal matrix ($\mathbf{\Phi}$) (with nonnegative entries) subject to $\mathbf{\Sigma} - \mathbf{\Phi}$ being PSD. We propose a flexible family of rank-constrained, nonlinear Semidefinite Optimization based formulations for this task. We introduce a reformulation of the problem as a smooth optimization problem with convex, compact constraints and propose a unified algorithmic framework, utilizing state of the art techniques in nonlinear optimization to obtain high-quality feasible solutions for our proposed formulation. At the same time, by using a variety of techniques from discrete and global optimization, we show that these solutions are *certifiably optimal* in many cases, even for problems with thousands of variables. Our techniques are general and make *no* assumption on the underlying problem data. The estimator proposed herein aids statistical interpretability and provides computational scalability and significantly improved accuracy when compared to current, publicly available popular methods for rank-constrained FA. We demonstrate the effectiveness of our proposal on an array of synthetic and real-life datasets. To our knowledge, this is the first paper that demonstrates how a previously intractable rank-constrained optimization problem can be solved to provable optimality by coupling developments in convex analysis and in global and discrete optimization.

Keywords: factor analysis, rank minimization, semidefinite optimization, first order methods, nonlinear optimization, global optimization, discrete optimization

1. Introduction

Factor Analysis (FA) (Anderson, 2003; Bartholomew et al., 2011; Mardia et al., 1979), a widely used methodology in classical and modern multivariate statistics, is used as a tool to obtain a parsimonious representation of the correlation structure among a set of variables in terms of a smaller number of common hidden factors. A basic FA model is of the form $\mathbf{x} = \mathbf{L}\mathbf{f} + \boldsymbol{\epsilon}$, where $\mathbf{x}_{p\times 1}$ is the observed random vector, $\mathbf{f}_{r_1\times 1}$ (with $r_1 \leq p$; note that we do not necessarily restrict r_1 to be small) is a random vector of common factor variables or scores, $\mathbf{L}_{p\times r_1}$ is a matrix of factor loadings and $\boldsymbol{\epsilon}_{p\times 1}$ is a vector of uncorrelated random variables. We assume that the variables are mean-centered, $\mathbf{f}$ and $\boldsymbol{\epsilon}$ are uncorrelated and without loss of generality, the covariance of $\mathbf{f}$ is the identity matrix. We will denote $\text{Cov}(\boldsymbol{\epsilon}) = \mathbf{\Phi} = \text{diag}(\Phi_1, \ldots, \Phi_p)$. It follows that

$$\mathbf{\Sigma} = \mathbf{\Sigma}_c + \mathbf{\Phi}, \tag{1}$$

where $\mathbf{\Sigma}$ is the covariance matrix of $\mathbf{x}$ and $\mathbf{\Sigma}_c := \mathbf{L}\mathbf{L}'$ is the covariance matrix corresponding to the common factors. Decomposition (1) suggests that $\mathbf{\Sigma}$ can be written as the sum of a positive semidefinite (PSD) matrix $\mathbf{\Sigma}_c$ of rank r_1 and a nonnegative diagonal matrix ($\mathbf{\Phi}$) corresponding to the errors. In particular, the variance of the ith coordinate of $\mathbf{x} := (x_1, \ldots, x_p)$, i.e., $\text{var}(x_i) = \sum_k l_{ik}^2 + \Phi_i, i = 1, \ldots, p$, splits into two parts, where $\mathbf{L} = ((l_{ik}))$. The first part $(\sum_k l_{ik}^2)$ is known as the *communality estimate* (since this is the variance of the factors common to all the x_i's) and the remaining part Φ_i is the variance specific to the ith variable (Φ_i's are also referred to as the *unique variances* or simply *uniquenesses*).

Formulation of the estimator: In decomposition (1), the assumption that the rank (r_1) of $\mathbf{\Sigma}_c$ is small compared to p is fairly stringent—see Guttman (1958); Shapiro (1982); ten Berge (1998) for a historical overview of the concept. In a classical paper of Guttman (1958), the author argued based on psychometric evidence that $\mathbf{\Sigma}_c$ is often found to have high algebraic rank. In psychometric case studies it is rather rare that the covariance structure can be *completely* explained by a *few* common factors corresponding to mental abilities—in fact there is evidence of at least hundreds of common factors being present with the number growing without an upper bound. Formally, this means that instead of assuming that $\mathbf{\Sigma}_c$ has *exactly* low-rank it is practical to assume that it can be well-approximated by a low-rank matrix, namely, $\mathbf{L}_1\mathbf{L}_1'$ with $\mathbf{L}_1 \in \mathbb{R}^{p\times r}$. More precisely, $\mathbf{L}_1\mathbf{L}_1'$ is the best rank-r approximation to $\mathbf{\Sigma}_c$ in the matrix q-norm (also known as the Schatten norm), as defined in (5), and $(\mathbf{\Sigma}_c - \mathbf{L}_1\mathbf{L}_1')$ is the *residual* component. Following psychometric terminology, $\mathbf{L}_1$ corresponds to the r *most significant* factors representative of mental abilities and the residual $\mathbf{\Sigma}_c - \mathbf{L}_1\mathbf{L}_1'$ corresponds to the remaining psychometric factors unexplained by $\mathbf{L}_1\mathbf{L}_1'$. Thus we can rewrite decomposition (1) as follows:

$$\mathbf{\Sigma} = \underbrace{\mathbf{L}_1\mathbf{L}_1'}_{:=\mathbf{\Theta}} + \underbrace{(\mathbf{\Sigma}_c - \mathbf{L}_1\mathbf{L}_1')}_{:=\mathcal{N}} + \mathbf{\Phi}, \tag{2}$$

where we use the notation $\mathbf{\Theta} = \mathbf{L}_1\mathbf{L}_1'$ and $\mathcal{N} = (\mathbf{\Sigma}_c - \mathbf{L}_1\mathbf{L}_1')$ with $\mathbf{\Theta} + \mathcal{N} = \mathbf{\Sigma}_c = \mathbf{\Sigma} - \mathbf{\Phi}$. Note that $\mathbf{\Theta}$ denotes the best rank-r approximation to $(\mathbf{\Sigma} - \mathbf{\Phi})$, with the residual component being $\mathcal{N} = \mathbf{\Sigma} - \mathbf{\Phi} - \mathbf{\Theta}$. Note that the entries in $\mathbf{\Phi}$ need to be non-negative[1] and $\mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}$. In fact, in the words of ten Berge (1998) (see p. 326)

1. Negative estimates of the diagonals of $\mathbf{\Phi}$ are unwanted since they correspond to variances, but some FA estimation procedures often lead to negative estimates of $\mathbf{\Phi}$—these are popularly known in the literature as Heywood cases and have invited a significant amount of discussion in the community.

©2017 Dimitris Bertsimas, Martin S. Copenhaver, and Rahul Mazumder.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-613.html.

"*... However, when* $\mathbf{\Sigma} - \mathbf{\Phi}$ *the covariance matrix for the common parts of the variables, would appear to be indefinite, that would be no less embarrassing than having a negative unique variance in* $\mathbf{\Phi}$*...*"

We further refer the reader to Mardia et al. (1979) discussing the importance of $\mathbf{\Sigma} - \mathbf{\Phi}$ being PSD.[2] We thus have the following natural structural constraints on the parameters:

$$\mathbf{\Theta} \succeq \mathbf{0}, \quad \mathbf{\Phi} = \text{diag}(\Phi, \ldots, \Phi_p) \succeq \mathbf{0} \quad \text{and} \quad \mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}. \tag{3}$$

Motivated by the above considerations, we present the following rank-constrained estimation problem for FA:

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{\Sigma} - (\mathbf{\Theta} + \mathbf{\Phi})\|_q^q \\ \text{s.t.} \quad & \text{rank}(\mathbf{\Theta}) \leq r \\ & \mathbf{\Theta} \succeq \mathbf{0} \\ & \mathbf{\Phi} = \text{diag}(\Phi_1, \ldots, \Phi_p) \succeq \mathbf{0} \\ & \mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}, \end{aligned} \tag{4}$$

where $\mathbf{\Theta} \in \mathbb{R}^{p \times p}$, $\mathbf{\Phi} \in \mathbb{R}^{p \times p}$ are the optimization variables, and for a real symmetric matrix $\mathbf{A}_{p \times p}$, its matrix q-norm, also known as the Schatten norm (or Schatten-von-Neumann norm) is defined as

$$\|\mathbf{A}\|_q := \left(\sum_{i=1}^{p} |\lambda_i(\mathbf{A})|^q \right)^{\frac{1}{q}}, \tag{5}$$

where $\lambda_i(\mathbf{A}),\ i = 1, \ldots, p$, are the (real) eigenvalues of $\mathbf{A}$.

Interpreting the estimator: The estimation criterion (4) seeks to *jointly* obtain the (low-rank) common factors and *uniquenesses* that best explain $\mathbf{\Sigma}$ in terms of minimizing the matrix q-norm of the error $\mathbf{\Sigma} - (\mathbf{\Theta} + \mathbf{\Phi})$ under the PSD constraints (3). Note that criterion (4) does not necessarily assume that $\mathbf{\Sigma}$ *exactly* decomposes into a low-rank PSD matrix and a non-negative diagonal matrix. Problem (4) enjoys curious similarities with Principal Component Analysis (PCA). In PCA, given a PSD matrix $\mathbf{\Sigma}$ the leading r principal component directions of $\mathbf{\Sigma}$ are obtained by minimizing $\|\mathbf{\Sigma} - \mathbf{\Theta}\|_q$ subject to $\mathbf{\Theta} \succeq \mathbf{0}$ and $\text{rank}(\mathbf{\Theta}) \leq r$. If the optimal solution $\mathbf{\Phi}$ to Problem (4) is *given*, Problem (4) is analogous to a rank-r PCA on the residual matrix $\mathbf{\Sigma} - \mathbf{\Phi}$—thus it is naturally desirable to have $\mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}$. In PCA one is interested in understanding the proportion of variance explained by the top-r principal component directions: $\sum_{i=1}^{r} \lambda_i(\mathbf{\Sigma}) / \sum_{i=1}^{p} \lambda_i(\mathbf{\Sigma})$. The denominator $\sum_{i=1}^{p} \lambda_i(\mathbf{\Sigma}) = \text{Tr}(\mathbf{\Sigma})$ accounts for the total variance explained by the covariance matrix $\mathbf{\Sigma}$. Analogously, the proportion of variance explained by $\widehat{\mathbf{\Theta}}_r$ (which denotes the best rank r approximation to $\mathbf{\Sigma} - \mathbf{\Phi}$) is given by $\sum_{i=1}^{r} \lambda_i(\mathbf{\Sigma} - \mathbf{\Phi}) / \sum_{i=1}^{p} \lambda_i(\mathbf{\Sigma} - \mathbf{\Phi})$—for this quantity to be interpretable it is imperative that $\mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}$. In the above argument, of course, we assumed that $\mathbf{\Phi}$ is given. In general, $\mathbf{\Phi}$ needs to be estimated: Problem (4) achieves this goal by *jointly* learning $\mathbf{\Phi}$ and $\mathbf{\Theta}$. We note that certain popular approaches of FA (see Sections 1.1 and 1.2) do not impose the PSD constraint $\mathbf{\Sigma} - \mathbf{\Phi}$ as a part of the estimation

2. However, the estimation method described in Mardia et al. (1979) does not guarantee that $\mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}$.

scheme—leading to indefinite $\mathbf{\Sigma} - \mathbf{\Phi}$—thereby rendering statistical interpretations troublesome. Our numerical evidence suggests that the quality of estimates of $\mathbf{\Theta}$ and $\mathbf{\Phi}$ obtained from Problem (4) outperform those obtained by other competing procedures which do not incorporate the PSD constraints into their estimation criteria.

Choice of r: In exploratory FA, it is standard to consider several choices of r and study the manner in which the proportion of variance explained by the common factors *saturates* with increasing r. We refer the reader to popularly used methods described in Anderson (2003); Bartholomew et al. (2011); Mardia et al. (1979) and more modern techniques (Bai and Ng, 2008, see also references therein) for the choice of r.

Estimate of Covariance Matrix: In the finite sample setting, we set $\mathbf{\Sigma}$ to be the sample covariance matrix. A *consequence* of solving Problem (4) is that we get an *estimate* for the covariance matrix given by $\hat{\mathbf{\Theta}} + \hat{\mathbf{\Phi}}$—in this sense, criterion (4) can be viewed as a regularization scheme: the rank constraint on $\mathbf{\Theta}$ encourages parsimony and the PSD constraints encourage interpretability, as discussed above.

In this paper, we propose a general computational framework to solve Problem (4) for any $q \geq 1$. The well-known Schatten q-norm appearing in the loss function is chosen for flexibility—it underlines the fact that our approach can be applied for *any* $q \geq 1$. Note that the estimation criterion (4) (even for the case $q = 1$) does not seem to appear in prior work on *approximate minimum rank Factor Analysis* (MRFA) (ten Berge and Kiers, 1991; Shapiro and ten Berge, 2002). However, we show in Proposition 1 that Problem (4) for the special case $q = 1$ turns out to be equivalent to MRFA. For $q = 2$, the loss function is the familiar squared Frobenius norm also used in MINRES, though the latter formulation is not equivalent to Problem (4), as explained in Section 1.1. We place more emphasis on studying the computational properties for the more common norms $q \in \{1, 2\}$.

The presence of the rank constraint in Problem (4) makes the optimization problem nonconvex. Globally optimizing Problem (4) or for that matter obtaining a good stationary point is quite challenging. We propose a new *equivalent* smooth formulation to Problem (4) which does not contain the combinatorial rank constraint. We employ simple and tractable sequential convex relaxation techniques with guaranteed convergence properties and excellent computational properties to obtain a stationary point for Problem (4). An important novelty of this paper is to present certifiable lower bounds on Problem (4) without resorting to structural assumptions, thus making it possible to solve Problem (4) to *provable* optimality. Towards this end we propose new methods and ideas that incorporate state-of-the-art developments in nonlinear and global optimization.[3]

1.1 A selective overview of related FA estimators

FA has a long and influential history which dates back more than a hundred years. The notion of FA possibly first appeared in Spearman (1904) for the one factor model, which was then generalized to the multiple factors model by various authors (see, for example,

3. The class of optimization problems studied in this paper involve global minimization of nonconvex, continuous semidefinite optimization problems. Computational methods for this class of problems are in a nascent stage; further, such methods are significantly less developed when compared to those for mixed integer linear optimization problems, thus posing a major challenge in this work.

Thurstone 1947). Significant contributions related to computational methods for FA have been nicely documented in Bai and Ng (2008); Harman and Jones (1966); Jöreskog (1967, 1978); Lawley and Maxwell (1962); Ledermann (1937); Rao (1973); Shapiro (1982); and ten Berge and Kiers (1991), among others.

We will briefly describe some widely used approaches for FA that are closely related to the approach pursued herein and also point out their connections.

Constrained Minimum Trace Factor Analysis (MTFA): This approach (ten Berge et al., 1981; Riccia and Shapiro, 1982) seeks to decompose $\mathbf{\Sigma}$ exactly into the sum of a diagonal matrix and a low-rank component, which are estimated via the following convex optimization problem:

$$\begin{array}{rl} \text{minimize} & \text{Tr}(\mathbf{\Theta}) \\ \text{s.t.} & \mathbf{\Theta} \succeq \mathbf{0} \\ & \mathbf{\Sigma} = \mathbf{\Theta} + \mathbf{\Phi} \\ & \mathbf{\Phi} = \text{diag}(\Phi_1, \ldots, \Phi_p) \succeq \mathbf{0}, \end{array} \tag{6}$$

with variables $\mathbf{\Theta}, \mathbf{\Phi}$. Because $\mathbf{\Theta}$ is PSD, $\text{Tr}(\mathbf{\Theta}) = \sum_{i=1}^{p} \lambda_i(\mathbf{\Theta})$ is a convex surrogate (Fazel, 2002) for the rank of $\mathbf{\Theta}$—Problem (6) may thus be viewed as a convexification of the rank minimization problem

$$\begin{array}{rl} \text{minimize} & \text{rank}(\mathbf{\Theta}) \\ \text{s.t.} & \mathbf{\Theta} \succeq \mathbf{0} \\ & \mathbf{\Sigma} = \mathbf{\Theta} + \mathbf{\Phi} \\ & \mathbf{\Phi} = \text{diag}(\Phi_1, \ldots, \Phi_p) \succeq \mathbf{0}. \end{array} \tag{7}$$

In general, Problems (6) and (7) are *not* equivalent. See Riccia and Shapiro (1982); Saunderson et al. (2012); Shapiro (1982) (and references therein) for further connections between the minimizers of (6) and (7).

A main difference between formulations (4) and (7) is that the former allows an error in the residual $(\mathbf{\Sigma} - \mathbf{\Theta} - \mathbf{\Phi})$ by constraining $\mathbf{\Theta}$ to have low-rank, unlike (7) which imposes a hard constraint $\mathbf{\Sigma} = \mathbf{\Theta} + \mathbf{\Phi}$. As noted earlier, this can be quite restrictive in various applications. Even if one views Problem (6) as imposing a less stringent requirement than that of Problem (7), we see two distinct advantages of Problem (4) over Problem (6): it offers the modeling flexibility of controlling the complexity, viz. rank, of solutions via the choice of r; and it provides smaller of estimates of rank for a comparable amount of explained variance, as substantiated by experimental findings presented in Section 5.

Approximate Minimum Rank Factor Analysis (MRFA): This method (see for example, ten Berge and Kiers, 1991; Shapiro and ten Berge, 2002) considers the following optimization problem:

$$\begin{array}{rl} \text{minimize} & \displaystyle\sum_{i=r+1}^{p} \lambda_i(\mathbf{\Sigma} - \mathbf{\Phi}) \\ \text{s.t.} & \mathbf{\Phi} = \text{diag}(\Phi_1, \ldots, \Phi_p) \succeq \mathbf{0} \\ & \mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}, \end{array} \tag{8}$$

where the optimization variable is $\mathbf{\Phi}$. Proposition 1 presented below establishes that Problem (8) is equivalent to the rank-constrained FA formulation (4) for the case $q = 1$. This connection does not appear to be formally established in ten Berge and Kiers (1991). We believe that criterion (4) for $q = 1$ is easier to interpret as an estimation criterion for FA models over (8). ten Berge and Kiers (1981, 1991) describe a method[4] for numerically optimizing (8)—as documented in the code for MRFA (ten Berge and Kiers, 2003), their implementation can handle problems of size $p \leq 20$.

Principal Component (PC) Factor Analysis: Principal Component factor analysis or PC in short (Connor and Korajczyk, 1986; Bai and Ng, 2008) implicitly assumes that $\mathbf{\Phi} = \sigma^2 \mathbf{I}_{p\times p}$, for some $\sigma^2 > 0$ and performs a low-rank PCA on $\mathbf{\Sigma}$. It is not clear how to estimate $\mathbf{\Phi}$ via this method such that $\Phi_i \geq 0$ and $\mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}$. Following Mardia et al. (1979), the $\mathbf{\Phi}$'s may be estimated after estimating $\widehat{\mathbf{\Theta}}$ via the update rule $\widehat{\mathbf{\Phi}} = \text{diag}(\mathbf{\Sigma} - \widehat{\mathbf{\Theta}})$—the estimates thus obtained, however, need not be non-negative. Furthermore, it is not guaranteed that the condition $\mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}$ is met.

Minimum Residual Factor Analysis (MINRES): This approach (Harman and Jones, 1966; Shapiro and ten Berge, 2002) considers the following optimization problem (with respect to the variable $\mathbf{L}_{p\times r}$):

$$\text{minimize} \quad \sum_{1\leq i\neq j\leq p} \left(\sigma_{ij} - (\mathbf{L}\mathbf{L}')_{ij}\right)^2, \tag{9}$$

where $\mathbf{\Sigma} := ((\sigma_{ij}))$, $\mathbf{\Theta} = \mathbf{L}\mathbf{L}'$ and the sum (in the objective) is taken over all the off-diagonal entries. Formulation (9) is equivalent to the nonconvex optimization problem

$$\begin{array}{rl} \text{minimize} & \|\mathbf{\Sigma} - (\mathbf{\Theta} + \mathbf{\Phi})\|_2^2 \\ \text{s.t.} & \text{rank}(\mathbf{\Theta}) \leq r \\ & \mathbf{\Theta} \succeq \mathbf{0} \\ & \mathbf{\Phi} = \text{diag}(\Phi_1, \ldots, \Phi_p), \end{array} \tag{10}$$

with variables $\mathbf{\Theta}$ and $\mathbf{\Phi}$, where Φ_i, $i \geq 1$, are unconstrained. If $\widehat{\mathbf{\Theta}}$ is a minimizer of Problem (10), then *any* $\widehat{\mathbf{L}}$ satisfying $\widehat{\mathbf{L}}\widehat{\mathbf{L}}' = \widehat{\mathbf{\Theta}}$ minimizes (9) and vice-versa.

Various heuristic approaches are used to for Problem (9). The `R` package `psych`, for example, uses a black box gradient based tool `optim` to minimize the nonconvex Problem (9) with respect to $\mathbf{L}$. Once $\widehat{\mathbf{L}}$ is estimated, the diagonal entries of $\mathbf{\Phi}$ are estimated as $\widehat{\Phi}_i = \sigma_{ii} - (\widehat{\mathbf{L}}\widehat{\mathbf{L}}')_{ii}$, for $i \geq 1$. Note that $\widehat{\Phi}_i$ obtained in this fashion may be negative[5] and the condition $\mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}$ may be violated.

Generalized Least Squares, Principal Axis and variants: The Ordinary Least Squares (OLS) method for FA (Bartholomew et al., 2011) considers formulation (10) with the additional constraint that $\Phi_i \geq 0\ \forall i$. The Weighted Least Squares (WLS) or the generalized least squares method (see for example, Bartholomew et al., 2011) considers a weighted least squares objective:

$$\|\mathbf{W}\left(\mathbf{\Sigma} - (\mathbf{\Theta} + \mathbf{\Phi})\right)\|_2^2 .$$

4. The method is similar to Algorithm 1 presented herein for the case of $q = 1$; however, ten Berge and Kiers (1981, 1991) rely on a heuristic procedure, as described in Bentler and Woodward (1980), for the subproblem with respect to $\mathbf{\Phi}$.
5. If $\widehat{\Phi}_i < 0$, some ad-hoc procedure is used to threshold it to a nonnegative quantity.

As in the ordinary least squares case, here too we assume that $\Phi_i \geq 0$. Various choices of $\mathbf{W}$ are possible depending upon the application, with $\mathbf{W} \in \{\mathbf{\Sigma}^{-1}, \mathbf{\Phi}^{-1}\}$ being a couple of popular choices.

The Principal Axis (PA) FA method (Bartholomew et al., 2011; Revelle, 2015) is popularly used to estimate factor model parameters based on criterion (10) along with the constraints $\Phi_i \geq 0 \ \forall i$. This method starts with a nonnegative estimate $\widehat{\mathbf{\Phi}}$ and performs a rank r eigendecomposition on $\mathbf{\Sigma} - \widehat{\mathbf{\Phi}}$ to obtain $\widehat{\mathbf{\Theta}}$. The matrix $\widehat{\mathbf{\Phi}}$ is then updated to match the diagonal entries of $\mathbf{\Sigma} - \widehat{\mathbf{\Theta}}$, and the above steps are repeated until the estimate $\widehat{\mathbf{\Phi}}$ stabilizes.[6] Note that in this procedure the estimate $\widehat{\mathbf{\Theta}}$ may fail to be PSD and the entries of $\widehat{\Phi}_i$ may be negative as well. Heuristic restarts and various initializations are often carried out to arrive at a reasonable solution (see for example discussions in Bartholomew et al. 2011).

In summary, the least squares stylized methods described above may lead to estimates that violate one or more of the constraints: $\mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}$, $\mathbf{\Theta} \succeq \mathbf{0}$ and $\mathbf{\Phi} \succeq \mathbf{0}$.

Maximum Likelihood for Factor Analysis: This approach (Bai and Li, 2012; Jöreskog, 1967; Mardia et al., 1979; Roweis and Ghahramani, 1999; Rubin and Thayer, 1982) is another widely used method in FA and typically assumes that the data follows a multivariate Gaussian distribution. This procedure maximizes a likelihood function and is quite different from the loss functions pursued in this paper and discussed above. The estimator need not exist for any $\mathbf{\Sigma}$—see for example Robertson and Symons (2007).

Most of the methods described in Section 1.1 are widely used and their implementations are available in statistical packages `psych` (Revelle, 2015), `nFactors` (Raiche and Magis, 2011), `GPArotation` (Bernaards and Jennrich, 2005), and others and are publicly available from `CRAN`.[7]

1.2 Broad categories of factor analysis estimators

A careful investigation of the methods described above suggests that they can be divided into two broad categories. Some of the above estimators explicitly incorporate a PSD structural assumption on the residual covariance matrix $\mathbf{\Sigma} - \mathbf{\Phi}$ in *addition* to requiring $\mathbf{\Theta} \succeq \mathbf{0}$ and $\mathbf{\Phi} \succeq \mathbf{0}$ while the others do not. As already pointed out, these constraints are important for statistical interpretability. We propose to distinguish between the following two broad categories of FA algorithms:

(A) This category is comprised of FA estimation procedures cast as nonlinear Semidefinite Optimization (SDO) problems—estimation takes place in the presence of constraints of the form $\mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}$, along with $\mathbf{\Theta} \succeq \mathbf{0}$ and $\mathbf{\Phi} \succeq \mathbf{0}$. Members of this category are MRFA, MTFA and more generally Problem (4).

Existing approaches for these problems are typically not scalable: for example, we are not aware of any algorithm (prior to this paper) for Problem (8) (MRFA) that scales to covariance matrices with p larger than thirty. Indeed, while theoretical guarantees of optimality exist in certain cases (Saunderson et al., 2012), the conditions required for such results to hold are generally difficult to verify in practice.

(B) This category includes classical FA methods which are not based on nonlinear SDO based formulations (as in Category (A)). MINRES, OLS, WLS, GLS, PC and PA based FA estimation procedures (as described in Section 1.1) belong to this category. These methods are generally scalable to problem sizes where p is of the order of a few thousand—significantly larger than most procedures belonging to Category (A)—and are implemented in open-source `R`-packages.

Contributions: Our contributions in this paper may be summarized as follows:

1. We consider a flexible family of FA estimators which can be obtained as solutions to rank-constrained nonlinear SDO problems. In particular, our framework provides a unifying perspective on several existing FA estimation approaches.

2. We propose a novel *exact* reformulation of the rank-constrained FA problem (4) as a smooth optimization problem with convex compact constraints. We also develop a unified algorithmic framework utilizing modern optimization techniques to obtain high quality solutions to Problem (4). Our algorithms, at every iteration, simply require computing a low-rank eigendecomposition of a $p \times p$ matrix and a structured scalable convex SDO. Our proposal is capable of solving FA problems involving covariance matrices having dimensions up to a few thousand, thereby making it on par with the most scalable FA methods used currently.[8]

3. Our SDO formulation enables us to estimate the underlying factors and unique variances under the restriction that the residual covariance matrix is PSD—a characteristic that is absent in several popularly used FA methods. This aids statistical interpretability, especially in drawing parallels with PCA and understanding the proportion of variance explained by a given number of factors. Methods proposed herein produce superior quality estimates, in terms of various performance metrics, when compared to existing FA approaches. To our knowledge, this is the first paper demonstrating that certifiably optimal solutions to a rank-constrained problem can be found for problems of realistic sizes, without making any assumptions on the underlying data.

4. Using techniques from discrete and global optimization, we develop a branch-and-bound algorithm which proves that the low-rank solutions found are often optimal in seconds for problems on the order of $p = 10$ variables, in minutes for problems on the order of $p = 100$, and in days for some problems on the order of $p = 4000$. As the selected rank increases, so too does the computational burden of proving optimality. It is particularly crucial to note that the optimal solutions for all problems we consider are found very quickly, and that vast majority of computational time is then spent on proving optimality. Hence, for a practitioner who is not particularly concerned with certifying optimality, our techniques for finding feasible solutions provide high-quality estimates quickly.

6. We are not aware of a proof, however, showing the convergence of this procedure.
7. `http://cran.us.r-project.org`

8. An implementation of our approach is available at `https://github.com/copenhaver/factoranalysis`.

5. We provide computational evidence demonstrating the favorable performance of our proposed method. Finally, to the best of our knowledge this is the first paper that views various FA methods in a unified fashion via a modern optimization lens and attempts to compare a wide range of FA techniques in large scale.

Structure of the paper: The paper is organized as follows. In Section 1 we propose a flexible family of optimization Problems (4) for the task of statistical estimation in FA models. Section 2 presents an exact reformulation of Problem (4) as a nonlinear SDO without the rank constraint. Section 3 describes the use of nonlinear optimization techniques such as the Conditional Gradient (CG) method (Bertsekas, 1999) adapted to provide feasible solutions (upper bounds) to our formulation. First order methods employed to compute the convex SDO subproblems are also described in the same section. In Section 4, we describe our method for certifying optimality of the solutions from Section 3 in the case when $q = 1$. In Section 5, we present computational results demonstrating the effectiveness of our proposed method in terms of (a) modeling flexibility in the choice of the number of factors r and the parameter q, (b) scalability, and (c) the quality of solutions obtained in a wide array of real and synthetic datasets—comparisons with several existing methods for FA are considered. Section 6 contains our conclusions.

2. Reformulations of Problem (4)

Let $\boldsymbol{\lambda}(\mathbf{A})$ denote the vector of eigenvalues of $\mathbf{A}$, arranged in decreasing order, i.e.,

$$\lambda_1(\mathbf{A}) \geq \lambda_2(\mathbf{A}) \geq \ldots \geq \lambda_p(\mathbf{A}). \tag{11}$$

The following proposition presents the first reformulation of Problem (4) as a continuous eigenvalue optimization problem with convex compact constraints. Proofs of all results can be found in Appendix A.

Proposition 1 (a) *For any $q \geq 1$, Problem (4) is equivalent to:*

$$\begin{array}{lll} (\text{CFA}_q) & \text{minimize} & f_q(\boldsymbol{\Phi};\boldsymbol{\Sigma}) := \displaystyle\sum_{i=r+1}^{p} \lambda_i^q(\boldsymbol{\Sigma}-\boldsymbol{\Phi}) \\ & \text{s.t.} & \boldsymbol{\Phi} = \text{diag}(\Phi_1,\ldots,\Phi_p) \succeq \mathbf{0} \\ & & \boldsymbol{\Sigma}-\boldsymbol{\Phi} \succeq \mathbf{0}, \end{array} \tag{12}$$

where $\boldsymbol{\Phi}$ is the optimization variable.

(b) *Suppose $\boldsymbol{\Phi}^*$ is a minimizer of Problem (12), and let*

$$\boldsymbol{\Theta}^* = \mathbf{U}\ \text{diag}\left(\lambda_1(\boldsymbol{\Sigma}-\boldsymbol{\Phi}^*),\ldots,\lambda_r(\boldsymbol{\Sigma}-\boldsymbol{\Phi}^*),0,\ldots,0\right)\mathbf{U}', \tag{13}$$

where $\mathbf{U}_{p\times p}$ is the matrix of eigenvectors of $\boldsymbol{\Sigma}-\boldsymbol{\Phi}^*$. Then $(\boldsymbol{\Theta}^*,\boldsymbol{\Phi}^*)$ is a solution to Problem (4).

Problem (12) is a nonlinear SDO in $\boldsymbol{\Phi}$, unlike the original formulation (4) that estimates $\boldsymbol{\Theta}$ and $\boldsymbol{\Phi}$ jointly. Note that the rank constraint does not appear in Problem (12) and the constraint set of Problem (12) is convex and compact. However, Problem (12) is nonconvex due to the nonconvex objective function $\sum_{i=r+1}^{p}\lambda_i^q(\boldsymbol{\Sigma}-\boldsymbol{\Phi})$. For $q = 1$ the function appearing in the objective of (12) is concave and for $q > 1$, it is neither convex nor concave.

Proposition 2 *The estimation Problem (4) is equivalent to*

$$\begin{array}{lll} \text{minimize} & \|\boldsymbol{\Sigma}-(\boldsymbol{\Theta}+\boldsymbol{\Phi})\|_q^q \\ \text{s.t.} & \text{rank}(\boldsymbol{\Theta}) \leq r \\ & \boldsymbol{\Theta} \succeq \mathbf{0} \\ & \boldsymbol{\Phi} = \text{diag}(\Phi_1,\ldots,\Phi_p) \succeq \mathbf{0} \\ & \boldsymbol{\Sigma}-\boldsymbol{\Phi} \succeq \mathbf{0} \\ & \boldsymbol{\Sigma}-\boldsymbol{\Theta} \succeq \mathbf{0}. \end{array} \tag{14}$$

Note that Problem (14) has an additional PSD constraint $\boldsymbol{\Sigma}-\boldsymbol{\Theta} \succeq \mathbf{0}$ which does not explicitly appear in Problem (4). It is interesting to note that the two problems are equivalent. By virtue of Proposition 2, Problem (14) can as well be used as the estimation criterion for rank constrained FA. However, we will work with formulation (4) because it is easier to interpret from a statistical perspecitve.

Special instances of (CFA_q): We show that some well-known FA estimation problems can be viewed as special cases of our general framework.

For $q = 1$, Problem (12) reduces to MRFA, as described in (8). For $q = 1$ and $r = 0$, Problem (12) reduces to MTFA (6). When $q = 2$, we get a variant of (10), i.e., a PSD *constrained* analogue of MINRES

$$\begin{array}{lll} \text{minimize} & \displaystyle\sum_{i=r+1}^{p} \lambda_i^2(\boldsymbol{\Sigma}-\boldsymbol{\Phi}) \\ \text{s.t.} & \boldsymbol{\Phi} = \text{diag}(\Phi_1,\ldots,\Phi_p) \succeq \mathbf{0} \\ & \boldsymbol{\Sigma}-\boldsymbol{\Phi} \succeq \mathbf{0}. \end{array} \tag{15}$$

Note that unlike Problem (10), Problem (15) explicitly imposes PSD constraints on $\boldsymbol{\Phi}$ and $\boldsymbol{\Sigma}-\boldsymbol{\Phi}$. The objective function in (12) is continuous but non-smooth. The function is differentiable at $\boldsymbol{\Phi}$ if and only if the r and $(r+1)$th eigenvalues of $\boldsymbol{\Sigma}-\boldsymbol{\Phi}$ are distinct (Lewis, 1996; Shapiro and ten Berge, 2002), i.e., $\lambda_{r+1}(\boldsymbol{\Sigma}-\boldsymbol{\Phi}) < \lambda_r(\boldsymbol{\Sigma}-\boldsymbol{\Phi})$. The non-smoothness of the objective function in Problem (12) makes the use of standard gradient based methods problematic (Bertsekas, 1999). Theorem 1 presents a reformulation of Problem (12) in which the objective function is continuously differentiable.

Theorem 1 (a) *The estimation criterion given by Problem (4) is equivalent to*[9]

$$\begin{array}{lll} \text{minimize} & g_q(\mathbf{W},\boldsymbol{\Phi}) := & \text{Tr}(\mathbf{W}(\boldsymbol{\Sigma}-\boldsymbol{\Phi})^q) \\ \text{s.t.} & & \boldsymbol{\Phi} = \text{diag}(\Phi_1,\ldots,\Phi_p) \succeq \mathbf{0} \\ & & \boldsymbol{\Sigma}-\boldsymbol{\Phi} \succeq \mathbf{0} \\ & & \mathbf{I} \succeq \mathbf{W} \succeq \mathbf{0} \\ & & \text{Tr}(\mathbf{W}) = p-r. \end{array} \tag{16}$$

(b) *The solution $\widehat{\boldsymbol{\Theta}}$ of Problem (4) can be recovered from the solution $\widehat{\mathbf{W}},\widehat{\boldsymbol{\Phi}}$ of Problem (16) via:*

$$\widehat{\boldsymbol{\Theta}} := \widehat{\mathbf{U}}\,\text{diag}(\widehat{\lambda}_1,\ldots,\widehat{\lambda}_r,0,\ldots,0)\widehat{\mathbf{U}}', \tag{17}$$

9. For any PSD matrix $\mathbf{A}$, with eigendecomposition $\mathbf{A} = \mathbf{U}_A\,\text{diag}(\lambda_1,\ldots,\lambda_p)\mathbf{U}_A'$, we define $\mathbf{A}^q := \mathbf{U}_A\,\text{diag}(\lambda_1^q,\ldots,\lambda_p^q)\mathbf{U}_A'$, for any $q \geq 1$.

where $\widehat{\mathbf{U}}$ is the matrix formed by the p eigenvectors corresponding to the eigenvalues $\widehat{\lambda}_1, \ldots, \widehat{\lambda}_p$ (arranged in decreasing order) of the matrix $\mathbf{\Sigma} - \widehat{\mathbf{\Phi}}$. Given $\widehat{\mathbf{\Phi}}$, any solution $\widehat{\mathbf{\Theta}}$ (given by (17)) is independent of q.

In Problem (16), if we partially minimize the function $g_q(\mathbf{W}, \mathbf{\Phi})$ over $\mathbf{\Phi}$ (with fixed $\mathbf{W}$), the resulting function is concave in $\mathbf{W}$. This observation leads to the following proposition.

Proposition 3 *The function $G_q(\mathbf{W})$ obtained upon (partially) minimizing $g_q(\mathbf{W}, \mathbf{\Phi})$ over $\mathbf{\Phi}$ (with $\mathbf{W}$ fixed) in Problem (16), given by*

$$\begin{array}{rll} G_q(\mathbf{W}) := & \inf\limits_{\mathbf{\Phi}} & g_q(\mathbf{W}, \mathbf{\Phi}) \\ & \text{s.t.} & \mathbf{\Phi} = \text{diag}(\Phi_1, \ldots, \Phi_p) \succeq \mathbf{0} \\ & & \mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}, \end{array} \tag{18}$$

is concave in $\mathbf{W}$. The sub-gradients of the function $G_q(\mathbf{W})$ exist and are given by:

$$\nabla G_q(\mathbf{W}) = (\mathbf{\Sigma} - \widehat{\mathbf{\Phi}}(\mathbf{W}))^q, \tag{19}$$

where $\widehat{\mathbf{\Phi}}(\mathbf{W})$ is a minimizer of the convex optimization Problem (18).

In light of Proposition 3, we present another reformulation of Problem (4) as the following concave minimization problem:

$$\begin{array}{rl} \text{minimize} & G_q(\mathbf{W}) \\ \text{s.t.} & \mathbf{I} \succeq \mathbf{W} \succeq \mathbf{0} \\ & \text{Tr}(\mathbf{W}) = p - r, \end{array} \tag{20}$$

where the function $G_q(\mathbf{W})$ is differentiable if and only if $\widehat{\mathbf{\Phi}}(\mathbf{W})$ is unique.

Note that by virtue of Proposition 1, Problems (12) and (20) are equivalent. Therefore, it is natural to ask whether one formulation might be favored over the other from a computational perspective. Towards this end, note that both Problems (12) and (20) involve the minimization of a non-smooth objective function, over convex compact constraints. However, the objective function of Problem (12) is nonconvex (for $q > 1$) whereas the one in Problem (20) is concave (for all $q \geq 1$). We will see in Section 3 that CG-based algorithms can be applied to a concave minimization problem (even if the objective function is not differentiable); however, CG applied to general non-smooth objective functions has limited convergence guarantees. Thus, formulation (20) is readily amenable to CG-based optimization algorithms, unlike formulation (12). This seems to make Problem (20) computationally more appealing than Problem (12).

3. Finding Upper Bounds

This section presents a unified computational framework for the class of problems (CFA_q). Problem (16) is a nonconvex smooth optimization problem and obtaining a stationary point is quite challenging. We propose iterative schemes based on the Conditional Gradient (CG) algorithm (Bertsekas, 1999)—a generalization of the Frank-Wolfe algorithm (Frank

and Wolfe, 1956)—to obtain a stationary point of the problem. The appealing aspect of our framework is that every iteration of the algorithm requires solving a convex SDO problem which is computationally tractable. While off-the-shelf interior point algorithms—for example `SDPT3` (Toh et al., 1999), `Yalmip` (Löfberg, 2004), and `MOSEK` (Andersen and Andersen, 2000)—can be used to solve the convex SDO problems, they typically do not scale well for large problems due to intensive memory requirements. In this vein, first order algorithms have received a lot of attention (Nesterov, 2004, 2005, 2007) in convex optimization of late, due to their low cost per iteration, low-memory requirements, and ability to deliver solutions of moderate accuracy for large problems within a modest time limit. We use first order methods to solve the convex SDO problems. We present one primary scheme based on the CG algorithm:

> **Algorithm 1:** This scheme, described in Section 3.1, applies CG on the optimization Problem (20), where the function $G_q(\mathbf{W})$ defined in (18) is concave (and possibly non-smooth).

In addition, in Appendix B we present an alternative approach that applies CG to Problem (16), where the objective function $g_q(\mathbf{W}, \mathbf{\Phi})$ is smooth.

To make notation simpler, we will use the following shorthand:

$$\mathbf{\Psi}_{p,p-r} := \{\mathbf{W} \in \mathbb{R}^{p \times p} : \mathbf{I} \succeq \mathbf{W} \succeq \mathbf{0}, \;\; \text{Tr}(\mathbf{W}) = p - r\} \tag{21}$$

and $\mathcal{F}_{\mathbf{\Sigma}} = \{\mathbf{\Phi} : \mathbf{\Sigma} - \mathbf{\Phi} \succcurlyeq \mathbf{0}, \;\; \mathbf{\Phi} = \text{diag}(\Phi_1, \ldots, \Phi_p) \succcurlyeq \mathbf{0}\}$.

3.1 A CG based algorithm for Problem (20)

The CG method for Problem (20) requires solving a linearization of the concave objective function. At iteration k, if $\mathbf{W}^{(k)}$ is the current estimate of $\mathbf{W}$, the new estimate $\mathbf{W}^{(k+1)}$ is obtained by

$$\mathbf{W}^{(k+1)} \in \underset{\mathbf{W} \in \mathbf{\Psi}_{p,p-r}}{\arg\min} \left\langle \nabla G_q(\mathbf{W}^{(k)}), \mathbf{W} \right\rangle = \underset{\mathbf{W} \in \mathbf{\Psi}_{p,p-r}}{\arg\min} \; \text{Tr}(\mathbf{W}(\mathbf{\Sigma} - \mathbf{\Phi}^{(k)})^q), \tag{22}$$

where by Proposition 3, $(\mathbf{\Sigma} - \mathbf{\Phi}^{(k)})^q$ is a sub-gradient of $G_q(\mathbf{W})$ at $\mathbf{W}^{(k)}$ with $\mathbf{\Phi}^{(k)}$ given by

$$\mathbf{\Phi}^{(k)} \in \underset{\mathbf{\Phi} \in \mathcal{F}_{\mathbf{\Sigma}}}{\arg\min} \; \text{Tr}(\mathbf{W}^{(k)}(\mathbf{\Sigma} - \mathbf{\Phi})^q) \tag{23}$$

No explicit line search is necessary here because the minimum will always be at the new point, i.e., $\mathbf{\Phi}^{(k)}$, due to the concavity of the objective function. The sequence $\mathbf{W}^{(k)}$ is recursively computed via (22) until the convergence criterion

$$G_q(\mathbf{W}^{(k)}) - G_q(\mathbf{W}^{(k+1)}) \leq \text{TOL} \cdot G_q(\mathbf{W}^{(k)}), \tag{24}$$

for some user-defined tolerance, TOL > 0, is met. A short description of the procedure appears in Algorithm 1.

Before we present the convergence rate of Algorithm 1, we will need to introduce some notation. For any point $\overline{\mathbf{W}}$ belonging to the feasible set of Problem (20) let us define $\Delta(\overline{\mathbf{W}})$ as follows:

$$\Delta(\overline{\mathbf{W}}) := \inf_{\mathbf{W} \in \mathbf{\Psi}_{p,p-r}} \langle \nabla G_q(\overline{\mathbf{W}}), \mathbf{W} - \overline{\mathbf{W}} \rangle. \tag{25}$$

Algorithm 1 A CG based algorithm for formulation (20)

1. Initialize with $\mathbf{W}^{(1)}$ ($k=1$), feasible for Problem (20) and repeat, for $k \geq 2$, Steps 2-3 until convergence criterion (24) is satisfied.
2. Update $\mathbf{\Phi}$ (with $\mathbf{W}$ fixed) by solving (23).
3. Update $\mathbf{W}$ (with $\mathbf{\Phi}$ fixed) by solving (22), to get $\mathbf{W}^{(k+1)}$.

Further, $\mathbf{W}^*$ satisfies the first order stationary condition for Problem (20) if $\mathbf{W}^*$ is feasible for the problem and $\Delta(\mathbf{W}^*) \geq 0$.

We now present Theorem 2 establishing the rate of convergence and associated convergence properties of Algorithm 1. The proof (along with all other omitted proofs) is contained in Appendix A.

Theorem 2 *If $\mathbf{W}^{(k)}$ is a sequence produced by Algorithm 1, then $G_q(\mathbf{W}^{(k)})$ is a monotone decreasing sequence and every limit point $\mathbf{W}^{(\infty)}$ of the sequence $\mathbf{W}^{(k)}$ is a stationary point of Problem* (20)*. Furthermore, Algorithm 1 has a convergence rate of $O(1/K)$ (with K denoting the iteration index) to a first order stationary point of Problem* (20)*, i.e.,*

$$\min_{i=1,\ldots,K} \left\{-\Delta(\mathbf{W}^{(i)})\right\} \leq \frac{G_q(\mathbf{W}^{(1)}) - G_q(\mathbf{W}^{(\infty)})}{K}. \tag{26}$$

3.2 Solving the convex SDO problems

Algorithm 1 requires sequentially solving convex SDO problems in $\mathbf{W}$ and $\mathbf{\Phi}$. We describe herein how these subproblems can be solved efficiently.

3.2.1 Solving the SDO problem with respect to $\mathbf{W}$

A generic SDO problem associated with Problem (22) requires updating $\mathbf{W}$ as

$$\widehat{\mathbf{W}} \in \underset{\mathbf{W} \in \mathbf{\Psi}_{p,p-r}}{\arg\min} \langle \mathbf{W}, \widetilde{\mathbf{W}} \rangle, \tag{27}$$

for some fixed symmetric $\widetilde{\mathbf{W}}_{p\times p}$, depending upon the algorithm and the choice of q, described as follows. For Algorithm 1 the update in $\mathbf{W}$ at iteration k for Problem (22), corresponds to $\widetilde{\mathbf{W}} = (\mathbf{\Sigma} - \mathbf{\Phi}^{(k+1)})^q$.

A solution to Problem (27) is given by $\widehat{\mathbf{W}} = \sum_{i=r+1}^{p} \mathbf{u}_i \mathbf{u}_i'$, where $\mathbf{u}_1, \ldots, \mathbf{u}_p$ are the eigenvectors of the matrix $\widetilde{\mathbf{W}}$, corresponding to the eigenvalues $\lambda_1(\widetilde{\mathbf{W}}), \ldots, \lambda_p(\widetilde{\mathbf{W}})$.

3.2.2 Solving the SDO problem with respect to $\mathbf{\Phi}$

The SDO problem arising from the update of $\mathbf{\Phi}$ is not as straightforward as the update with respect to $\mathbf{W}$. Before presenting the general case, it helps to consider a few special cases of (CFA$_q$). For $q=1$ the objective function of Problem (16)

$$g_1(\mathbf{W}, \mathbf{\Phi}) = \langle \mathbf{W}, \mathbf{\Sigma} \rangle - \sum_{i=1}^{p} w_{ii} \Phi_i \tag{28}$$

is linear in $\mathbf{\Phi}$ (for fixed $\mathbf{W}$). For $q=2$, the objective function of Problem (16)

$$g_2(\mathbf{W}, \mathbf{\Phi}) = \mathrm{Tr}(\mathbf{W}\mathbf{\Sigma}^2) + \sum_{i=1}^{p} (w_{ii}\Phi_i^2 - 2\langle \mathbf{w}_i, \boldsymbol{\sigma}_i \rangle \Phi_i) \tag{29}$$

is a convex quadratic in $\mathbf{\Phi}$ (for fixed $\mathbf{W}$).

For Algorithm 1, the partial minimizations with respect to $\mathbf{\Phi}$, for $q=1$ and $q=2$, require minimizing Problems (28) and (29), respectively.

Various instances of optimization problems with respect to $\mathbf{\Phi}$ such as those appearing in Algorithm 1 can be viewed as special cases of the following family of SDO problems:

$$\underset{\mathbf{\Phi} \in \mathcal{F}_{\mathbf{\Sigma}}}{\text{minimize}} \quad \sum_{i=1}^{p} \left(c_i \Phi_i^2 + d_i \Phi_i\right) \tag{30}$$

where $c_i \geq 0$ and d_i for $i = 1, \ldots, p$ are problem parameters that depend upon the choice of algorithm and q. We now present a first order convex optimization scheme for solving (30).

A First Order Scheme for Problem (30)

With the intention of providing a simple and scalable algorithm for the convex SDO problem, we use the Alternating Direction Method of Multipliers (Bertsekas, 1999; Boyd et al., 2011) (ADMM). We introduce a splitting variable $\mathbf{\Lambda} = \mathbf{\Sigma} - \mathbf{\Phi}$ and rewrite Problem (30) in the following equivalent form:

$$\begin{aligned} \underset{\mathbf{\Phi}, \mathbf{\Lambda}}{\text{minimize}} \quad & \sum_{i=1}^{p} (c_i \Phi_i^2 + d_i \Phi_i) \\ \text{s.t.} \quad & \mathbf{\Phi} = \mathrm{diag}(\Phi_1, \ldots, \Phi_p) \succeq \mathbf{0} \\ & \mathbf{\Lambda} \succeq \mathbf{0} \\ & \mathbf{\Lambda} = \mathbf{\Sigma} - \mathbf{\Phi}. \end{aligned} \tag{31}$$

The Augmented Lagrangian for the above problem is:

$$\mathcal{L}_\rho(\mathbf{\Phi}, \mathbf{\Lambda}, \boldsymbol{\nu}) := \sum_{i=1}^{p} (c_i \Phi_i^2 + d_i \Phi_i) + \left\langle \boldsymbol{\nu}, \mathbf{\Lambda} - (\mathbf{\Sigma} - \mathbf{\Phi}) \right\rangle + \frac{\rho}{2} \|\mathbf{\Lambda} - (\mathbf{\Sigma} - \mathbf{\Phi})\|_2^2, \tag{32}$$

where $\rho > 0$ is a scalar and $\langle \cdot, \cdot \rangle$ denotes the standard trace inner product. ADMM involves the following three updates:

$$\mathbf{\Phi}^{(k+1)} \in \underset{\mathbf{\Phi} = \mathrm{diag}(\Phi_1, \ldots, \Phi_p) \succeq \mathbf{0}}{\arg\min} \quad \mathcal{L}_\rho(\mathbf{\Phi}, \mathbf{\Lambda}^{(k)}, \boldsymbol{\nu}^{(k)}), \tag{33}$$

$$\mathbf{\Lambda}^{(k+1)} \in \underset{\mathbf{\Lambda} \succeq \mathbf{0}}{\arg\min} \quad \mathcal{L}_\rho(\mathbf{\Phi}^{(k+1)}, \mathbf{\Lambda}, \boldsymbol{\nu}^{(k)}), \tag{34}$$

$$\boldsymbol{\nu}^{(k+1)} = \boldsymbol{\nu}^{(k)} + \rho(\mathbf{\Lambda}^{(k+1)} - (\mathbf{\Sigma} - \mathbf{\Phi}^{(k+1)})), \tag{35}$$

and produces a sequence $\{(\mathbf{\Phi}^{(k)}, \boldsymbol{\lambda}^{(k)}, \boldsymbol{\nu}^{(k)})\}$, $k \geq 1$ — the convergence properties of the algorithm are quite well known (Boyd et al., 2011).

Problem (33) can be solved in closed form as

$$\Phi_i^{(k+1)} = \frac{\rho}{\rho + 2c_i} \max\left\{ (\sigma_{ii} - \lambda_{ii}^{(k)}) - \frac{(d_i + \nu_{ii}^{(k)})}{\rho}, 0 \right\}, \quad i = 1, \ldots, p. \tag{36}$$

The update with respect to $\mathbf{\Lambda}$ in (34) requires an eigendecomposition:

$$\begin{aligned} \mathbf{\Lambda}^{(k+1)} =& \arg\min_{\mathbf{\Lambda} \succeq \mathbf{0}} \left\| \mathbf{\Lambda} - (\mathbf{\Sigma} - \mathbf{\Phi}^{(k+1)} - \frac{1}{\rho}\boldsymbol{\nu}^{(k)}) \right\|_2^2 \\ =& \mathcal{P}_{S_p^+}\left(\mathbf{\Sigma} - \mathbf{\Phi}^{(k+1)} - \frac{1}{\rho}\boldsymbol{\nu}^{(k)} \right), \end{aligned} \tag{37}$$

where, the operator $\mathcal{P}_{S_p^+}(\mathbf{A})$ denotes the projection of a symmetric matrix $\mathbf{A}$ onto the cone of PSD matrices of dimension $p \times p$: $\mathcal{P}_{S_p^+}(\mathbf{A}) = \mathbf{U}_A \operatorname{diag}\left(\max\{\lambda_1, 0\}, \ldots, \max\{\lambda_p, 0\} \right) \mathbf{U}_A'$, where, $\mathbf{A} = \mathbf{U}_A \operatorname{diag}(\lambda_1, \ldots, \lambda_p) \mathbf{U}_A'$ is the eigendecomposition of $\mathbf{A}$.

Stopping criterion: The ADMM iterations (33)—(35) are continued till the values of $\|\mathbf{\Lambda}^{(k+1)} - (\mathbf{\Sigma} - \mathbf{\Phi}^{(k+1)})\|_2$ and the relative change in the objective values of Problem (30) become smaller than a certain threshold, say, TOL $\times \alpha$, where $\alpha \in \{10^{-1}, \ldots, 10^{-3}\}$—this is typically taken to be smaller than the convergence threshold for the CG iterations (TOL).

Computational cost of Problem (30): The most intensive computational stage in the ADMM procedure is in performing the projection operation (37); this requires $O(p^3)$ operations due to the associated eigendecomposition. This needs to be done for as many ADMM steps, until convergence.

Since Problem (30) is embedded inside iterative procedures like Algorithm 1, the estimates of $(\mathbf{\Phi}, \mathbf{\Lambda}, \boldsymbol{\nu})$ obtained by solving Problem (30) for a iteration index (of the CG algorithm) provides a good warm-start for the Problem (30) in the subsequent CG iteration. This is often found to decrease the number of iterations required by the ADMM algorithm to converge to a prescribed level of accuracy.[10]

3.3 Computational cost of Algorithm 1

For Algorithm 1 and other CG-based algorithms for factor analysis (see Appendix B), the computational bottleneck is in performing the eigendecomposition of a $p \times p$ matrix: the $\mathbf{W}$ update requires performing a low-rank eigendecomposition of a $p \times p$ matrix and the $\mathbf{\Phi}$ update requires solving a problem of the form (30), which also costs $O(p^3)$. Since eigendecompositions can easily be done for p of the order of a few thousands, the proposed algorithms can be applied to that scale.

Note that most existing popular algorithms for FA belonging to Category (B) (see Section 1.2) also perform an eigendecomposition with cost $O(p^3)$. Thus it appears that

10. The utility of warm starts is another compelling reason to apply a first-order-based approach instead of interior point methods. Indeed, warm starts are well-known to perform quite poorly when incorporated into interior point methods—often they can perform worse than cold starts (Yildirim and Wright, 2002; John and Yildirim, 2008). Given the need to repeatedly solve similarly structured SDOs for both upper bounds (as presented in this section) as well as lower bounds (as presented in Section 4), the ability to effectively incorporate warm start information is crucial.

Category (B) and the algorithms proposed herein have the same computational complexity and hence these two estimation approaches are equally scalable.

4. Certificates of Optimality via Lower Bounds

In this section, we outline our approach to computing lower bounds to (CFA_q) via techniques from global optimization and matrix analysis. In particular, we focus on the case when $q = 1$.[11] We begin with an overview of the method. We then discuss initialization parameters for the method as well as branching rules and other refinements employed in our approach.

4.1 Overview of Method

Our primary problem of interest is to provide lower bounds to (CFA_1), i.e.,

$$\underset{\mathbf{\Phi} \in \mathcal{F}_{\mathbf{\Sigma}}}{\text{minimize}} \sum_{i=r+1}^{p} \sigma_i(\mathbf{\Sigma} - \mathbf{\Phi}), \tag{38}$$

or equivalently vis-à-vis Theorem 1,

$$\underset{\substack{\mathbf{W} \in \mathbf{\Psi}_{p,p-r} \\ \mathbf{\Phi} \in \mathcal{F}_{\mathbf{\Sigma}}}}{\text{minimize}} \langle \mathbf{W}, \mathbf{\Sigma} - \mathbf{\Phi} \rangle. \tag{39}$$

One possible approach is to consider *convex* lower bounds to (39).

Definition 3 *For a function $f : \Gamma \subseteq \mathbb{R}^n \to \mathbb{R}$ we define its convex envelope on Γ, denoted* $\text{convenv}_\Gamma(f)$*, to be the largest convex function g with $g \leq f$. In symbols,*

$$g = \sup\{h : \Gamma \to \mathbb{R} \mid h \text{ convex on } \Gamma \text{ and } h \leq f\}.$$

The convex envelope acts as the best possible convex lower bound for a given function. Further, its precise form is well-known for certain classes of functions; one such function of principle interest here is described in the following theorem. This is in some contexts referred to as a McCormick hull and is widely used throughout the nonconvex optimization literature (Floudas, 1999; Tawarmalani and Sahinidis, 2002a).

Theorem 4 (Al-Khayyal and Falk, 1983) *If $f : \Gamma = [0,1] \times [\ell, u] \to \mathbb{R}$ is defined by $f(x,y) = -xy$, then the convex envelope of f on Γ is precisely*

$$\text{convenv}_\Gamma(f)(x,y) = \max\{-ux, \ell - \ell x - y\}.$$

Further, $|f(x,y) - \text{convenv}_\Gamma(f)(x,y)| \leq (u - \ell)/4$. In particular, if $|u - \ell| \to 0$, then $\text{convenv}_\Gamma(f) \to f$.

11. The general case for $q > 1$ can be addressed using similar (although more complicated) techniques as applied to Problem (16), again applying principles developed in global optimization.

Using this result we proceed to describe our approach for computing lower bounds to (CFA_1) as in (39). First observe that if $\Phi_i \in [\ell_i, u_i]$ and $W_{ii} \in [0,1]$, then

$$\begin{aligned}\text{convenv}(-W_{ii}\Phi_i)(W_{ii}, \Phi_i) &= \max\{-u_i W_{ii}, \ell_i - \ell_i W_{ii} - \Phi_i\} \\ &= -\min\{u_i W_{ii}, \Phi_i + \ell_i W_{ii} - \ell_i\}.\end{aligned}$$

Hence, the best possible convex lower bound to the objective in Problem (39), namely $\langle \mathbf{W}, \mathbf{\Sigma} - \mathbf{\Phi}\rangle = \langle \mathbf{W}, \mathbf{\Sigma}\rangle - \sum_i W_{ii}\Phi_i$, for $\boldsymbol{\ell} \leq \text{diag}(\mathbf{\Phi}) \leq \mathbf{u}$ and $\mathbf{I} \succeq \mathbf{W} \succeq \mathbf{0}$ is

$$\langle \mathbf{W}, \mathbf{\Sigma}\rangle - \sum_{i=1}^{p} \min\{u_i W_{ii}, \Phi_i + \ell_i W_{ii} - \ell_i\}.$$

By introducing auxiliary variables $\mathbf{e} \in \mathbb{R}^p$ to represent these convex envelopes, we obtain the following linear SDO that is a lower bound to (39):

$$(\text{LS}_{\boldsymbol{\ell},\mathbf{u}}) \quad \begin{array}{lll} \underset{\mathbf{W},\mathbf{\Phi},\mathbf{e}}{\text{minimize}} & \langle \mathbf{W}, \mathbf{\Sigma}\rangle - \sum_{i=1}^{p} e_i & \\ \text{s. t.} & \mathbf{\Phi} \in \mathcal{F}_{\mathbf{\Sigma}} & \\ & \mathbf{W} \in \mathbf{\Psi}_{p,p-r} & \\ & \left.\begin{array}{r} \ell_i \leq \Phi_i \leq u_i \\ e_i \leq \Phi_i + \ell_i W_{ii} - \ell_i \\ e_i \leq u_i W_{ii} \end{array}\right\} & \forall\, i \end{array}$$

The approach now lies in iteratively refining the lower and upper bounds on the diagonal entries of $\mathbf{\Phi}$, denoted $\boldsymbol{\ell}$ and $\mathbf{u}$, respectively, in order to improve the quality of the approximations obtained via convex envelopes (*cf.* Theorem 4). This classical scheme is known as *spatial branch and bound* and is shown in pseudocode in Algorithm 2 as it applies to solving (39) by way of using ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$).

In words, Algorithm 2 involves treating a given "node" $\mathtt{n} = [\boldsymbol{\ell}, \mathbf{u}]$, which represents bounds on $\mathbf{\Phi}$, namely, $\boldsymbol{\ell} \leq \text{diag}(\mathbf{\Phi}) \leq \mathbf{u}$. Here we solve ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$) with the lower and upper bounds $\boldsymbol{\ell}$ and $\mathbf{u}$, respectively, and see whether the resulting new feasible solution is better (lower in objective value) than the best known incumbent solution encountered thus far. We then see if the the bound for this node as obtained via ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$) is better than the currently known best feasible solution; if it is not at least the current best feasible solution's objective value (up to some numerical tolerance), then we must further branch on this node, generating two new nodes $\mathtt{n}_1$ and $\mathtt{n}_2$ which partition the existing node $\mathtt{n}$. Throughout, we keep track of the worst lower bound encountered, which allows for us to terminate the algorithm early while still having a provable suboptimality guarantee on the best feasible solution $\mathbf{\Phi}_f \in \mathcal{F}_{\mathbf{\Sigma}}$ found thus far.

In light of Theorem 4, we have as a corollary the following theorem.

Theorem 5 *Given numerical tolerance* $\mathtt{TOL} > 0$*, Algorithm 2 (with an appropriate branching rule)*[12] *terminates in finitely many iterations and solves* (CFA_1) *to within an additive optimality gap of at most* $\mathtt{TOL}$*. Further, if Algorithm 2 is terminated early (i.e., before* $\mathtt{Nodes} = \emptyset$*), then the best feasible solution* $\mathbf{\Phi}_f$ *at termination is guaranteed to be within an additive optimality gap of* $z_f - z_{\text{lb}}$*.*

12. A branching rule that is sufficient for convergence is selecting $i \in \arg\max_i (u_i^c - \ell_i^c)$ and $\alpha = (u_i^c + \ell_i^c)/2$.

The algorithm we have considered here omits some important details. After discussing properties of ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$) in Section 4.2, we will discuss various aspects of Algorithm 2. In Section 4.3, we detail how to choose input $\mathbf{u}^0$. We then turn our attention to branching (line 5) in Section 4.4. In Section 4.5, we use results from matrix analysis coupled with ideas from the modern practice of discrete optimization to make tailored refinements to Algorithm 2. Finally, in Section 4.6 we include a discussion of node selection strategies.

Algorithm 2 Spatial branch and bound scheme to solve (39). The inputs are as follows: (a) upper bounds $\mathbf{u}^0$ such that for any i and any $\mathbf{\Phi} \in \mathcal{F}_{\mathbf{\Sigma}}$, we have $\Phi_i \leq u_i^0$ (see Section 4.3); (b) optimality tolerance $\mathtt{TOL}$; and (c) initial feasible solution $\mathbf{\Phi}_f \in \mathcal{F}_{\mathbf{\Sigma}}$.

1. Initialize $z_f \leftarrow \sum_{i>r} \sigma_i(\mathbf{\Sigma} - \mathbf{\Phi}_f)$; $\mathtt{Nodes} \leftarrow \{([\mathbf{0}, \mathbf{u}^0], -\infty)\}$; and $z_{\text{lb}} \leftarrow -\infty$.
2. While $\mathtt{Nodes} \neq \emptyset$, remove some node $([\boldsymbol{\ell}^c, \mathbf{u}^c], z^c) \in \mathtt{Nodes}$.
3. Solve ($\text{LS}_{\boldsymbol{\ell}^c,\mathbf{u}^c}$). Let $\mathbf{\Phi}$ be an optimal solution with z the optimal objective value; set $z_u \leftarrow \sum_{i>r} \sigma_i(\mathbf{\Sigma} - \mathbf{\Phi})$.
4. If $z_u < z_f$ (i.e., a better feasible solution is found), update the best feasible solution found thus far ($\mathbf{\Phi}_f$) to be $\mathbf{\Phi}$ and update the corresponding value (z_f) to z_u.
5. If $z < z_f - \mathtt{TOL}$ (i.e., a $\mathtt{TOL}$–optimal solution has not yet been found), then pick some $i \in \{1, \ldots, p\}$ and some $\alpha \in (\ell_i^c, u_i^c)$. Then add two new nodes to $\mathtt{Nodes}$:
$$\left(\prod_{j<i}[\ell_j^c, u_j^c] \times [\ell_i^c, \alpha] \times \prod_{j>i}[\ell_j^c, u_j^c], z\right) \quad \text{and} \quad \left(\prod_{j<i}[\ell_j^c, u_j^c] \times [\alpha, u_i^c] \times \prod_{j>i}[\ell_j^c, u_j^c], z\right).$$
6. Update the best lower bound $z_{\text{lb}} \leftarrow \min_{([\boldsymbol{\ell},\mathbf{u}],z)\in\mathtt{Nodes}} z$ and return to Step 2.

Global Optimization—State of the Art: We close this section by discussing similarities between the branch-and-bound approach we develop here and existing methods in nonconvex optimization. Our approach is very similar in spirit to approaches to global optimization (Floudas, 1999), and in particular for (nonconvex) quadratic optimization problems, quadratically-constrained convex optimization problems, and bilinear optimization problems (Hansen et al., 1993; Bao et al., 2009; Tawarmalani and Sahinidis, 2002b; Tawarmalani et al., 2013; Costa and Liberti, 2012; Anstreicher and Burer, 2010; Misener and Floudas, 2012). The primary similarity is that we work within a branch and bound framework using successively better convex lower bounds. However, while global optimization software for a variety of nonconvex problems with underlying vector variables is generally well-developed (as evidenced by solvers like `BARON`, see Sahinidis 2014), this is not the case for problems with underlying matrix variables and semidefinite constraints.

The presence of semidefinite structure presents several substantial computational challenges. First and foremost, algorithmic implementations for solving linear SDOs are not nearly as advanced as those which exist for linear optimization problems. Therefore, each

subproblem, which is itself a linear SDO, carries a larger computational cost than the usual corresponding linear program which typically arises in other global optimization problems with vector variables. Secondly, a critical component of the success of global optimization software is the ability to quickly resolve multiple instances of subproblems which have similar structure. Corresponding methods for SDOs, as solved via interior point methods, are generally not well-developed. Finally, semidefinite structure complicates the traditional process of computing convex envelopes. Such computations are critical to the success of modern global optimization solvers like `BARON`.

There are a variety of other possible approaches to computing lower bounds to (CFA_q). One such approach is the method of moments (Lasserre, 2009). However, for problems of the size we are considering, such an approach is likely not computationally feasible, so we do not make a direct comparison here. There is also recent work in complementarity constraints literature (Bai et al., 2016) which connects rank-constrained optimization problems to copositive optimization (Burer, 2012). In short, such an approach turns (38) into an equivalent convex problem; despite the transformation, the new problem is not particularly amenable to computation at this time. For this reason, we do not consider the copositive optimization approach.

4.2 Properties of ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$)

We now examine properties of ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$), the main subproblem of interest. Observe that it is a linear SDO problem, and therefore we can consider its dual, namely

$$
\begin{array}{lll}
 & \underset{q,\boldsymbol{f}_u,\boldsymbol{f}_\ell,\boldsymbol{\mu},\boldsymbol{\sigma},\mathbf{M},\mathbf{N},\mathbf{P}}{\text{maximize}} & (p-r)q - \mathbf{u}'\boldsymbol{f}_u - \text{Tr}(\mathbf{N}) - \langle \mathbf{P}, \boldsymbol{\Sigma} - \text{diag}(\boldsymbol{\ell})\rangle \\
 & \text{s.t.} & \boldsymbol{\mu} + \boldsymbol{\sigma} = \mathbf{1} \\
(\text{DS}_{\boldsymbol{\ell},\mathbf{u}}) & & \text{diag}(\mathbf{P}) + \boldsymbol{f}_u - \boldsymbol{f}_\ell - \boldsymbol{\mu} = \mathbf{0} \\
 & & \boldsymbol{\Sigma} - \text{diag}(\mathbf{u}) + \mathbf{M} + \mathbf{N} + \text{diag}(\text{diag}(\mathbf{u}-\boldsymbol{\ell})\boldsymbol{\mu}) - q\mathbf{I} = \mathbf{0} \\
 & & \boldsymbol{f}_u, \boldsymbol{f}_\ell, \boldsymbol{\mu}, \boldsymbol{\sigma} \geq \mathbf{0} \\
 & & \mathbf{M}, \mathbf{N}, \mathbf{P} \succcurlyeq \mathbf{0}.
\end{array}
$$

Observation 1 *We now include some remarks about structural properties of* ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$) *and its dual* ($\text{DS}_{\boldsymbol{\ell},\mathbf{u}}$).

1. *If* $\text{rank}(\boldsymbol{\Sigma}) = p$ *then the Slater condition (Boyd and Vandenberghe, 2004) holds and hence there is strong duality, so we can work with* ($\text{DS}_{\boldsymbol{\ell},\mathbf{u}}$) *instead of* ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$) *as an exact reformulation.*

2. *There exists an optimal solution to the dual with* $\boldsymbol{f}_u = \mathbf{0}$. *This is a variable reduction which is not immediately obvious. Note that* $\boldsymbol{f}_u$ *appears as the multiplier for the constraints in the primal of the form* $\text{diag}(\boldsymbol{\Phi}) \leq \mathbf{u}$. *To claim that we can set* $\boldsymbol{f}_u = \mathbf{0}$ *it suffices to show that the corresponding constraints in the primal can be ignored. Namely, if* $(\mathbf{W}^*, \boldsymbol{\Phi}^*)$ *solves* ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$) *with the constraints* $\Phi_i \leq u_i \; \forall i$ *omitted, then the pair* $(\mathbf{W}^*, \widetilde{\boldsymbol{\Phi}})$ *is feasible and optimal to* ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$) *with all the constraints included, where* $\widetilde{\boldsymbol{\Phi}}$ *is defined by*

$$\widetilde{\Phi}_i = \min\{\Phi_i^*, u_i\}.$$

Hereinafter we set $\boldsymbol{f}_u = \mathbf{0}$ *and omit the constraint* $\text{diag}(\boldsymbol{\Phi}) \leq \mathbf{u}$ *(with the caveat that, upon solving a problem and identifying some* $\boldsymbol{\Phi}^*$, *we must instead work with* $\min\{\boldsymbol{\Phi}^*, \text{diag}(\mathbf{u})\}$, *taken entrywise).*

Solving Subproblems: We briefly detail how to solve ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$). In light of the discussion in Section 3.2, we choose to apply a first-order method. Observe that we cannot solve the primal form ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$) within the branch-and-bound framework unless we solve it to optimality. Therefore, we instead choose to work with its dual ($\text{DS}_{\boldsymbol{\ell},\mathbf{u}}$). We apply an off-the-shelf solver `SCS` (O'Donoghue et al., 2016) to solve ($\text{DS}_{\boldsymbol{\ell},\mathbf{u}}$) and find reasonably accurate, *feasible* solutions for this dual problem, which guarantees that we have a lower bound to ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$).[13] In this way, we maintain the provable optimality properties of Algorithm 2 without needing to solve nodes in the branch-and-bound tree to full optimality.

4.3 Input Parameters

In solving the root node of Algorithm 2, we must begin with some choice of $\mathbf{u}^0 = \mathbf{u}$. An obvious first choice for u_i is $u_i = \Sigma_{ii}$, but one can do better. Let us optimally set u_i, defining it as

$$
u_i := \begin{array}{ll} \underset{x \in \mathbb{R}}{\text{maximize}} & x \\ \text{s.t.} & \boldsymbol{\Sigma} - x\mathbf{E}^{(i)} \succcurlyeq \mathbf{0}, \end{array} \tag{40}
$$

where $\mathbf{E}^{(i)} \in \mathbb{R}^{p \times p}$ is a matrix with all zeros except $\mathbf{E}^{(i)}_{ii} = 1$. These bounds are useful because if $\boldsymbol{\Phi} \in \mathcal{F}_{\boldsymbol{\Sigma}}$, then $\Phi_i \leq u_i$. Note that problem (40) is a linear SDO for which strong duality holds. Its dual is precisely

$$
u_i = \begin{array}{ll} \underset{\mathbf{M} \in \mathbb{R}^{p \times p}}{\text{minimize}} & \langle \mathbf{M}, \boldsymbol{\Sigma} \rangle \\ \text{s.t.} & M_{ii} = 1 \\ & \mathbf{M} \succcurlyeq \mathbf{0}, \end{array} \tag{41}
$$

a linear SDO in standard form with a single equality constraint. By a result of (Barvinok, 1995; Pataki, 1998), there exists a rank one solution to (41). This implies that (41) can actually be solved as a convex quadratic program:

$$
u_i = \begin{array}{ll} \underset{\mathbf{m} \in \mathbb{R}^p}{\text{minimize}} & \mathbf{m}'\boldsymbol{\Sigma}\mathbf{m} \\ \text{s.t.} & m_i^2 = 1. \end{array} = \begin{array}{ll} \underset{\mathbf{m}}{\text{minimize}} & \mathbf{m}'\boldsymbol{\Sigma}\mathbf{m} \\ \text{s.t.} & m_i = 1. \end{array} \tag{42}
$$

This formulation given in (42) is computationally inexpensive to solve (given a large number of specialized convex quadratic problem solvers), in contrast to both formulations (40) and (41).[14]

13. One notable feature of the ADMM-based approach is that we can extract an approximately feasible primal solution $\boldsymbol{\Phi}$, which is useful for branching. Note that in Algorithm 2, we can replace the best incumbent solution if we find a new $\boldsymbol{\Phi}$ which has better objective value $\sum_{i>r} \sigma_i(\boldsymbol{\Sigma} - \boldsymbol{\Phi})$. Because $\boldsymbol{\Phi}$ may not be feasible (i.e., $\boldsymbol{\Phi} \notin \mathcal{F}_{\boldsymbol{\Sigma}}$), we take care here. Namely, compute $t = \sum_{i>r} \lambda_i(\boldsymbol{\Sigma} - \boldsymbol{\Phi})$, where $\lambda_1(\boldsymbol{\Sigma} - \boldsymbol{\Phi}) \geq \cdots \geq \lambda_p(\boldsymbol{\Sigma} - \boldsymbol{\Phi})$ are the sorted eigenvalues of $\boldsymbol{\Sigma} - \boldsymbol{\Phi}$. If $t < z_f$, then we perform an iteration of CG scheme for finding feasible solutions (outlined in Section 3) to find a feasible $\bar{\boldsymbol{\Phi}} \in \mathcal{F}_{\boldsymbol{\Sigma}}$. We then use this as a possible candidate for replacing the incumbent.

14. In the case when $\boldsymbol{\Sigma} \succ \mathbf{0}$, it is not necessary to use quadratic optimization problems to compute $\mathbf{u}$. In this case once can apply a straightforward Schur complement argument (Boyd and Vandenberghe, 2004) to show that $\mathbf{u}$ can be computed by solving for inverse of p different $(p-1) \times (p-1)$ matrices (or equivalently, by finding the diagonal of the precision matrix $\boldsymbol{\Sigma}^{-1}$).

4.4 Branching

Here we detail two methods for branching (line 5 in Algorithm 2). The problem of branching is as follows: having solved ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$) for some particular $\mathtt{n} = [\boldsymbol{\ell}, \mathbf{u}]$, we must choose some $i \in \{1, \ldots, p\}$ and split the interval $[\ell_i, u_i]$ to create two new subproblems. We begin with a simple branching rule. Given a solution $(\mathbf{W}^*, \boldsymbol{\Phi}^*, \mathbf{e}^*)$ to ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$), compute $i \in \text{argmax}_i |e_i^* - W_{ii}^* \Phi_i^*|$ and branch on variable Φ_i, generating two new subproblems with the intervals

$$\prod_{j<i}[\ell_j, u_j] \times [\ell_i, \Phi_i^*] \times \prod_{j>i}[\ell_j, u_j] \quad \text{and} \quad \prod_{j<i}[\ell_j, u_j] \times [\Phi_i^*, u_i] \times \prod_{j>i}[\ell_j, u_j].$$

Observe that, so long as $\max_i |e_i^* - W_{ii}^* \Phi_i^*| > 0$, the solution $(\mathbf{W}^*, \boldsymbol{\Phi}^*, \mathbf{e}^*)$ is not optimal for either of the subproblems created.

We now briefly describe an alternative rule which we employ instead. We again pick the branching index i as before, but now the two new nodes we generate are

$$\prod_{j<i}[\ell_j, u_j] \times [\ell_i, (1-\epsilon)\Phi_i^* + \epsilon \ell_i] \times \prod_{j>i}[\ell_j, u_j] \quad \text{and} \quad \prod_{j<i}[\ell_j, u_j] \times [(1-\epsilon)\Phi_i^* + \epsilon \ell_i, u_i] \times \prod_{j>i}[\ell_j, u_j],$$

where $\epsilon \in [0, 1)$ is some parameter. For the computational experiments, we set $\epsilon = 0.4$.

Such an approach, which lowers the location of the branch in the ith interval $[\ell_i, u_i]$ from Φ_i^*, serves to improve the objective value from the first node, while hurting the objective value from the second node (here by objective value, we mean the objective value of the optimal solution to the two new subproblems). In this way, it spreads out the distance between the two, and so it is more likely that the first node may have an objective value that is higher than $z_f - \mathtt{TOL}$ than before, and hence, this would mean there are fewer nodes necessary to consider to solve for an additive gap of $\mathtt{TOL}$. While this heuristic explanation is only partially satisfying, we have observed throughout a variety of numerical experiments that this rule, even though simple, performs better across a variety of example classes than the basic branching rule outlined. At the same time, recent work on the theory of branching rules supports such a heuristic rule (le Bodic and Nemhauser, 2015). In Section 5.4, we give evidence on the impact of the use of the modified branching rule.

4.5 Weyl's Method—Pruning and Bound Tightening

In this subsection, we develop another method for lower bounds for the factor analysis problem. While we use it to supplement our approach detailed throughout Section 4, it is of interest as a standalone method, particularly for its computational speed and simplicity. In Section 5.3, we discuss the performance of this approach in both contexts.

The central problem of interest in factor analysis involves the spectrum of a symmetric matrix ($\boldsymbol{\Sigma} - \boldsymbol{\Phi}$) which is the difference of two other symmetric matrices ($\boldsymbol{\Sigma}$ and $\boldsymbol{\Phi}$). From a linear algebraic perspective, the spectrum of the sum of real symmetric matrices is an extensively studied problem (Horn and Johnson, 2013; Bhatia, 2007).Therefore, it is natural to inquire how such results carry over to our setting. We discuss the implications of a well-known result from this literature, namely *Weyl's inequality* (see, e.g., Bhatia 2007):

Theorem 6 (Weyl's Inequality) *For symmetric matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{p \times p}$ with sorted eigenvalues*

$$\lambda_1(\mathbf{A}) \geq \lambda_2(\mathbf{A}) \geq \cdots \geq \lambda_p(\mathbf{A}) \quad \text{and} \quad \lambda_1(\mathbf{B}) \geq \lambda_2(\mathbf{B}) \geq \cdots \geq \lambda_p(\mathbf{B})$$

one has for any $k \in \{1, \ldots, p\}$ that

$$\lambda_k(\mathbf{A} + \mathbf{B}) \geq \lambda_{k+j}(\mathbf{A}) + \lambda_{p-j}(\mathbf{B}) \; \forall j = 0, \ldots, p-k.$$

For any vector $\mathbf{x} \in \mathbb{R}^p$ we let $\{x_{(i)}\}_{i=1}^p$ denote sorted $\{x_i\}_{i=1}^p$ with $x_{(1)} \geq x_{(2)} \geq \cdots \geq x_{(p)}$. Using this notation, we arrive at the following theorem.

Theorem 7 *For any diagonal matrix $\bar{\boldsymbol{\Phi}}$ one has that*

$$\underset{\boldsymbol{\Phi} \in \mathcal{F}_{\boldsymbol{\Sigma}}}{\text{minimize}} \sum_{i>r} \sigma_i(\boldsymbol{\Sigma} - \boldsymbol{\Phi}) \geq \underset{\boldsymbol{\Phi} \in \mathcal{F}_{\boldsymbol{\Sigma}}}{\text{minimize}} \left(\sum_{i>r} \max_{j=0,\ldots,p-i} \left\{ \lambda_{i+j}(\boldsymbol{\Sigma} - \bar{\boldsymbol{\Phi}}) + \text{diag}\left(\bar{\boldsymbol{\Phi}} - \boldsymbol{\Phi}\right)_{(p-j)}, 0 \right\} \right).$$

Proof Apply Weyl's inequality with $\mathbf{A} = \boldsymbol{\Sigma} - \text{diag}(\bar{\boldsymbol{\Phi}})$ and $\mathbf{B} = \bar{\boldsymbol{\Phi}} - \boldsymbol{\Phi}$, and use the fact that $\boldsymbol{\Phi} \in \mathcal{F}_{\boldsymbol{\Sigma}}$ so $\boldsymbol{\Sigma} - \boldsymbol{\Phi} \succcurlyeq \mathbf{0}$. ■

Weyl's Method: The new lower bound introduced in Theorem 7 is a nonconvex problem in general. We begin by discussing one situation in which Theorem 7 provides computationally tractable (and fast) lower bounds; we deem this *Weyl's method*, detailed as follows:

1. Compute bounds u_i as in (40), so that for all $\boldsymbol{\Phi} \in \mathcal{F}_{\boldsymbol{\Sigma}}$, one has $\Phi_i \leq u_i \; \forall i$.
2. For each $r \in \{1, \ldots, p\}$, one can compute a lower bound to (CFA_1) (for a given r) of

$$\sum_{i>r} \max\{\lambda_i(\boldsymbol{\Sigma} - \text{diag}(\mathbf{u})), 0\},$$

by taking Theorem 7 with $\bar{\boldsymbol{\Phi}} = \text{diag}(\mathbf{u})$.

As per the above remarks, computing $\mathbf{u}$ in Step 1 of Weyl's method can be carried out efficiently. Step 2 only relies on computing the eigenvalues of $\boldsymbol{\Sigma} - \text{diag}(\mathbf{u})$. Therefore, this lower bounding procedure is quite simple to carry out. What is perhaps surprising is that this simple lower bounding procedure is effective as a standalone method. We describe such results in Section 5.3. We now turn our attention to how Weyl's method can be utilized within the branch and bound tree as described in Algorithm 2.

Pruning: We begin by considering how Weyl's method can be used for *pruning*. The notion of pruning for branch and bound trees is grounded in the theory and practice of discrete optimization. In short, pruning in the elimination of nodes from the tree without actually solving them. We make this precise in our context.

Consider some point in the branch and bound process in Algorithm 2, where we have some collection of nodes, $([\boldsymbol{\ell}^c, \mathbf{u}^c], z^c) \in \mathtt{Nodes}$. Recall that z^c is the optimal objective

value of the parent node of $\mathtt{n}$. Vis-à-vis Weyl's method, we know *a priori*, without solving $(\mathrm{LS}_{\boldsymbol{\ell}^c,\mathbf{u}^c})$, that

$$\underset{\substack{\boldsymbol{\Phi}\in\mathcal{F}_{\boldsymbol{\Sigma}}\\ \boldsymbol{\ell}^c\leq\mathrm{diag}(\boldsymbol{\Phi})\leq\mathbf{u}^c}}{\text{minimize}} \sum_{i>r}\sigma_i(\boldsymbol{\Sigma}-\boldsymbol{\Phi}) \geq \sum_{i>r}\max\{\lambda_i(\boldsymbol{\Sigma}-\mathrm{diag}(\mathbf{u}^c)),0\}.$$

Hence, if $z_f - \mathtt{TOL} < \sum_{i>r}\max\{\lambda_i(\boldsymbol{\Sigma}-\mathrm{diag}(\mathbf{u}^c)),0\}$, where z_f is as in Algorithm 2, then node $\mathtt{n}$ can be discarded, i.e., there is no need to actually compute $(\mathrm{LS}_{\boldsymbol{\ell}^c,\mathbf{u}^c})$ or further consider this branch. This is because if we were to solve $(\mathrm{LS}_{\boldsymbol{\ell}^c,\mathbf{u}^c})$, and then branch again, solving further down this branch to optimality, then the final lower bound obtained would necessarily be at least as large as the best feasible objective already found (within tolerance $\mathtt{TOL}$).

In this way, because Weyl's method is relatively fast, this provides a simple method for pruning. In the computational results detailed in Section 5.3, we always use Weyl's method to discard nodes which are not fruitful to consider.

Bound Tightening: We now turn our attention to another way in which Weyl's method can be used to improve the performance of Algorithm 2—*bound tightening*. In short, bound tightening is the use of implicit constraints to strengthen bounds obtained for a given node. We detail this with the same node notation as above. Namely, consider a given node $\mathtt{n} = [\boldsymbol{\ell}^c, \mathbf{u}^c]$. Fix some $j \in \{1,\ldots,p\}$ and let $\alpha \in (\ell_j^c, u_j^c)$. If we have that

$$z_f - \mathtt{TOL} < \sum_{i>r}\max\{\lambda_i(\boldsymbol{\Sigma}-\mathrm{diag}(\tilde{\mathbf{u}})),0\},$$

where $\tilde{\mathbf{u}}$ is $\mathbf{u}^c$ with the jth entry replaced by α, then we can replace the node $\mathtt{n}$ with the "tightened" node $\tilde{\mathtt{n}} = [\tilde{\boldsymbol{\ell}}, \mathbf{u}^c]$, where $\tilde{\boldsymbol{\ell}}$ is $\boldsymbol{\ell}^c$ with the jth entry replaced by α.

We consider why this is valid. Suppose that one were to solve $(\mathrm{LS}_{\boldsymbol{\ell}^c,\mathbf{u}^c})$ and choose to branch on index j at α. Then one would create two new nodes: $[\boldsymbol{\ell}^c, \tilde{\mathbf{u}}]$ and $[\tilde{\boldsymbol{\ell}}, \mathbf{u}^c]$. We would necessarily then prune away the node $[\boldsymbol{\ell}^c, \tilde{\mathbf{u}}]$ as just described; hence, we can replace $[\boldsymbol{\ell}^c, \mathbf{u}^c]$ without loss of generality with $[\tilde{\boldsymbol{\ell}}, \mathbf{u}^c]$. Note that here $\alpha \in (\ell_j^c, u_j^c)$ and $j \in \{1,\ldots,p\}$ were arbitrary. Hence, for each j, one can choose the largest such $\alpha_j \in (\ell_j^c, u_j^c)$ (if one exists) so that $z_f - \mathtt{TOL} < \sum_{i>k}\max\{\lambda_i(\boldsymbol{\Sigma}-\mathrm{diag}(\tilde{\mathbf{u}})),0\}$, and then replace $\boldsymbol{\ell}^c$ by $\tilde{\boldsymbol{\ell}}$.[15]

Such a procedure is somewhat expensive (because of its use of repeated eigenvalue calculations), but can be thought of as "optimal" pruning via Weyl's method. In our experience the benefit of bound tightening does not warrant its computational cost when used at every node in the branch-and-bound tree except in a small number of problems. For this reason, in the computational results in Section 5.3 we only employ bound tightening at the root node $\mathtt{n} = [\mathbf{0}, \mathbf{u}^0]$.

4.6 Node Selection

In this section, we briefly describe our method of node selection. The problem of node selection has been considered extensively in discrete optimization and is still an active area of research. Here we describe a simple node selection strategy.

15. An obvious choice to find such an α is a grid-search-based bisection method. For simplicity we use a linear search on a grid instead of resorting to the bisection method.

To be precise, consider some point in Algorithm 2 were we have a certain collection of nodes, $([\boldsymbol{\ell}^c, \mathbf{u}^c], z^c) \in \mathtt{Nodes}$. The most obvious node selection strategy is to pick the node $\mathtt{n}$ for which z^c is smallest among all nodes in $\mathtt{Nodes}$. In this way, the algorithm is likely to improve the gap $z_f - z_{\mathrm{lb}}$ at every iteration. Such greedy selection strategies tend to not perform particularly well in general global optimization problems (see, e.g., Tawarmalani and Sahinidis, 2002a).

For these reasons, we employ a slightly modified greedy selection strategy which utilizes Weyl's method. For a given node $\mathtt{n}$, we also consider its corresponding lower bound w^c obtained from Weyl's method, namely, $w^c := \sum_{i>r}\max\{\lambda_i(\boldsymbol{\Sigma}-\mathrm{diag}(\mathbf{u}^c)),0\}$. For each node, we now consider $\max\{z^c, w^c\}$. There are two cases to consider:

1. With probability β, we select the node with smallest value of $\max\{z^c, w^c\}$.

2. In the remaining scenarios (occurring with probability $1-\beta$), we choose randomly between selecting the node with smallest value of z^c and the node with smallest value of w^c. To be precise, let Z be the minimum of z^c over all nodes and likewise let W be the minimum of w^c over all nodes. Then with (independent) probability β, we choose the node with worst z^c *or* w^c (i.e., with $\min\{z^c, w^c\} = \min\{Z, W\}$); with probability $1-\beta$, if $Z < W$ we choose a node with $w^c = W$, and if $Z > W$ we choose a node with $z^c = Z$.

In this way, we allow for the algorithm to switch between trying to make progress towards improving the convex envelope bounds and making progress towards improving the best of the two bounds (the convex envelope bounds along with the Weyl bounds). We set $\beta = 0.9$ for all computational experiments. It is possible that a more dynamic branching strategy could perform substantially better; however, the method here has a desirable level of simplicity.

We close by noting that while this node selection strategy appears naïve, it is not necessarily so simple. Improved node selection strategies from discrete optimization often take into account some sort of duality theory. Weyl's inequality is at its core a result from duality theory (principally Wielandt's minimax principle; see Bhatia, 2007), and therefore our strategy is not as unsophisticated as it might appear on first inspection.

5. Computational experiments

In this section, we perform various computational experiments to study the properties of our different algorithmic proposals for (CFA_q), for $q \in \{1,2\}$. Using a variety of statistical measures, we compare our methods with existing popular approaches for FA, as implemented in standard `R` statistical packages `psych` (Revelle, 2015), `nFactors` (Raiche and Magis, 2011), and `GPArotation` (Bernaards and Jennrich, 2005). We then turn our attention to certificates of optimality as described in Section 4 for (CFA_1).

5.1 Synthetic examples

For our synthetic experiments, we considered distinctly different groups of examples. Classes A_1 and A_2 have subspaces of the low-rank common factors which are random and the values

of Φ_i are taken to be equally spaced. The underlying matrix corresponding to the common factors in type A_1 is exactly low-rank, while this is not the case in type A_2.

Class $A_1(R/p)$. We generated a matrix $\mathbf{L}_{p\times R} := ((L_{ij}))$ (with $R < p$) with $L_{ij} \overset{\text{iid}}{\sim} N(0,1)$. The *unique variances* $\Phi_1, \ldots, \Phi_p$, are taken to be proportional to p equi-spaced values on the interval $[\lambda_R(\mathbf{L}'\mathbf{L}), \lambda_1(\mathbf{L}'\mathbf{L})]$ such that $\Phi_i = \bar{\phi} \cdot \left(\lambda_1(\mathbf{L}'\mathbf{L}) + (\lambda_R(\mathbf{L}'\mathbf{L}) - \lambda_1(\mathbf{L}'\mathbf{L}))\frac{i-1}{p}\right)$, for $1 \le i \le p$. Here $\bar{\phi}$, which controls the ratio of the variances between the *uniquenesses* and the common latent factors is chosen such that $\sum_{i=1}^{p} \Phi_i = \mathrm{Tr}(\mathbf{L}\mathbf{L}')$, i.e., the contribution to the total variance from the common factors matches that from the uniqueness factors. The covariance matrix is thus given by: $\mathbf{\Sigma} = \mathbf{L}\mathbf{L}' + \mathbf{\Phi}$.

Class $A_2(p)$. Here $\mathbf{L}_{p\times p}$ is generated as $L_{ij} \overset{\text{iid}}{\sim} N(0,1)$. We did a full singular value decomposition on $\mathbf{L}$—let $\mathbf{U}_L$ denote the set of p (left) singular vectors. We created a positive definite matrix with exponentially decaying eigenvalues as follows: $\widetilde{\mathbf{L}}\widetilde{\mathbf{L}}' = \mathbf{U}_L\,\mathrm{diag}(\lambda_1, \ldots, \lambda_p)\mathbf{U}_L'$, where the eigenvalues were chosen as $\lambda_i = 0.8^{i/2}, i = 1, \ldots, p$. We chose the diagonal entries of $\mathbf{\Phi}$ (like data Type-A_1), as a scalar multiple ($\bar{\phi}$) of a uniformly spaced grid in $[\lambda_p, \lambda_1]$ and $\bar{\phi}$ was chosen such that $\sum_i \Phi_i = \mathrm{Tr}(\widetilde{\mathbf{L}}\widetilde{\mathbf{L}}')$.

In contrast, classes B_1, B_2, and B_3 are qualitatively different from the aforementioned ones—the subspaces corresponding to the common factors are more structured, and hence different from the coherence-like assumptions on the eigenvectors which are necessary for nuclear norm based methods (Saunderson et al., 2012) to work well.

Class $B_1(R/p)$. We set $\mathbf{\Theta} = \mathbf{L}\mathbf{L}'$, where $\mathbf{L}_{p\times R}$ is given by

$$L_{ij} = \begin{cases} 1, & i \le j \\ 0, & i > j. \end{cases}$$

Class $B_2(r/R/p)$. Here we set $\mathbf{\Theta} = \mathbf{L}\mathbf{L}'$, where $\mathbf{L}_{p\times R}$ is such that

$$L_{ij} = \begin{cases} 1, & i,j = 1, \ldots, r \\ \overset{\text{iid}}{\sim} N(0,1), & i > r, j = 1, \ldots, R \\ 0, & i = 1, \ldots, r,\ j > r. \end{cases}$$

Class $B_3(r/R/p)$. Here we define $\mathbf{\Theta} = \mathbf{L}\mathbf{L}'$, where $\mathbf{L}_{p\times R}$ is such that

$$L_{ij} = \begin{cases} 1, & j = 1, \ldots, r,\ i \le j \\ \overset{\text{iid}}{\sim} N(0,1), & j > r,\ i = 1, \ldots, R \\ 0, & i > j,\ j = 1, \ldots, r. \end{cases}$$

In all the B classes, we generated $\Phi_i \overset{\text{iid}}{\sim} \mathrm{abs}(N(0,1))$ and the covariance matrix $\mathbf{\Sigma}$ was taken to be $\mathbf{\Sigma} = \mathbf{\Theta} + \alpha\mathbf{\Phi}$, where α is so that $\mathrm{Tr}(\mathbf{\Theta}) = \alpha\,\mathrm{Tr}(\mathbf{\Phi})$.

Comparisons with other FA methods

We performed a suite of experiments using Algorithm 1 for the cases $q \in \{1, 2\}$. We compared our proposed algorithm with the following popular FA estimation procedures as described in Section 1.1:

1. MINRES: minimum residual factor analysis
2. WLS: weighted least squares method with weights being the *uniquenesses*
3. PA: this is the principal axis factor analysis method
4. MTFA: constrained minimum trace factor analysis—formulation (6)
5. PC: The method of principal component factor analysis
6. MLE: this is the maximum likelihood estimator (MLE)
7. GLS: the generalized least squares method

For MINRES, WLS, GLS, and PA, we used the implementations available in the `R` package `psych` (Revelle, 2015) available from `CRAN`. For MLE we investigated the methods `factanal` from `R` package `stats` and the `fa` function from `R` package `psych`. The estimates obtained by the different MLE implementations were similar in our experiments; therefore, we report the results obtained from `factanal`.

For MTFA, we used our own implementation by adapting the ADMM algorithm (Section 3.2.2) to solve Problem (6). For the experiments in Section 5.1, we took the convergence thresholds for Algorithm 1 as $\mathrm{TOL} = 10^{-5}$ and ADMM as $\mathrm{TOL} \times \alpha = 10^{-9}$. For the PC method we followed the description in Bai and Ng (2008) (as described in Section 1.1)—the $\mathbf{\Phi}$ estimates were thresholded at zero if they became negative.

Note that all the methods considered in the experiments, apart from MTFA, allow the user to specify the desired number of factors in the problem. Since standard implementations of MINRES, WLS and PA require $\mathbf{\Sigma}$ to be a correlation matrix, we standardized all covariance matrices to correlation matrices at the outset.

5.1.1 Performance measures

We consider the following measures of "goodness of fit" (see Bartholomew et al. 2011 and references therein) to assess the performances of the different FA estimation procedures.

Estimation error in $\mathbf{\Phi}$: We use the following measure to assess the quality of an estimator for $\mathbf{\Phi}$:

$$\mathrm{Error}(\mathbf{\Phi}) := \sum_{i=1}^{p} (\widehat{\Phi}_i - \Phi_i)^2. \tag{43}$$

The estimation of $\mathbf{\Phi}$ plays an important role in FA—given a good estimate $\widehat{\mathbf{\Phi}}$, the r-common factors can be obtained by a rank-r eigendecomposition on the residual covariance matrix $\mathbf{\Sigma} - \widehat{\mathbf{\Phi}}$.

Proportion of variance explained and semi-definiteness of $(\mathbf{\Sigma} - \mathbf{\Phi})$: A fundamental objective in FA lies in understanding how well the r-common factors explain the residual covariance, i.e., $(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$—a direct analogue of what is done in PCA, as explained in Section 1. For a given r, the proportion of variance explained by the r common factors is given by

$$\text{Explained Variance} = \sum_{i=1}^{r} \lambda_i(\widehat{\mathbf{\Theta}}) / \sum_{i=1}^{p} \lambda_i(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}}). \tag{44}$$

As r increases, the explained variance increases to one. This trade-off between r and "Explained Variance" plays an important role in exploratory FA and in particular the choice

of r. For the expression (44) to be meaningful, it is desirable to have $\mathbf{\Sigma} - \widehat{\mathbf{\Phi}} \succeq \mathbf{0}$. Note that our framework (CFA$_q$), and in particular MTFA, estimates $\mathbf{\Phi}$ under a PSD constraint on $\mathbf{\Sigma} - \mathbf{\Phi}$. However, as seen in our experiments $(\widehat{\mathbf{\Phi}}, \widehat{\mathbf{\Theta}})$ estimated by the remaining methods MINRES, PA, WLS, GLS, MLE and others often violate the PSD condition on $\mathbf{\Sigma} - \widehat{\mathbf{\Phi}}$ for some choices of r, thereby rendering the interpretation of "Explained Variance" troublesome.

For the MTFA method with estimator $\widehat{\mathbf{\Theta}}$, the measure (44) applies only for the value of $r = \text{rank}(\widehat{\mathbf{\Theta}})$ and the explained variance is one.

For the methods we have included in our comparisons, we report the values of "Explained Variance" as delivered by the R-package implementations.

Proximity between $\widehat{\mathbf{\Theta}}$ and $\mathbf{\Theta}$: A relevant measure of the proximity between $\mathbf{\Theta}$ and its estimate $(\widehat{\mathbf{\Theta}})$ is given by

$$\text{Error}(\mathbf{\Theta}) := \|\widehat{\mathbf{\Theta}} - \mathbf{\Theta}_r\|_2^2 / \|\mathbf{\Theta}_r\|_2^2, \tag{45}$$

where $\mathbf{\Theta}_r$ is the best rank-r approximation to $\mathbf{\Theta}$ and can be viewed as the natural "oracle" counterpart of $\widehat{\mathbf{\Theta}}$. Note that MTFA does not incorporate any constraint on rank$(\widehat{\mathbf{\Theta}})$ in its formulation. Since the estimates obtained by this procedure satisfy $\widehat{\mathbf{\Theta}} = \mathbf{\Sigma} - \widehat{\mathbf{\Phi}}$, rank$(\widehat{\mathbf{\Theta}})$ may be quite different from r.

Discussion of experimental results. We next discuss our findings from the numerical experiments for the synthetic datasets.

Table 1 shows the performances of the various methods for different choices of p and R for class A_1. For the problems (CFA$_q$), $q \in \{1, 2\}$, we present the results of Algorithm 1. (Results obtained by the approach in Appendix B were similar.) In all the examples, with the exception of MTFA, we set the number of factors to be $r = (R-1)$, one less than the rank of the covariance matrix for the common underlying factors; in other words, the "remaining" rank one component can be considered as noise. For MTFA, the rank of $\widehat{\mathbf{\Theta}}$ was computed as the number of eigenvalues of $\widehat{\mathbf{\Theta}}$ larger than 10^{-5}. MTFA and (CFA$_q$), $q \in \{1, 2\}$, estimate $\mathbf{\Phi}$ with zero error—significantly better than competing methods. While MTFA and (CFA$_q$), $q \in \{1, 2\}$, result in estimates such that $\mathbf{\Sigma} - \widehat{\mathbf{\Phi}}$ is PSD, other methods, however, fail to do so; indeed, the discrepancy can often be quite large. MTFA performs poorly in terms of estimating $\mathbf{\Theta}$ since the estimated $\mathbf{\Theta}$ has rank different than r. In terms of the proportion of variance explained (CFA$_q$) performs significantly better than all other methods. The notion of "Explained Variance" by MTFA for $r = (R-1)$ is not applicable since the rank of the estimated $\mathbf{\Theta}$ is larger than r.

Figure 1 displays results for type A_2. Here we present the results for (CFA$_q$), $q \in \{1, 2\}$, using Algorithm 1. For all the methods (with the exception of MTFA) we computed estimates of $\mathbf{\Theta}$ and $\mathbf{\Phi}$ for a range of values of r. MTFA and (CFA$_1$) do a perfect job in estimating $\mathbf{\Phi}$ and both deliver PSD matrices $\mathbf{\Sigma} - \widehat{\mathbf{\Phi}}$. MTFA computes solutions $(\widehat{\mathbf{\Theta}})$ with a higher numerical rank and with large errors in estimating $\mathbf{\Theta}$ (for smaller values of r). Among the four performance measures corresponding to MTFA, Error$(\mathbf{\Theta})$ is the only one that varies with different r values. Each of the other three measures deliver a single value corresponding to $r = \text{rank}(\widehat{\mathbf{\Theta}})$. Overall, it appears that (CFA$_q$) is significantly better than all other methods.

Figure 2 shows the results for classes B_1, B_2, and B_3. We present the results for (CFA$_q$) for $q \in \{1, 2\}$ using Algorithm 1 as before. Figure 2 shows the performance of the different

Performance measure	Method Used (CFA$_1$)	(CFA$_2$)	MTFA	MINRES	WLS	PA	PC	MLE	Problem Size (R/p)
Error($\mathbf{\Phi}$)	0.0	0.0	0.0	699.5	700.1	700.0	639.9	699.6	3/200
Explained Var.	0.6889	0.6889	-	0.2898	0.2898	0.2898	0.2956	0.2898	
$\lambda_{\min}(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$	0.0	0.0	0.0	-0.6086	-0.6194	-0.6195	-0.6140	-0.6034	
Error($\mathbf{\Theta}$)	0.0	0.0	689.68	0.1023	0.1079	0.1146	0.7071	0.1111	
Error($\mathbf{\Phi}$)	0.0	0.0	0.0	197.18	197.01	196.98	139.60	197.18	5/200
Explained Var.	0.8473	0.8473	-	0.3813	0.3813	0.3814	0.3925	0.3813	
$\lambda_{\min}(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$	0.0	0.0	0.0	-0.4920	-0.5071	-0.5081	-0.4983	-0.4919	
Error($\mathbf{\Theta}$)	0.0	0.0	190.26	0.0726	0.0802	0.0817	1.1638	0.0725	
Error($\mathbf{\Phi}$)	0.0	0.0	0.0	39.85	39.18	39.28	2.47	40.04	10/200
Explained Var.	0.9371	0.9371	-	0.4449	0.4452	0.4452	0.4686	0.4449	
$\lambda_{\min}(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$	0.0	0.0	0.0	-0.2040	-0.2289	-0.2329	-0.2060	-0.2037	
Error($\mathbf{\Theta}$)	0.0	0.0	35.94	0.0531	0.0399	0.0419	2.47	0.0560	
Error($\mathbf{\Phi}$)	0.0	0.0	0.0	3838	3859	3858	3715	3870	2/500
Explained Var.	0.7103	0.7103	-	0.3036	0.3033	0.3033	0.3056	0.3031	
$\lambda_{\min}(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$	0.0	0.0	0.0	-0.7014	-0.7126	-0.7125	-0.7121	-0.7168	
Error($\mathbf{\Theta}$)	0.0	0.0	3835.6	0.0726	0.0770	0.0766	0.5897	0.1134	
Error($\mathbf{\Phi}$)	0.0	0.0	0.0	1482.9	1479.1	1478.9	1312.9	1485.1	5/500
Explained Var.	0.8311	0.8311	-	0.3753	0.3754	0.3754	0.3798	0.3752	
$\lambda_{\min}(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$	0.0	0.0	0.0	-0.5332	-0.5433	-0.5445	-0.5423	-0.5379	
Error($\mathbf{\Theta}$)	0.0	0.0	1459.5	0.1176	0.0752	0.0746	1.1358	0.1301	
Error($\mathbf{\Phi}$)	0.0	0.0	0.0	301.94	301.08	301.04	160.29	302.1	10/500
Explained Var.	0.9287	0.9287	-	0.4443	0.4444	0.4444	0.4538	0.4443	
$\lambda_{\min}(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$	0.0	0.0	0.0	-0.3290	-0.3280	-0.3281	-0.3210	-0.3302	
Error($\mathbf{\Theta}$)	0.0	0.0	291.44	0.0508	0.0415	0.0412	2.3945	0.0516	
Error($\mathbf{\Phi}$)	0.0	0.0	0.0	19123	19088	19090	18770	19120	2/1000
Explained Var.	0.6767	0.6767	-	0.2885	0.2886	0.2886	0.2898	0.2885	
$\lambda_{\min}(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$	0.0	0.0	0.0	-0.6925	-0.7335	-0.7418	-0.7422	-0.6780	
Error($\mathbf{\Theta}$)	0.0	0.0	19037	50.76	0.3571	0.0984	0.8092	18.5930	
Error($\mathbf{\Phi}$)	0.0	0.0	0.0	6872	6862	6861	6497	6876	5/1000
Explained Var.	0.8183	0.8183	-	0.3716	0.3717	0.3717	0.3739	0.3716	
$\lambda_{\min}(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$	0.0	0.0	0.0	-0.6202	-0.6785	-0.6797	-0.6788	-0.6209	
Error($\mathbf{\Theta}$)	0.0	0.0	6818.1	0.1706	0.0776	0.0780	1.1377	0.1696	
Error($\mathbf{\Phi}$)	0.0	0.0	0.0	1682	1681	1681	1311	1682	10/1000
Explained Var.	0.9147	0.9147	-	0.4360	0.4360	0.4360	0.4408	0.4360	
$\lambda_{\min}(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$	0.0	0.0	0.0	-0.2643	-0.2675	-0.2676	-0.2631	-0.2643	
Error($\mathbf{\Theta}$)	0.0	0.0	1654.3	0.0665	0.0568	0.0570	2.4202	0.0665	

Table 1: Comparative performances of the various FA methods for data of type A_1, for different choices of R and p. In all the above methods (apart from MTFA) r was taken to be $(R-1)$. In all the cases rank$(\widehat{\mathbf{\Theta}})$ obtained by MTFA is seen to be R. The "-" symbol implies that the notion of explained variance is not meaningful for MTFA for $r = R-1$. No method in Category (B) satisfies $\mathbf{\Sigma} - \mathbf{\Phi} \succeq \mathbf{0}$. Methods proposed in this paper seem to significantly outperform their competitors, across the different performance measures.

methods in terms of four different metrics: error in $\mathbf{\Phi}$ estimation, proportion of variance explained, violation of the PSD constraint on $\mathbf{\Sigma} - \mathbf{\Phi}$, and error in $\mathbf{\Theta}$ estimation. For the case of B_1, we see that the proportion of explained variance for (CFA$_q$) reaches one at a rank smaller than that of MTFA—this shows that the nonconvex criterion (CFA$_q$) provides smaller estimates of the rank than its convex relaxation MTFA when one seeks a model that explains the full proportion of residual variance. This result is qualitatively different from the behavior seen for A_1 and A_2 where the benefit of (CFA$_q$) over MTFA was mainly due to its flexibility to control the rank of $\mathbf{\Theta}$. Algorithms in Category (A) do an excellent job in estimating $\mathbf{\Phi}$. All other competing methods perform poorly in estimating $\mathbf{\Phi}$ for small/moderate values of r. We observe that none of the methods apart from (CFA$_q$) and MTFA lead to PSD estimates of $\mathbf{\Sigma} - \widehat{\mathbf{\Phi}}$ (unless r becomes sufficiently large which corresponds to a model with a saturated fit). In terms of the proportion of variance explained, our proposal performs much better than the competing methods. We see that the error in $\mathbf{\Theta}$ estimation incurred by (CFA$_q$), increases marginally as soon as the rank r becomes larger than a certain value for B_1. Note that around the same values of r, the proportion of explained variance reaches one in both these cases, thereby suggesting that this is possibly not a region of statistical interest. In summary, Figure 2 suggests that (CFA$_q$) performs very well compared to all its competitors.

Summary: All methods of Category (B) (see Section 1.2) used in the experimental comparisons perform worse than Category (A) in terms of measures Error($\mathbf{\Phi}$), Error($\mathbf{\Theta}$) and Explained Variance for small/moderate values of r. They also lead to indefinite estimates of $\mathbf{\Sigma} - \widehat{\mathbf{\Phi}}$. MTFA performs well in estimating $\mathbf{\Phi}$ but fails in estimating $\mathbf{\Theta}$ mainly due to the lack in flexibility of imposing a rank constraint; in some cases the trace heuristic falls short of doing a good job in approximating the rank function when compared to its nonconvex counterpart (CFA$_q$). The estimation methods proposed herein have a significant edge over existing methods in producing high quality solutions across various performance metrics.

5.2 Real data examples

This section describes the performance of different FA methods on some real-world benchmark datasets popularly used in the context of FA. These datasets can be found in the `R` libraries `datasets` (R Core Team, 2014), `psych` (Revelle, 2015), and `FactoMineR` (Husson et al., 2015) and are as follows:

- The bfi data set has 2800 observations on 28 variables (25 personality self-reported items and 3 demographic variables).
- The neo data set has 1000 measurements for $p = 30$ dimensions.
- The Harman data set is a correlation matrix of 24 psychological tests given to 145 seventh- and eighth-grade children.
- The geomorphology data set is a collection of geomorphological data collected across $p = 10$ variables and 75 observations. (The geomorphology data set originally has $p = 11$, but we remove the one categorical feature, leaving $p = 10$.)

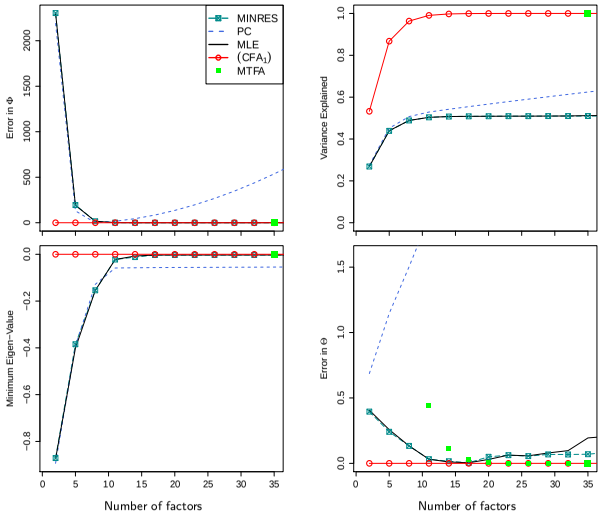


Figure 1: Performance of various FA methods for examples from class $A_2(p = 200)$ as a function of the number of factors. The y-axis label "Minimum Eigenvalue" refers to $\lambda_{\min}(\mathbf{\Sigma} - \widehat{\mathbf{\Phi}})$. We present the results of (CFA$_1$), as obtained via Algorithm 1—the results of (CFA$_2$) were similar, and hence omitted from the plot. Our methods seems to perform better than all the other competitors. For large values of r, as the fits saturate, the methods become similar. The methods (as available from the `R` package implementations) that experienced convergence difficulties do not appear in the plot.

- The JO data set records athletic performance in Olympic events, and involves 24 observations with $p = 58$. This example is distinctive because the corresponding correlation matrix $\mathbf{\Sigma}$ is not full rank (having more variables than observations).

We present the results in Figure 3. We also experimented with other methods—such as WLS and GLS—but the results were similar to MINRES and hence have not been shown in the figure. For the real examples, most of the performance measures described in Section 5.1.1 do not apply[16]; however, the notion of explained variance (44) does apply. We used this metric to compare the performance of different competing estimation procedures. We observe that solutions delivered by Category (B) explain the maximum amount of residual variance for a given rank r, which is indeed desirable, especially in the light of its analogy with PCA on the residual covariance matrix $\mathbf{\Sigma} - \mathbf{\Phi}$.

16. Understanding performance of FA methods on real data is difficult because it is an unsupervised problem. However, we can understand the performances of different methods by drawing parallels with PCA in terms of the proportion of variance explained of the matrix $\mathbf{\Sigma} - \mathbf{\Phi}$—see our discussion in Section 1.

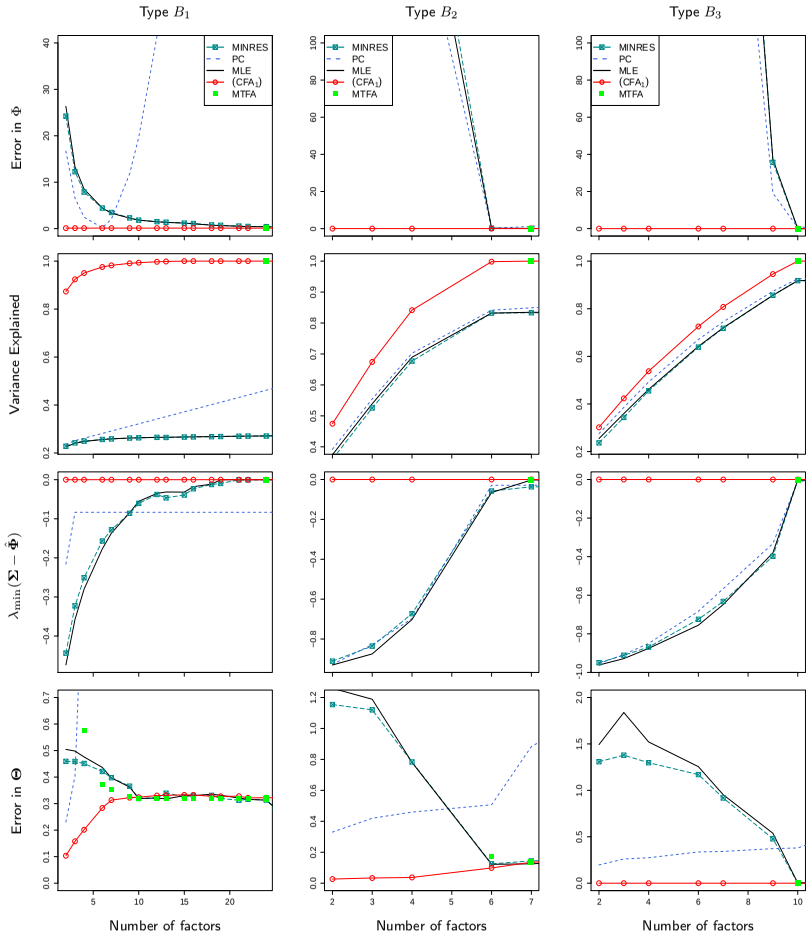


Figure 2: Performance of different methods for instances of $B_1(30/100)$, $B_2(5/10/100)$, and $B_3(5/10/100)$. We see that (CFA$_1$) exhibits very good performance across all instances, significantly outperforming the competing methods (the results of (CFA$_2$) were similar).

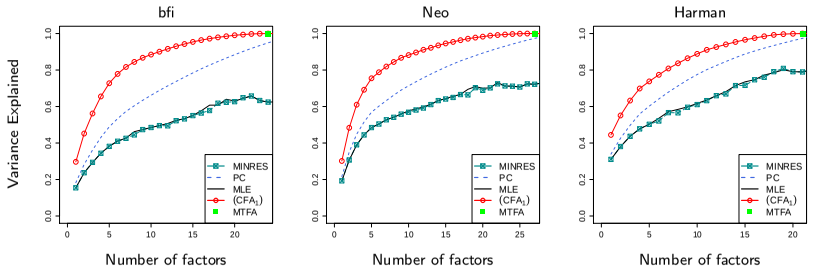


Figure 3: Figure showing proportion of variance explained by different methods for real-data examples. We see that in terms of the proportion of explained variance, (CFA$_1$) delivers the largest values for different values of r, which is indeed desirable. (CFA$_1$) also shows nice flexibility in delivering different models with varying r, in contrast to MTFA which delivers one model with proportion of variance explained equal to one. The results of (CFA$_2$) were found to be similar to (CFA$_1$).

5.3 Certificates of Optimality via Algorithm 2

We now turn our attention to certificates of optimality using Algorithm 2. Computational results of Algorithm 2 for a variety of problem sizes across all six classes can be found in Tables 2, 3, 4, 5, 6, 7, 8, 9, 10, and 11. In general, we provide results for p ranging between 10 and 4000. Parametric choices are outlined in depth in Table 2.[17] We always initialize Algorithm 2 with an initial feasible solution as found via Algorithm 1 so that we can understand if the estimators found via the CG-based approach are indeed optimal. In particular, if the best feasible objective value does not change throughout the branch-and-bound algorithm, then the initial estimator was indeed optimal (within the numerical tolerance).

Root Node Gap: Let us first consider the gap at the root node. In classes A_1, B_1, B_2, and B_3, we see that the Weyl bound at the root node often provides a better bound than the one given by using convex envelopes. Indeed, the bound provided by Weyl's method can in many instances certify optimality (up to numerical tolerance) at the *root node*. For example, this is the case in many instances of classes A_1 and half of the instances in B_3. Given that Weyl's method is computationally inexpensive (only requiring the computation of p convex quadratic optimization problems and an eigenvalue decomposition), this suggests that Weyl's inequality as used within the context of factor analysis is particularly fruitful.

In contrast, in class A_2 and the real examples, we see that the convex envelope bound tends to perform better. Because of the structure of Weyl's inequality, Weyl's method is well-suited for correlation matrices $\mathbf{\Sigma}$ with very quickly decaying eigenvalues. Examples in

17. All computational experiments are performed in a shared cluster computing environment with highly variable demand, and therefore runtimes are not necessarily a reliable measure of problem complexity; hence, the number of nodes considered is always displayed. Further, Algorithm 2 is highly parallelizable, like many branch-and-bound algorithms; however, our implementation is serial. Therefore, with improvements in code design, it is very likely that runtimes can be substantially improved beyond those shown here.

		Root node			Terminal node			
Problem size (R/p)	Instance	Upper bound	CE LB	Weyl LB	Upper bound	Lower bound	Nodes	Time (s)
2/10	1	1.54	1.44	1.43	1.54	1.44	1	0.14
	2	1.50	1.40	1.40	1.50	1.40	1	0.38
	3	1.44	1.25	1.32	1.44	1.35	5	0.20
3/10	1	0.88	0.49	0.70	0.88	0.78	78	4.08
	2	0.49	0.25	0.39	0.49	0.39	1	1.27
	3	0.52	0.17	0.42	0.52	0.42	14	0.34
5/10	1	0.43	−0.90	0.10	0.43	0.33	28163	19432.70
	2	0.06	−0.75	0.00	0.06	0.00	1	0.20
	3	0.17	−0.98	0.00	0.17	0.07	3213	380.66
2/20	1	3.99	3.91	3.94	3.99	3.94	1	0.19
	2	4.64	4.58	4.60	4.64	4.60	1	2.07
	3	3.34	3.26	3.28	3.34	3.28	1	0.66
3/20	1	2.33	2.06	2.24	2.33	2.33	1	0.21
	2	2.55	2.38	2.49	2.55	2.49	1	6.62
	3	2.04	1.86	1.97	2.04	1.97	1	3.51
5/20	1	0.61	−0.07	0.49	0.61	0.51	626	75.98
	2	0.50	−0.01	0.40	0.50	0.40	41	2.88
	3	0.92	0.18	0.81	0.92	0.82	267	24.81

Table 2: Computational results for Algorithm 2 for class $C = A_1(R/p)$. All computations are performed in `julia` using SDO solvers `MOSEK` (for primal feasible solutions) and `SCS` (for dual feasible solutions within tolerance 10^{-3}). Computation time does not include preprocessing (such as computation of $\mathbf{u}$ as in (40) and finding an initial incumbent feasible solution $\mathbf{\Phi}_f$ as computed via the conditional gradient algorithm in Section 3). We always use default tolerance `TOL` $= 0.1$ for algorithm termination. Parameters for branching, pruning, node selection, etc., are detailed throughout Section 4. For each problem size, we run three instances (the instance number is the random seed provided to `julia`). Upper bounds denote z_f, which is the best feasible solution found thus far (either at the root node or at algorithm termination). At the root node, we display two lower bounds: the lower bound arising from convex envelopes (denoted "CE LB") and the one arising from the Weyl bound (denoted "Weyl LB"). Note that for lower bound at the termination node, we mean the worst bound $\max\{z^c, w^c\}$ (see Section 4.6; z^c is from the convex envelope approach, while w^c comes from Weyl's method). "Nodes" indicates the number of nodes considered in the course of execution, while "Time (s)" denotes the runtime (in seconds). We set r^*, the rank used within Algorithm 2, to $r^* = R - 1$, where R is the generative rank displayed. All results displayed to two decimals. Computations run on high-demand, shared cluster computing environment with variable architectures. Runtime is capped at 400000s (approximately 5 days), and any instance which is still running at that time is marked with an asterisk next to its runtime.

		Root node			Terminal node			
Problem size (R/p)	Instance	Upper bound	CE LB	Weyl LB	Upper bound	Lower bound	Nodes	Time (s)
3/50	1	7.16	7.09	7.14	7.16	7.14	1	18.66
	2	6.74	6.66	6.71	6.74	6.71	1	13.19
	3	6.78	6.74	6.77	6.78	6.77	1	33.28
5/50	1	3.04	2.86	3.01	3.04	3.01	1	19.13
	2	3.32	3.12	3.29	3.32	3.29	1	7.33
	3	3.70	3.56	3.67	3.70	3.67	1	52.53
10/50	1	0.88	−0.15	0.81	0.88	0.81	1	6.22
	2	1.20	0.08	1.12	1.20	1.12	1	17.41
	3	1.14	−0.02	1.07	1.14	1.07	1	8.88
3/100	1	13.72	13.69	13.72	13.72	13.72	1	125.17
	2	14.07	14.03	14.06	14.07	14.06	1	63.25
	3	14.67	14.64	14.66	14.67	14.66	1	28.36
5/100	1	8.54	8.46	8.53	8.54	8.53	1	117.71
	2	7.16	7.04	7.14	7.16	7.14	1	70.09
	3	7.07	6.94	7.06	7.07	7.06	1	78.60
10/100	1	2.62	2.10	2.58	2.62	2.58	1	36.91
	2	2.81	2.38	2.78	2.81	2.78	1	9.25
	3	3.16	2.64	3.12	3.16	3.12	1	49.04

Table 3: Computational results for larger examples from class $C = A_1(R/p)$. Same parameters as in Table 2 but with larger problem instances. Again we set $r^* = R - 1$.

these two classes do not have such a spectrum, and indeed Weyl's method does not provide the best root node bound.[18] Because neither Weyl's method nor the convex envelope bound strictly dominate one another at the root node across all examples, our approach incorporating both can leverage the advantages of each.

Observe that the root node gap (either in terms of the absolute difference between the initial feasible solution found and the better of the convex envelope bound and the Weyl bound) tends to be smaller when r^* is much smaller than p. This suggests that the approach we take is well-suited to certify optimality of particularly low-rank decompositions in noisy factor analysis settings. We see that this phenomenon is true across all classes. The numerical results suggest that corresponding theoretical guarantees for Weyl's method are potentially of interest in their own right and warrant further consideration.

Finally, we remark that if the true convex envelope of the objective over the set of semidefinite constraints was taken, then the convex envelope objective would always be non-negative. However, because we have taken the convex envelope of the objective over the polyhedral constraints only, this is not the case.

Performance at Termination: It is particularly encouraging that the initial feasible solutions provided via Algorithm 1 remain the best feasible solution throughout the execution of Algorithm 2 for all but three problem instances. (Of course, this need not be

18. However, it is worth remarking that Weyl's method still provides lower bounds on the rank of solutions to the noiseless factor analysis problem. Hence, even in settings where Weyl's method is not necessarily well-suited for proving optimality for the noisy factor analysis problem, it can still be applied successfully to lower bound rank for noiseless factor analysis.

			Root node			Terminal node			
Problem size (p)	Instance	r^* used	Upper bound	CE LB	Weyl LB	Upper bound	Lower bound	Nodes	Time (s)
10	1	2	0.98	−0.29	0.26	0.96	0.86	1955	91.99
	2		0.85	−0.28	0.29	0.85	0.75	736	37.70
	3		0.89	−0.29	0.23	0.89	0.79	1749	53.30
10	1	3	0.53	−0.43	0.13	0.53	0.43	3822	409.68
	2		0.58	−0.45	0.15	0.58	0.48	8813	1623.38
	3		0.64	−0.44	0.10	0.59	0.49	13061	1671.61
20	1	2	5.13	4.14	2.13	5.13	5.03	29724	36857.46
	2		4.68	3.44	2.00	4.68	4.58	28707	35441.37
	3		4.83	3.67	2.00	4.83	4.73	17992	11059.55
20	1	3	4.26	2.68	1.55	4.26	4.05	86687	400002.3*
	2		3.88	2.00	1.46	3.88	3.71	132721	400003.7*
	3		4.03	2.25	1.48	4.03	3.84	100042	400002.1*
50	1	3	17.83	16.54	11.49	17.83	17.44	35228	400002.5*
	2		18.57	17.34	12.19	18.57	18.20	34006	400007.1*
	3		18.21	17.06	11.96	18.21	17.86	35874	400004.9*
100	1	3	38.65	37.50	33.08	38.65	38.18	20282	400015.2*
	2		38.84	37.66	33.25	38.84	38.37	17306	400013.7*
	3		39.02	37.86	33.67	39.02	38.58	19889	400004.9*
100	1	5	30.98	29.05	25.69	30.98	29.91	16653	400005.6*
	2		31.37	29.39	26.07	31.37	30.32	26442	400021.4*
	3		31.20	29.24	26.13	31.20	30.15	19445	400043.4*

Table 4: Computational results for class $C = A_2(p)$. All parameters as per Table 2. Here we show the behavior across a variety of choices of the parameter r^*. Recall that the class A_2 is generated by a common variance component with high rank and exponentially decaying eigenvalues.

universally true.) This is an important observation to make because without a provable optimality scheme such as the one we consider here, it is difficult to quantify the performance of heuristic upper bound methods. As we demonstrate here, despite the only local guarantees of solutions obtained via a conditional gradient scheme, they tend to perform quite well in the setting of factor analysis. Indeed, even in the three examples where the best feasible solution is improved, the improved solution is found very early in the branching process.

Across the different example classes, we see that in general the gap tends to decrease more when r^* is small relative to p and p is smaller. To appropriately contextualize and appreciate the number of nodes solved for a problem with $p = 100$ on the timescale of 100s, with state-of-the-art implementations of interior point methods, solving a single node in the branch and bound tree can take on the order of 40s (for the specifically structured problems of interest and on the same machine). In other words, if one were to naïvely use interior point methods, it would only be possible to solve approximately three nodes during a 100s time limit. In contrast, by using a first-order method approach which facilitates warm starts, we are able to solve hundreds of nodes in the same amount of time.

We see that Algorithm 2 performs particularly well for classes A_1 (Tables 2 and 3) and B_1 (Table 5), for problems of reasonable size with relatively small underlying ranks. This

			Root node			Terminal node			
Problem size (r/p)	Instance	r^* chosen	Upper bound	CE LB	Weyl LB	Upper bound	Lower bound	Nodes	Time (s)
4/10	1	1	0.25	0.06	0.11	0.25	0.20	24	0.66
	2	1	0.17	−0.04	0.04	0.17	0.12	17	0.50
	3	1	0.16	−0.08	0.00	0.16	0.11	26	0.16
6/10	1	2	0.09	−0.27	0.00	0.09	0.04	77	0.62
	2	2	0.07	−0.36	0.00	0.07	0.02	118	0.75
	3	2	0.07	−0.36	0.00	0.07	0.02	125	3.79
10/10	1	3	0.10	−0.46	0.00	0.10	0.05	2162	166.44
	2	3	0.10	−0.62	0.00	0.10	0.05	3282	406.82
	3	4	0.06	−0.49	0.00	0.04	0.00	122	11.69
4/20	1	1	0.26	0.10	0.13	0.26	0.21	21	0.17
	2	1	0.19	0.01	0.07	0.19	0.14	15	0.40
	3	1	0.19	−0.04	0.04	0.19	0.14	26	0.59
6/20	1	2	0.10	−0.24	0.01	0.10	0.05	73	1.47
	2	2	0.08	−0.32	0.00	0.08	0.03	103	2.82
	3	2	0.08	−0.30	0.00	0.08	0.03	96	1.64
10/20	1	3	0.13	−0.44	0.00	0.13	0.08	2602	113.82
	2	3	0.12	−0.59	0.00	0.12	0.07	3971	188.93
	3	3	0.15	−0.45	0.01	0.15	0.10	3910	531.04
6/50	1	2	0.11	−0.15	0.04	0.11	0.06	50	0.66
	2	2	0.10	−0.18	0.01	0.10	0.05	61	1.95
	3	2	0.10	−0.20	0.01	0.10	0.05	64	0.58
10/50	1	3	0.17	−0.35	0.03	0.17	0.12	2283	333.62
	2	3	0.17	−0.46	0.02	0.17	0.12	4104	544.71
	3	3	0.19	−0.36	0.03	0.19	0.14	4362	620.61
6/100	1	2	0.12	−0.07	0.05	0.12	0.05	1	0.71
	2	2	0.11	−0.09	0.04	0.11	0.07	12	0.82
	3	2	0.11	−0.12	0.03	0.11	0.06	30	0.46
10/100	1	3	0.19	−0.22	0.06	0.19	0.14	1535	237.19
	2	3	0.19	−0.28	0.05	0.19	0.14	2625	383.27
	3	3	0.20	−0.26	0.05	0.20	0.15	2829	355.08

Table 5: Computational results for class $C = B_1(R/p)$. We choose r^* during computation as the largest r such that the CG method of Section 3 is strictly positive (up to additive tolerance 0.05; we use a smaller value here because the objective values are smaller across this class). For this class, examples can be preprocessed because $\mathbf{\Sigma} \sim B_1(R/p)$ has a block of size $R \times R$ in the upper left, with all other entries set to zero except the diagonal. Hence, it suffices to perform factor analysis with the truncated matrix $\mathbf{\Sigma}_{1:R,1:R}$.

is highly encouraging. Class A_1 forms a set of prototypical examples for which theoretical recovery guarantees perform well; in stark contrast, problems such as those in B_1 which have highly structured underlying factors tend to not satisfy the requirements of such recovery guarantees. Indeed, if $\mathbf{\Sigma} \sim B_1(R/p)$, then there generally appears to be a rank $R/2$ matrix $\mathbf{\Theta} \succcurlyeq \mathbf{0}$ so that $\mathbf{\Sigma} - \mathbf{\Theta}$ is positive semidefinite and diagonal. In such a problem, for r^* on the order of $R/2 - 1$, we provide nearly optimal solutions within a reasonable time frame.

Further, we note that similar results to those obtained for classes A_1 and B_1 are obtained for the other classes and are detailed in Tables 4 (class A_2), 6 (class B_2), 7 (class B_3), and

			Root node			Terminal node			
Problem size ($r/R/p$)	Instance	r^* chosen	Upper bound	CE LB	Weyl LB	Upper bound	Lower bound	Nodes	Time (s)
2/5/20	1	4	1.15	0.4	1.03	1.15	1.05	517	146.72
	2	4	1.08	0.50	0.97	1.08	0.98	366	60.56
	3	4	0.89	0.29	0.79	0.89	0.79	1	0.56
3/5/20	1	4	1.24	0.46	1.11	1.24	1.14	933	232.74
	2	4	0.99	0.55	0.92	0.99	0.92	1	0.87
	3	4	0.72	0.08	0.62	0.72	0.62	123	20.16
2/5/100	1	4	7.54	7.46	7.53	7.54	7.53	1	84.20
	2	4	6.45	6.34	6.44	6.45	6.44	1	119.62
	3	4	7.20	7.09	7.18	7.20	7.18	1	97.17
3/5/100	1	4	8.40	8.32	8.39	8.40	8.39	1	135.73
	2	4	5.79	5.70	5.78	5.79	5.78	1	38.19
	3	4	7.58	7.47	7.57	7.58	7.57	1	73.51
2/10/100	1	9	3.56	3.11	3.53	3.56	3.53	1	71.10
	2	9	3.00	2.61	2.97	3.00	2.97	1	85.47
	3	9	2.58	2.14	2.55	2.58	2.55	1	22.99
3/10/100	1	9	3.23	2.80	3.20	3.23	3.20	1	50.63
	2	9	2.94	2.56	2.91	2.94	2.91	1	51.99
	3	9	2.57	2.12	2.54	2.57	2.54	1	72.45

Table 6: Computational results for class $B_2(r/R/p)$. Parameters as set in Table 2. Here r^* is chosen during computation as largest k such that the CG method of Section 3 is strictly positive (up to 0.1). Here $r^* = R - 1$ ends up being an appropriate choice.

8 and 9. In a class such as A_2, which is generated as a high rank matrix (with decaying spectrum) with added individual variances, theoretical recovery guarantees do not generally apply, so again it is encouraging to see that our approach still makes significant progress towards proving optimality. Further, as shown in Table 11, for a variety of problems with p on the order of 1000 or 4000, solutions can be found in seconds and optimality can be certified within minutes via Weyl bounds, with no need for convex envelope bounds as computed via Algorithm 2, so long as the rank r^* is sufficiently small (for classes A_1, B_2, and B_3 on the order of hundreds, and for A_2 on the order of tens). This strongly supports the value of such an eigenvalue-based approach. When computing lower bounds solely via Weyl's method, the only necessary computations are solving quadratic programs (to find $\mathbf{u}$ as in Section 4.3) and an eigenvalue decomposition. As Table 11 suggests, for sufficiently small rank, one can still quickly find certifiably optimal solutions even for very large-scale factor analysis problems.

A particularly interesting example that stands apart from the rest is JO from the set of real examples. In particular, the correlation matrix for this example is not full rank (as there are fewer observations than variables). As displayed in Table 10, solutions found via conditional gradient methods are certified to be optimal at the root node. Indeed, the initial $\mathbf{u}_0$ as in Section 4.3 can be quickly verified to be $\mathbf{u}_0 = \mathbf{0}$ for this example, i.e., factor analysis is only possible with no individual variances. Hence, Algorithm 2 terminates at the root node without needing any branching.

			Root node			Terminal node			
Problem size ($r/R/p$)	Instance	r^* chosen	Upper bound	CE LB	Weyl LB	Upper bound	Lower bound	Nodes	Time (s)
2/5/20	1	3	0.30	−0.30	0.13	0.30	0.20	777	219.75
	2	3	0.22	−0.11	0.12	0.22	0.12	1	2.61
	3	3	0.19	−0.17	0.10	0.19	0.10	1	1.57
3/5/20	1	2	1.00	0.62	0.82	1.00	0.90	217	48.43
	2	2	0.95	0.47	0.74	0.95	0.85	305	26.14
	3	2	0.45	0.06	0.30	0.45	0.36	98	12.31
2/5/100	1	3	0.55	0.24	0.40	0.55	0.45	7099	47651.57
	2	3	0.28	−0.07	0.17	0.28	0.18	266	1431.88
	3	3	0.19	−0.07	0.11	0.19	0.11	1	574.61
3/5/100	1	2	1.03	0.70	0.81	1.03	0.93	4906	34514.69
	2	2	0.90	0.66	0.78	0.90	0.80	95	596.96
	3	2	0.55	0.39	0.47	0.55	0.47	1	1787.14
2/10/100	1	8	0.12	−0.58	0.05	0.12	0.05	1	77.38
	2	8	0.11	−0.53	0.05	0.11	0.05	1	41.24
	3	8	0.10	−0.43	0.05	0.10	0.05	1	110.17
3/10/100	1	6	0.33	−0.73	0.20	0.33	0.22	27770	400000.0*
	2	7	0.25	−0.43	0.17	0.25	0.17	1	128.63
	3	7	0.31	−0.28	0.24	0.31	0.24	1	213.72

Table 7: Computational results for class $B_3(r/R/p)$. Parameters as set in Table 2. As in Tables 5 and 6, here r^* is chosen during computation as largest r such that the CG method of Section 3 produces a feasible solution with strictly positive objective (up to 0.1).

	Root node			Terminal node			
r^* used	Upper bound	CE LB	Weyl LB	Upper bound	Lower bound	Nodes	Time (s)
1	4.06	3.78	2.53	4.06	3.96	44	0.64
2	2.64	2.04	1.42	2.64	2.54	1885	142.06
3	1.56	0.62	0.61	1.56	1.46	11056	2458.23
4	0.88	−0.33	0.28	0.88	0.78	66877	60612.61
5	0.36	−0.88	0.0	0.36	0.25	155759	400012.4*

Table 8: Computational results for the geomorphology example ($p = 10$). Parameters as set in Table 2. Here we display results for the choices of $r^* \in \{1, 2, 3, 4, 5\}$ (for $r^* > 5$, the upper bound is below the numerical tolerance, so we do not include those).

	Root node			Terminal node			
r^* used	Upper bound	CE LB	Weyl LB	Upper bound	Lower bound	Nodes	Time (s)
1	9.88	9.64	5.89	9.88	9.78	158	30.13
2	7.98	7.54	4.22	7.98	7.88	31710	49837.17
3	6.53	5.85	3.01	6.53	6.35	81935	400003.8*

Table 9: Computational results for the Harman example ($p = 24$). Parameters as set in Table 2. Here we display results for the choices of $r^* \in \{1, 2, 3\}$.

r^*	Optimal value	Time (s)
1	51.85	5.59
2	46.30	6.86
3	41.29	6.08
4	36.54	6.08
5	32.36	1.21
6	28.72	7.83
7	25.39	1.13
8	22.10	3.87
9	19.20	4.97
10	16.63	4.17
11	14.30	7.87

r^*	Optimal value	Time (s)
12	12.52	3.37
13	10.81	2.97
14	9.25	1.07
15	7.78	6.21
16	6.44	5.40
17	5.15	0.91
18	3.98	8.50
19	2.84	5.33
20	1.87	2.15
21	1.07	2.13
22	0.48	1.65

Table 10: Computational results for the example JO ($p = 58$). Parameters as set in Table 2. Here we display results for the choices of $r^* \in \{1, 2, \ldots, 22\}$ (for $r^* > 22$, the upper bound is below the numerical tolerance, so we do not include those). For all choices of r^*, the optimal solution is found and certified to be optimal at the root node, and therefore we do not display any information other than the optimal values.

Finally, we note that all synthetic examples we have considered have equal proportions of common and individual variances (although, of course, this is not exploited by our approach as this information is not *a priori* possible to specify without additional contextual information). If one modifies the classes so that the proportion of the common variances is higher than the proportion of individual variances (in the generative example), then Algorithm 2 is able to deliver better bounds on a smaller time scale. (Results are not included here.) This is not particularly surprising because the branch-and-bound approach we take essentially hinges on how well the products $W_{ii}\Phi_i$ can be approximated. When there is underlying factor structure with a lower proportion of individual variances, the scale of $W_{ii}\Phi_i$ is smaller and hence these products are easier to approximate well.

5.4 Additional Considerations

We now turn our attention to assessing the benefits of various algorithmic modifications as presented in Section 4. We illustrate the impact of these by focusing on four representative examples from across the classes: $A_1(3/10)$, $B_1(6/50)$, $B_3(3/5/20)$, and $G :=$ geomorphology.

Performance of Branching Strategy: We begin by considering the impact of our branching strategy as developed in Section 4.4. The results across the four examples are shown in Table 12. Recall that $\epsilon \in [0, 1)$ controls the extent of deviation from the canonical branching location, with $\epsilon = 0$ corresponding to no deviation. Across all examples, we see that the number of nodes considered to prove optimality is approximately convex in $\epsilon \in [0, 0.5]$. In particular, for all examples, the "optimal" choice of ϵ is not the canonical choice of $\epsilon = 0$. This contrast is stark for the examples $B_3(3/5/20)$ and G. Indeed, for these two examples, the number of nodes considered when $\epsilon = 0$ is over five times larger than for any $\epsilon \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$.

Example	r^*	Upper bound	Lower bound
$A_1(100/1000)$	10	460.35	457.46
	30	322.35	313.62
	50	198.70	197.13
	70	103.72	102.78
	90	28.65	28.34
$A_2(1000)$	10	184.22	183.18
	20	61.24	60.34
	30	20.25	19.50
	40	6.63	6.02
	50	2.17	1.71
$B_2(10/90/1000)$	20	379.02	377.23
	40	236.10	234.83
	60	122.08	121.33
	80	34.49	34.24
$B_2(20/90/1000)$	20	378.37	376.64
	40	235.16	233.94
	60	121.69	120.96
	80	34.33	34.09
$B_3(10/50/1000)$	10	384.40	384.04
	20	233.25	232.97
	30	105.68	105.49
	40	0.69	0.60
$B_3(20/150/1000)$	20	426.31	422.13
	45	285.56	282.33
	70	174.00	171.70
	95	86.54	85.15
	120	20.30	19.80

Example	r^*	Upper bound	Lower bound
$A_1(360/4000)$	100	1334.40	1327.60
	150	991.64	986.15
	200	693.02	688.87
	250	434.64	431.80
	300	213.29	211.75
	350	30.75	30.50
$A_2(4000)$	10	733.41	733.09
	20	244.13	243.85
	30	79.94	79.68
	40	26.17	25.96
	50	8.57	8.39
	60	2.80	2.66
	70	0.90	0.79
$B_2(80/360/4000)$	100	1360.1	1354.04
	150	1008.90	1004.04
	200	703.78	700.21
	250	440.56	438.16
	300	215.87	214.62
	350	31.02	30.91
$B_3(120/360/4000)$	100	1081.20	1078.74
	150	630.25	628.54
	200	253.50	252.50
	240	7.89	7.46

Table 11: Computational results across several classes for larger scale instances with $p \in \{1000, 4000\}$. Here for a given example, the random seed is set as 1. Results are displayed across a variety of choices of rank r^*. Class B_1 is not shown because of the reduction noted in Table 5 (which reduces this class to a much smaller-dimensional class, and hence it is not truly large scale). The "Upper bound" denotes the upper bound found by the conditional gradient method, while "Lower bound" denotes the Weyl bound at the root node (no convex envelope bounds via Algorithm 2 are shown here because of the large nature of the SDO-based convex envelope lower bounds which are prohibitive to solve for problems of this size). These instances are completed in `MATLAB` because of its superior support (over `julia`) for solving large-scale quadratic optimization problems, which is the only computation necessary for computing Weyl bounds.

Example	r^*	Nodes considered for $\epsilon =$					
		0.0	0.1	0.2	0.3	0.4	0.5
$A_1(3/10)$	2	103	72	59	62	78	101
$B_1(6/50)$	2	39	33	29	40	50	53
$B_3(3/5/20)$	2	8245	762	278	189	217	203
geomorphology	1	937	162	59	46	44	52

Table 12: Computational results for effect of branching strategy across different examples. In particular, we consider how the number of nodes needed to prove optimality (up to numerical tolerance) changes across different choices of ϵ, where ϵ is as in Section 4.4. Recall that $\epsilon = 0$ corresponds to a naïve choice of branching, while $\epsilon > 0$ corresponds to a modified branching. All other parameters as in Table 2, including the branching index. For synthetic examples, the instances are initialized with random seed 1.

Example	r^*	Nodes considered for	
		Naïve strategy	Modified strategy
$A_1(3/10)$	2	99	78
$B_1(6/50)$	2	135	50
$B_3(3/5/20)$	2	375	217
geomorphology	1	43	44

Table 13: Computational results for effect of node selection strategy across different examples. In particular, we consider how the number of nodes needed to prove optimality (up to numerical tolerance) changes across the naïve and modified branching strategies as described in Section 4.4. All other parameters as in Table 2. For synthetic examples, the instances are initialized with random seed 1.

Example	r^*	Upper Bound	CE LB with tightening	CE LB without tightening
$A_1(3/10)$	2	0.88	0.70	0.39
$B_1(6/50)$	2	0.11	−0.15	−0.17
$B_3(3/5/20)$	2	1.00	0.62	0.51
geomorphology	1	4.06	3.78	3.78

Table 14: Computational results for effect of root node bound tightening across different examples. In particular, we consider how the convex envelope lower bound (denoted "CE LB") compares with and without bound tightening at the root node (see Section 4.5). All other parameters as in Table 2. "Upper bound" displayed is as in Table 2 and is included for scale (i.e., to compare the relative impact of tightening). For synthetic examples, the instances are initialized with random seed 1.

Example	r^*	Nodes considered for TOL =		
		0.10	0.05	0.025
$A_1(3/10)$	2	78	287	726
$B_1(6/50)$	2	1	50	262
$B_3(3/5/20)$	2	217	1396	5132
geomorphology	1	44	217	978

Table 15: Computational results for numerical tolerance TOL across different examples. In particular, we consider how the number of nodes needed to prove optimality changes as a function of the additive numerical tolerance TOL chosen for algorithm termination. All other parameters as in Table 2. For synthetic examples, the instances are initialized with random seed 1.

In other words, the alternative branching strategy can have a substantial impact on the number of nodes considered in the branch-and-bound tree. As a direct consequence, this strategy can drastically reduce the computation time needed to prove optimality. As the examples suggest, it is likely that ϵ should be chosen dynamically during algorithm execution, as the particular choice depends on a given problem's structure. However, we set $\epsilon = 0.4$ for all other computational experiments because this appears to offer a distinct benefit over the naïve strategy of setting $\epsilon = 0$.

Performance of Node Selection Strategy: We now turn our attention to the node selection strategy as detailed in Section 4.6. Recall that node selection considers how to pick which node to consider next in the current branch-and-bound tree at any iteration of Algorithm 2. We compare two strategies: the naïve strategy which selects the node with worst convex envelope bound (as explicitly written in Algorithm 2) and the modified strategy which employs randomness and Weyl bounds to consider nodes which might not be advantageous to fathom when only convex envelope bounds are considered.

The comparison is shown in Table 13. We see that this strategy is advantageous overall (with only the example G giving a negligible decrease in performance). The benefit is particularly strong for examples from the B classes which have highly structured underlying factors. For such examples, there is a large difference between the convex envelope bounds and the Weyl bounds at the root node (see e.g. Table 5). Hence, an alternative branching strategy which incorporates the eigenvalue information provided by Weyl bounds has potential to improve beyond the naïve strategy. Indeed, this appears to be the case across all examples where such behavior occurs.

Performance of Bound Tightening All computational results employ bound tightening as developed in Section 4.5, but only at the root node. Bound tightening, which requires repeated eigenvalue computations, is a computationally expensive process. For this reason, we have chosen not to employ bound tightening at every node in the branch-and-bound tree. From a variety of computational experiments, we observed that the most important node for bound tightening is the root node, and therefore it is a reasonable choice to only employ bound tightening there. Consequently, we employ pruning via Weyl's method (detailed in Section 4.5 as well) at all nodes in the branch-and-bound tree. (Recall that bound tightening can be thought of as optimal pruning via Weyl's method.)

In Table 14 we show the impact of bound tightening at the root node in terms of the improvement in the lower bounds provided by convex envelopes. The results for the class A_1 are particularly distinctive. Indeed, for this class bound tightening has a substantial impact on the quality of the convex envelope bound (for the example $A_1(3/10)$ given, the improvement is from a relative gap at the root node of 56% to a gap of 20%). For the examples shown, bound tightening offers the least improvement in the real example geomorphology. In light of Table 8 this is not too surprising, as Weyl's method (at the root node) is not particular effective for this example. As Weyl's inequality is central to bound tightening, problems for which Weyl's inequality is not particularly effective tend to experience less benefit from bound tightening at the root node.

Influence of Optimality Tolerance We close this section by assessing the influence of the optimality tolerance for termination, TOL. In particular, we study how the number of

nodes to prove optimality changes as a function of additive gap necessary for termination. The corresponding results across the four examples are shown in Table 15. Not surprisingly, as the gap necessary for termination is progressively halved, the corresponding number of nodes considered increases substantially. However, it is important to note that even though the gap at termination is smaller as this tolerance decreases (by design), for these examples the best feasible solution remains unchanged. In other words, the increase in the number of nodes is the price for more accurately proving optimality and not for finding better feasible solutions. Indeed, as noted earlier, the solutions found via conditional gradient methods at the outset are of remarkably high quality.

6. Conclusions

We analyze the classical rank-constrained FA problem from a computational perspective. We proposed a general, flexible family of rank-constrained, nonlinear SDO-based formulations for the task of approximating an observed covariance matrix $\mathbf{\Sigma}$ as the sum of a PSD low-rank component $\mathbf{\Theta}$ and a diagonal matrix $\mathbf{\Phi}$ (with nonnegative entries) subject to $\mathbf{\Sigma}-\mathbf{\Phi}$ being PSD. Our framework enables us to estimate the underlying factors and unique variances under the restriction that the residual covariance matrix is semidefinite—this is important for statistical interpretability and understanding the proportion of variance explained by a given number of factors. This constraint, however, seems to ignored by most other widely used methods in FA.

We introduce a novel *exact* reformulation of the rank-constrained FA problem as a smooth optimization problem with convex, compact constraints. We present a unified algorithmic framework, utilizing modern techniques in nonlinear optimization and first order methods in convex optimization to obtain high-quality solutions for the FA problem. At the same time, we use techniques from discrete and global optimization to demonstrate that these solutions are often provably optimal. We provide computational evidence demonstrating that the methods proposed herein provide high quality estimates with improved accuracy when compared to existing, popularly-used methods in FA.

In this work we have demonstrated that a previously intractable rank optimization problem can be solved to provable optimality. We envision that ideas similar to those used here can be used to address an even larger class of estimation problems with underlying matrix structure. In this way, we anticipate significant progress on such problems in the next decade, particularly in light of myriad advances throughout distinct areas of modern optimization.

Acknowledgments

Copenhaver is supported by the U.S. Department of Defense, Office of Naval Research, through the National Defense Science and Engineering Graduate (NDSEG) Fellowship. Mazumder is partially supported by ONR Grant N000141512342. We gratefully acknowledge the thoughtful and detailed feedback of two anonymous referees which has led to a substantial improvement in the quality, organization, and completeness of the paper. Copenhaver would also like to thank the JuliaOpt team, especially Joey Huchette, for assistance with experimental features in the `julia` language.

Appendix A

This appendix contains all proofs for results presented in the main text.

Proof of Proposition 1:

(a) We start by observing that for any two real symmetric matrices $\mathbf{A}, \mathbf{B}$ (with dimensions $p \times p$) and the matrix q-norm, a result due to Mirsky (this is also known as the q-Wielandt-Hoffman inequality; see e.g. Stewart and Sun 1990, p. 205) states:

$$\|\mathbf{A}-\mathbf{B}\|_q \geq \|\boldsymbol{\lambda}(\mathbf{A})-\boldsymbol{\lambda}(\mathbf{B})\|_q, \tag{46}$$

where $\boldsymbol{\lambda}(\mathbf{A})$ and $\boldsymbol{\lambda}(\mathbf{B})$ denote the vector of eigenvalues of $\mathbf{A}$ and $\mathbf{B}$, respectively, arranged in decreasing order, i.e., $\lambda_1(\mathbf{A}) \geq \lambda_2(\mathbf{A}) \geq \ldots \geq \lambda_p(\mathbf{A})$ and $\lambda_1(\mathbf{B}) \geq \lambda_2(\mathbf{B}) \geq \ldots \geq \lambda_p(\mathbf{B})$. Using this result for Problem (4), it is easy to see that for fixed $\mathbf{\Phi}$

$$\{\mathbf{\Theta} : \mathbf{\Theta} \succeq \mathbf{0}, \text{rank}(\mathbf{\Theta}) \leq r\} = \{\mathbf{\Theta} : \boldsymbol{\lambda}(\mathbf{\Theta}) \geq \mathbf{0}, \|\boldsymbol{\lambda}(\mathbf{\Theta})\|_0 \leq r\}, \tag{47}$$

where $\|\boldsymbol{\lambda}(\mathbf{\Theta})\|_0$ counts the number of non-zero elements of $\boldsymbol{\lambda}(\mathbf{\Theta})$. If we partially minimize the objective function in Problem (4), with respect to $\mathbf{\Theta}$ (with $\mathbf{\Phi}$ fixed), and use (46) along with (47), we have the following inequality:

$$\begin{array}{rll} & \inf\limits_{\mathbf{\Theta}} & \|(\mathbf{\Sigma}-\mathbf{\Phi})-\mathbf{\Theta}\|_q^q \quad \text{s.t.} \quad \mathbf{\Theta} \succeq \mathbf{0}, \text{rank}(\mathbf{\Theta}) \leq r \\ \geq & \inf\limits_{\boldsymbol{\lambda}(\mathbf{\Theta})} & \|\boldsymbol{\lambda}(\mathbf{\Sigma}-\mathbf{\Phi})-\boldsymbol{\lambda}(\mathbf{\Theta})\|_q^q \quad \text{s.t.} \quad \boldsymbol{\lambda}(\mathbf{\Theta}) \geq \mathbf{0}, \|\boldsymbol{\lambda}(\mathbf{\Theta})\|_0 \leq r. \end{array} \tag{48}$$

Since $\mathbf{\Sigma}-\mathbf{\Phi} \succeq \mathbf{0}$, it follows that the minimum objective value of the r.h.s. optimization Problem (48) is given by $\sum_{i=r+1}^p \lambda_i^q(\mathbf{\Sigma}-\mathbf{\Phi})$ and is achieved for $\lambda_i(\mathbf{\Theta}) = \lambda_i(\mathbf{\Sigma}-\mathbf{\Phi})$ for $i = 1, \ldots, r$. This leads to the following inequality:

$$\inf_{\mathbf{\Theta}:\mathbf{\Theta}\succeq\mathbf{0},\text{rank}(\mathbf{\Theta})\leq r} \|(\mathbf{\Sigma}-\mathbf{\Phi})-\mathbf{\Theta}\|_q^q \geq \sum_{i=r+1}^p \lambda_i^q(\mathbf{\Sigma}-\mathbf{\Phi}). \tag{49}$$

Furthermore, if $\mathbf{U}_{p\times p}$ denotes the matrix of p eigenvectors of $\mathbf{\Sigma}-\mathbf{\Phi}$, then the following choice of $\mathbf{\Theta}^*$:

$$\mathbf{\Theta}^* := \mathbf{U}\ \text{diag}\left(\lambda_1(\mathbf{\Sigma}-\mathbf{\Phi}), \ldots, \lambda_r(\mathbf{\Sigma}-\mathbf{\Phi}), 0, \ldots, 0\right)\ \mathbf{U}', \tag{50}$$

gives equality in (49). This leads to the following result:

$$\inf_{\mathbf{\Theta}:\mathbf{\Theta}\succeq\mathbf{0},\text{rank}(\mathbf{\Theta})\leq r} \|\mathbf{\Sigma}-(\mathbf{\Theta}+\mathbf{\Phi})\|_q^q = \|\mathbf{\Sigma}-(\mathbf{\Theta}^*+\mathbf{\Phi})\|_q^q = \sum_{i=r+1}^p \lambda_i^q(\mathbf{\Sigma}-\mathbf{\Phi}). \tag{51}$$

(b) The minimizer $\mathbf{\Theta}^*$ of Problem (51) is given by (50). In particular, if $\mathbf{\Phi}^*$ solves Problem (12) and we compute $\mathbf{\Theta}^*$ via (50) (with $\mathbf{\Phi} = \mathbf{\Phi}^*$), then the tuple $(\mathbf{\Phi}^*, \mathbf{\Theta}^*)$ solves Problem (4). This completes the proof of the proposition.

■

Proof of Proposition 2: We build upon the proof of Proposition 1. Note that any $\mathbf{\Phi}$ that is feasible for Problems (14) and (4) is PSD. Observe that $\mathbf{\Theta}^*$ (appearing in the proof of Proposition 1) as given by (50) satisfies:

$$\mathbf{\Sigma} - \mathbf{\Phi} - \mathbf{\Theta}^* \succeq \mathbf{0} \implies \mathbf{\Sigma} - \mathbf{\Theta}^* \succeq \mathbf{0}, \tag{52}$$

where, the right hand side of (52) follows since $\mathbf{\Phi} \succeq \mathbf{0}$. We have thus established that the solution $\mathbf{\Theta}^*$ to the following problem

$$\begin{array}{ll} \underset{\mathbf{\Theta}}{\text{minimize}} & \|(\mathbf{\Sigma} - \mathbf{\Phi}) - \mathbf{\Theta}\|_q^q \\ \text{s.t.} & \mathbf{\Theta} \succeq \mathbf{0} \\ & \text{rank}(\mathbf{\Theta}) \leq r, \end{array} \tag{53}$$

is feasible for the following optimization problem:

$$\begin{array}{ll} \underset{\mathbf{\Theta}}{\text{minimize}} & \|(\mathbf{\Sigma} - \mathbf{\Phi}) - \mathbf{\Theta}\|_q^q \\ \text{s.t.} & \mathbf{\Theta} \succeq \mathbf{0} \\ & \text{rank}(\mathbf{\Theta}) \leq r \\ & \mathbf{\Sigma} - \mathbf{\Theta} \succeq \mathbf{0}. \end{array} \tag{54}$$

Since Problem (54) involves minimization over a subset of the feasible set of Problem (53), it follows that $\mathbf{\Theta}^*$ is also a minimizer for Problem (54). This completes the proof of the equivalence. ■

Proof of Theorem 1:

(a) The proof is based on ideas appearing in Overton and Womersley (1992), where it was shown that the sum of the top r eigenvalues of a real symmetric matrix can be written as the solution to a linear SDO problem.

By an elegant classical result due to Fan (Stewart and Sun, 1990), the smallest $(p-r)$ eigenvalues of a real symmetric matrix $\mathbf{A}$ can be written as

$$\begin{array}{rll} \sum_{i=r+1}^{p} \lambda_i(\mathbf{A}) = & \inf & \text{Tr}\,(\mathbf{V}'\mathbf{A}\mathbf{V}) \\ & \text{s.t.} & \mathbf{V}'\mathbf{V} = \mathbf{I}, \end{array} \tag{55}$$

where the optimization variable is $\mathbf{V} \in \mathbb{R}^{p \times (p-r)}$. We will show that the solution to the above nonconvex problem can be obtained via the following linear (convex) SDO problem:

$$\begin{array}{ll} \text{minimize} & \text{Tr}\,(\mathbf{W}\mathbf{A}) \\ \text{s.t.} & \mathbf{I} \succeq \mathbf{W} \succeq \mathbf{0} \\ & \text{Tr}(\mathbf{W}) = p - r. \end{array} \tag{56}$$

Clearly, Problem (56) is a convex relaxation of Problem (55)—hence its minimum value is at least smaller than $\sum_{i=r+1}^{p} \lambda_i(\mathbf{A})$. By standard results in convex analysis (Rockafellar, 1996), it follows that the minimum of the above linear SDO problem (56) is attained at the extreme points of the feasible set of (56). The extreme points (see for example, Overton and Womersley, 1992; Vandenberghe and Boyd, 1996) of this set are given by the set of orthonormal matrices of rank $p-r$:

$$\left\{\mathbf{V}\mathbf{V}' : \mathbf{V} \in \mathbb{R}^{p \times (p-r)} : \mathbf{V}'\mathbf{V} = \mathbf{I}\right\}.$$

It thus follows that the (global) minima of Problems (56) and (55) are the same. Applying this result to the PSD matrix $\mathbf{A} = (\mathbf{\Sigma} - \mathbf{\Phi})^q$ appearing in the objective of (12), we arrive at (16). This completes the proof of part **(a)**.

(b) The statement follows from (50).

■

Proof of Proposition 3: Observe that, for every fixed $\mathbf{\Phi}$, the function $g_q(\mathbf{W}, \mathbf{\Phi})$ is concave (in fact, it is linear). Since $G_q(\mathbf{W})$ is obtained by taking a point-wise infimum (with respect to $\mathbf{\Phi}$) of the concave function $g_q(\mathbf{W}, \mathbf{\Phi})$, the resulting function $G_q(\mathbf{W})$ is concave (Boyd and Vandenberghe, 2004).

The expression of the sub-gradient (19) is an immediate consequence of Danskin's Theorem (Bertsekas, 1999; Rockafellar, 1996). ■

Proof of Theorem 2: Using the concavity of the function $G_q(\mathbf{W})$ we have:

$$\begin{array}{rcl} G_q(\mathbf{W}^{(i+1)}) & \leq & G_q(\mathbf{W}^{(i)}) + \langle \nabla G_q(\mathbf{W}^{(i)}), \mathbf{W}^{(i+1)} - \mathbf{W}^{(i)} \rangle \\ & = & G_q(\mathbf{W}^{(i)}) + \Delta(\mathbf{W}^{(i)}). \end{array} \tag{57}$$

Note that $\Delta(\mathbf{W}^{(i)}) \leq 0$ and $G_q(\mathbf{W}^{(i+1)}) \leq G_q(\mathbf{W}^{(i)})$ for all $i \leq k$. Hence the (decreasing) sequence of objective values converge to $G_q(\mathbf{W}^{(\infty)})$ (say) and $\Delta(\mathbf{W}^{(\infty)}) \geq 0$. Adding up the terms in (57) from $i = 1, \ldots, k$ we have:

$$G_q(\mathbf{W}^{(\infty)}) - G_q(\mathbf{W}^{(1)}) \leq G_q(\mathbf{W}^{(k+1)}) - G_q(\mathbf{W}^{(1)}) \leq -k \min_{i=1,\ldots,k} \left\{-\Delta(\mathbf{W}^{(k)})\right\} \tag{58}$$

from which (26) follows. ■

Proof of Theorem 5: By construction,

$$\max_{[\boldsymbol{\ell},\mathbf{u}] \in \texttt{Nodes}} \|\mathbf{u} - \boldsymbol{\ell}\|_1$$

decreases monotonically to zero as the iterative scheme proceeds. Therefore, to show convergence it suffices to show that the additive error at a node $\texttt{n} = [\boldsymbol{\ell}, \mathbf{u}] \in \texttt{Nodes}$ is $O(\|\mathbf{u} - \boldsymbol{\ell}\|_1)$. Fix an arbitrary node $\texttt{n} = [\boldsymbol{\ell}, \mathbf{u}] \in \texttt{Nodes}$. Without loss of generality, we may assume that (CFA_1) as restricted to $\text{diag}(\mathbf{\Phi}) \in [\boldsymbol{\ell}, \mathbf{u}]$ has a feasible solution; as

such, let $\mathbf{\Phi}^*$ be an optimal solution. Likewise, let $\mathbf{\Phi}^\dagger$ be an optimal solution ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$). First note that $\sum_{i=r+1}^p \sigma_i(\mathbf{\Sigma}-\mathbf{\Phi}^*) \leq \sum_{i=r+1}^p \sigma_i(\mathbf{\Sigma}-\mathbf{\Phi}^\dagger)$. Now, by two applications of the bound in Theorem 4 and the fact that $\mathbf{\Phi}^\dagger$ is optimal to ($\text{LS}_{\boldsymbol{\ell},\mathbf{u}}$), we have that $\sum_{i=r+1}^p \sigma_i(\mathbf{\Sigma}-\mathbf{\Phi}^\dagger) \leq \sum_{i=r+1}^p \sigma_i(\mathbf{\Sigma}-\mathbf{\Phi}^*) + \|\mathbf{u}-\boldsymbol{\ell}\|_1/2$. Hence, the additive error satisfies

$$\left| \sum_{i=r+1}^p \sigma_i(\mathbf{\Sigma}-\mathbf{\Phi}^\dagger) - \sum_{i=r+1}^p \sigma_i(\mathbf{\Sigma}-\mathbf{\Phi}^*) \right| \leq \|\mathbf{u}-\boldsymbol{\ell}\|_1/2,$$

as was to be shown. ■

Appendix B

This appendix contains an alternative conditional gradient method for finding feasible solutions. Since Problem (16) involves the minimization of a smooth function over a compact convex set, the CG method requires iteratively solving the following convex optimization problem:

$$\begin{array}{ll} \text{minimize} & \left\langle \nabla g_p(\mathbf{W}^{(k)},\mathbf{\Phi}^{(k)}), \begin{pmatrix}\mathbf{W}\\ \mathbf{\Phi}\end{pmatrix} \right\rangle \\ \text{s.t.} & \mathbf{W} \in \mathbf{\Psi}_{p,p-r} \\ & \mathbf{\Phi} \in \mathcal{F}_{\mathbf{\Sigma}}, \end{array} \tag{59}$$

where $\nabla g_p(\mathbf{W}^{(k)},\mathbf{\Phi}^{(k)})$ is the derivative of $g_p(\mathbf{W}^{(k)},\mathbf{\Phi}^{(k)})$ at the current iterate ($\mathbf{W}^{(k)}$, $\mathbf{\Phi}^{(k)}$). Note that due to the separability of the constraints in $\mathbf{W}$ and $\mathbf{\Phi}$, Problem (59) splits into two independent optimization problems with respect to $\mathbf{W}$ and $\mathbf{\Phi}$. The overall procedure is outlined in Algorithm 3.

Since $\nabla_{\mathbf{W}} g_p(\mathbf{W}^{(k)},\mathbf{\Phi}^{(k)}) = (\mathbf{\Sigma}-\mathbf{\Phi}^{(k)})^q$, the update for $\mathbf{W}$ appearing in (60) requires solving:

$$\underset{\mathbf{W}\in\mathbf{\Psi}_{p,p-r}}{\text{minimize}} \quad \langle \mathbf{W}, (\mathbf{\Sigma}-\mathbf{\Phi}^{(k)})^q \rangle. \tag{63}$$

Similarly, the update for $\mathbf{\Phi}$ appearing in (61) requires solving:

$$\underset{\mathbf{\Phi}\in\mathcal{F}_{\mathbf{\Sigma}}}{\text{minimize}} \quad \sum_{i=1}^p \Phi_i \ell_i, \tag{64}$$

where the vector $(\ell_1,\ldots,\ell_p)$ is given by $\text{diag}(\ell_1,\ldots,\ell_p) = -q \ \text{diag}\left(\mathbf{W}^{(k)}(\mathbf{\Sigma}-\mathbf{\Phi})^{q-1}\right)$, where $\text{diag}(\mathbf{A})$ is a diagonal matrix having the same diagonal entries as $\mathbf{A}$. The sequence $(\mathbf{W}^{(k)},\mathbf{\Phi}^{(k)})$ is recursively computed via Algorithm 3 until a convergence criterion is met:

$$g_p(\mathbf{W}^{(k)},\mathbf{\Phi}^{(k)}) - g_p(\mathbf{W}^{(k+1)},\mathbf{\Phi}^{(k+1)}) \leq \text{TOL}\cdot g_p(\mathbf{W}^{(k)},\mathbf{\Phi}^{(k)}), \tag{65}$$

for some user-defined tolerance $\text{TOL} > 0$.

Algorithm 3 A CG based algorithm for Problem (16)

1 Initialize with $(\mathbf{W}^{(1)},\mathbf{\Phi}^{(1)})$, feasible for Problem (16) and repeat the following Steps 2-3 until the convergence criterion described in (65) is met.

2 Solve the linearized Problem (59), which requires solving two separate (convex) SDO problems over $\mathbf{W}$ and $\mathbf{\Phi}$:

$$\overline{\mathbf{W}}^{(k+1)} \in \underset{\mathbf{W}\in\mathbf{\Psi}_{p,p-r}}{\arg\min} \ \langle \mathbf{W}, \nabla_{\mathbf{W}} g_p(\mathbf{W}^{(k)},\mathbf{\Phi}^{(k)}) \rangle \tag{60}$$

$$\overline{\mathbf{\Phi}}^{(k+1)} \in \underset{\mathbf{\Phi}\in\mathcal{F}_{\mathbf{\Sigma}}}{\arg\min} \ \langle \mathbf{\Phi}, \nabla_{\mathbf{\Phi}} g_p(\mathbf{W}^{(k)},\mathbf{\Phi}^{(k)}) \rangle \tag{61}$$

where $\nabla_{\mathbf{W}} g_p(\mathbf{W},\mathbf{\Phi})$ (and $\nabla_{\mathbf{\Phi}} g_p(\mathbf{W},\mathbf{\Phi})$) is the partial derivative with respect to $\mathbf{W}$ (respectively, $\mathbf{\Phi}$).

3 Obtain the new iterates:

$$\begin{array}{rcl} \mathbf{W}^{(k+1)} & = & \mathbf{W}^{(k)} + \eta_k(\overline{\mathbf{W}}^{(k+1)} - \mathbf{W}^{(k)}), \\ \mathbf{\Phi}^{(k+1)} & = & \mathbf{\Phi}^{(k)} + \eta_k(\overline{\mathbf{\Phi}}^{(k+1)} - \mathbf{\Phi}^{(k)}). \end{array} \tag{62}$$

with $\eta_k \in [0,1]$ chosen via an Armijo type line-search rule (Bertsekas, 1999).

A tuple $(\mathbf{W}^*,\mathbf{\Phi}^*)$ satisfies the first order stationarity conditions (Bertsekas, 1999) for Problem (16), if the following condition is satisfied:

$$\begin{array}{lll} \inf & \left\langle \nabla g_p(\mathbf{W}^*,\mathbf{\Phi}^*), \begin{pmatrix}\mathbf{W}-\mathbf{W}^*\\ \mathbf{\Phi}-\mathbf{\Phi}^*\end{pmatrix} \right\rangle & \geq \quad 0 \\ \text{s.t.} & \mathbf{W} \in \mathbf{\Psi}_{p,p-r} & \\ & \mathbf{\Phi} \in \mathcal{F}_{\mathbf{\Sigma}}. & \end{array} \tag{66}$$

Note that $\mathbf{\Phi}^*$ defined above also satisfies the first order stationarity conditions for Problem (12).

The following theorem presents a global convergence guarantee for Algorithm 3:

Theorem 8 *(Bertsekas, 1999) Every limit point* $(\mathbf{W}^{(\infty)},\mathbf{\Phi}^{(\infty)})$ *of a sequence* $(\mathbf{W}^{(k)},\mathbf{\Phi}^{(k)})$ *produced by Algorithm 3 is a first order stationary point of the optimization Problem* (16).

Numerical experiments (in line with those from Section 5) suggest that Algorithm 3 performs similarly to Algorithm 1, and therefore we only present the results for Algorithm 1 in the main text. Algorithm 1 has the advantage that it does not require a line search, unlike Algorithm 3. Finally, we note that for Algorithm 3 the update for $\mathbf{W}$ at iteration k for solving Problem (63) corresponds to $\widetilde{\mathbf{W}} = (\mathbf{\Sigma}-\mathbf{\Phi}^{(k)})^q$.

References

F.A. Al-Khayyal and J.E. Falk. Jointly constrained biconvex programming. *Mathematics of Operations Research*, 8:273–286, 1983.

E.D. Andersen and K.D. Andersen. *High Performance Optimization*, chapter The Mosek Interior Point Optimizer for Linear Programming: An Implementation of the Homogeneous Algorithm. Springer, 2000.

T. Anderson. *An Introduction to Multivariate Statistical Analysis.* Wiley, New York, 3rd edition, 2003.

K.M. Anstreicher and S. Burer. Computable representations for convex hulls of low-dimensional quadratic forms. *Mathematical Programming, Series B*, 124:33–43, 2010.

J. Bai and K. Li. Statistical analysis of factor models of high dimension. *The Annals of Statistics*, 40(1):436–465, 2012.

J. Bai and S. Ng. Large dimensional factor analysis. *Foundations and Trends in Econometrics*, 3(2):89–163, 2008.

L. Bai, J.E. Mitchell, and J.-S. Pang. On conic QPCCs, conic QCQPs and completely positive programs. *Mathematical Programming*, 159(1-2):109–136, 2016.

X. Bao, N.V. Sahinidis, and M. Tawarmalani. Multiterm polyhedral relaxations for nonconvex, quadratically constrained quadratic programs. *Optimization Methods and Software*, 24:485–504, 2009.

D. Bartholomew, M. Knott, and I. Moustaki. *Latent Variable Models and Factor Analysis: A Unified Approach.* Wiley, 2011.

A. Barvinok. Problems of distance geometry and convex properties of quadratic maps. *Discrete and Computational Geometry*, 12:189–202, 1995.

P.M. Bentler and J.A. Woodward. Inequalities among lower-bounds to reliability: With applications to test construction and factor analysis. *Psychometrika*, 45:249–267, 1980.

C. Bernaards and R. Jennrich. Gradient projection algorithms and software for arbitrary rotation criteria in factor analysis. *Educational and Psychological Measurement*, 65:676–696, 2005.

D. Bertsekas. *Nonlinear Programming.* Athena Scientific, 2nd edition, 1999.

R. Bhatia. *Perturbation bounds for matrix eigenvalues.* SIAM, 2007.

S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 3(1):1042–1068, 2011.

S. Burer. *Handbook on Semidefinite, Conic and Polynomial Optimization*, chapter Copositive programming, pages 201–218. Springer, 2012.

G. Connor and R. Korajczyk. Performance measurement with the arbitrage pricing theory: A new framework for analysis. *Journal of Financial Economics*, 15(3):373–394, 1986.

A. Costa and L. Liberti. Relaxations of multilinear convex envelopes: dual is better than primal. In *International Symposium on Experimental Algorithms*, pages 87–98. Springer, 2012.

M. Fazel. *Matrix Rank Minimization with Applications.* PhD thesis, Stanford University, 2002.

C.A. Floudas. *Deterministic global optimization: theory, algorithms, and applications.* Kluwer, 1999.

M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956.

L. Guttman. To what extent can communalities reduce rank? *Psychometrika*, 23(4):297–308, 1958.

P. Hansen, B. Jaumard, M. Ruiz, and J. Xiong. Global minimization of indefinite quadratic functions subject to box constraints. *Naval Research Logistics*, 40:373–392, 1993.

H. Harman and W. Jones. Factor analysis by minimizing residuals (MINRES). *Psychometrika*, 31:351–368, 1966.

R.A. Horn and C.R. Johnson. *Matrix analysis.* Cambridge University Press, 2nd edition, 2013.

F. Husson, J. Josse, S. Le, and J. Mazet. *FactoMineR: Multivariate Exploratory Data Analysis and Data Mining*, 2015. URL `http://CRAN.R-project.org/package=FactoMineR`. R package version 1.31.3.

E. John and E.A. Yildirim. Implementation of warm-start strategies in interior-point methods for linear programming in fixed dimension. *Computational Optimization and Applications*, 41:151–183, 2008.

K. Jöreskog. Some contributions to maximum likelihood factor analysis. *Psychometrika*, 32 (4):443–482, 1967.

K. Jöreskog. Structural analysis of covariance and correlation matrices. *Psychometrika*, 43 (4):443–477, 1978.

J.B. Lasserre. *Moments, positive polynomials and their applications.* Imperial College Press, 2009.

D. Lawley and A. Maxwell. Factor analysis as a statistical method. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 12(3):209–229, 1962.

P. le Bodic and G.L. Nemhauser. An abstract model for branching and its application to mixed integer programming. *arXiv preprint arXiv:1511.01818*, 2015.

W. Ledermann. On the rank of the reduced correlational matrix in multiple-factor analysis. *Psychometrika*, 2(2):85–93, 1937.

A. Lewis. Derivatives of spectral functions. *Mathematics of Operations Research*, 21(3): 576–588, 1996.

J. Löfberg. YALMIP: A toolbox for modeling and optimization in Matlab. In *International Symposium on Computer Aided Control Systems Design*, pages 284–289. IEEE, 2004.

K. Mardia, J. Kent, and J. Bibby. *Multivariate analysis*. Academic Press, 1979.

R. Misener and C.A. Floudas. Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCQP) through piecewise-linear and edge-concave relaxations. *Mathematical Programming, Series B*, 136:155–182, 2012.

Y. Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer, 2004.

Y. Nesterov. Smooth minimization of non-smooth functions. *Mathematical Programming, Series A*, 103:127–152, 2005.

Y. Nesterov. Gradient methods for minimizing composite objective function. Technical report, Center for Operations Research and Econometrics (CORE), Catholic University of Louvain, 2007. Technical Report number 76.

B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169(3):1042–1068, 2016.

M. Overton and R. Womersley. On the sum of the largest eigenvalues of a symmetric matrix. *SIAM Journal of Matrix Analysis and Applications*, 13:41–45, 1992.

G. Pataki. On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues. *Mathematics of Operations Research*, 23:339–358, 1998.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, 2014. URL `http://www.R-project.org/`.

G. Raiche and D. Magis. nFactors: Parallel analysis and non graphical solutions to the Cattell Scree test. URL `http://cran.r-project.org/web/packages/nFactors/index.html`, 2011.

C. R. Rao. *Linear Statistical Inference and Its Applications*. Wiley, New York, 1973.

W. Revelle. *psych: Procedures for Psychological, Psychometric, and Personality Research*, 2015. URL `http://CRAN.R-project.org/package=psych`. R package version 1.5.6.

G. Riccia and A. Shapiro. Minimum rank and minimum trace of covariance matrices. *Psychometrika*, 47:443–448, 1982.

D. Robertson and J. Symons. Maximum likelihood factor analysis with rank-deficient sample covariance matrices. *Journal of Multivariate Analysis*, 98(4):813–828, 2007.

R.T. Rockafellar. *Convex Analysis*. Princeton University Press, 1996.

S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. *Neural Computation*, 11(2):305–345, 1999.

D. Rubin and D. Thayer. EM algorithms for ML factor analysis. *Psychometrika*, 47(1): 69–76, 1982.

N. V. Sahinidis. *BARON 14.3.1: Global Optimization of Mixed-Integer Nonlinear Programs*, User's Manual, 2014.

J. Saunderson, V. Chandrasekaran, P. Parrilo, and A. Willsky. Diagonal and low-rank matrix decompositions, correlation matrices, and ellipsoid fitting. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1395–1416, 2012.

A. Shapiro. Rank-reducibility of a symmetric matrix and sampling theory of minimum trace factor analysis. *Psychometrika*, 47:187–199, 1982.

A. Shapiro and J. ten Berge. Statistical inference of minimum rank factor analysis. *Psychometrika*, 67:79–94, 2002.

C. Spearman. "General Intelligence," Objectively Determined and Measured. *American Journal of Psychology*, 15:201–293, 1904.

G. Stewart and J.-G. Sun. *Matrix Perturbation Theory*. Academic Press, 1990.

M. Tawarmalani and N.V. Sahinidis. *Convexification and global optimization in continuous and mixed-integer nonlinear programming: theory, algorithms, software, and applications*, volume 65 of *Nonconvex Optimization and its Applications*. Kluwer, 2002a.

M. Tawarmalani and N.V. Sahinidis. Convex extensions and envelopes of lower semi-continuous functions. *Mathematical Programming, Series A*, 93:247–263, 2002b.

M. Tawarmalani, J-P.P. Richard, and C. Xiong. Explicit convex and concave envelopes through polyhedral subdivisions. *Mathematical Programming, Series A*, 138:531–577, 2013.

J. ten Berge. Some recent developments in factor analysis and the search for proper communalities. In *Advances in data science and classification*, pages 325–334. Springer, 1998.

J. ten Berge and H. Kiers. Computational aspects of the greatest lower bound to the reliability and constrained minimum trace factor analysis. *Psychometrika*, 46:201–213, 1981.

J. ten Berge and H. Kiers. A numerical approach to the approximate and the exact minimum rank of a covariance matrix. *Psychometrika*, 56:309–315, 1991.

J. ten Berge and H. Kiers. The minimum rank factor analysis program MRFA. URL `http://www.ppsw.rug.nl/~kiers/`, 2003.

J. ten Berge, T. Snijders, and F.E. Zegers. Computational aspects of the greatest lower bound to the reliability and constrained minimum trace factor analysis. *Psychometrika*, 46:201–213, 1981.

L.L. Thurstone. *Multiple Factor Analysis: a Development and Expansion of the Vectors of the Mind*. University of Chicago Press, 1947.

K.C. Toh, M.J. Todd, and R.H. Tutuncu. SDPT3 – a Matlab software package for semidefinite programming. *Optimization Methods and Software*, 11:545–581, 1999.

L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

E.A. Yildirim and S. Wright. Warm-start strategies in interior-point methods for linear programming. *SIAM Journal on Optimization*, 12:782–810, 2002.

Journal of Machine Learning Research 18 (2017) 1-52 Submitted 12/15; Revised 2/17; Published 4/17

Group Sparse Optimization via $\ell_{p,q}$ Regularization

Yaohua Hu MAYHHU@SZU.EDU.CN
College of Mathematics and Statistics
Shenzhen University
Shenzhen 518060, P. R. China

Chong Li CLI@ZJU.EDU.CN
School of Mathematical Sciences
Zhejiang University
Hangzhou 310027, P. R. China

Kaiwen Meng MKWFLY@126.COM
School of Economics and Management
Southwest Jiaotong University
Chengdu 610031, P. R. China

Jing Qin QINJING@CUHK.EDU.HK
School of Life Sciences
The Chinese University of Hong Kong
Shatin, New Territories, Hong Kong
and Shenzhen Research Institute
The Chinese University of Hong Kong
Shenzhen 518057, P. R. China

Xiaoqi Yang* MAYANGXQ@POLYU.EDU.HK
Department of Applied Mathematics
The Hong Kong Polytechnic University
Kowloon, Hong Kong

Editor: Mark Schmidt

Abstract

In this paper, we investigate a group sparse optimization problem via $\ell_{p,q}$ regularization in three aspects: theory, algorithm and application. In the theoretical aspect, by introducing a notion of group restricted eigenvalue condition, we establish an oracle property and a global recovery bound of order $\mathcal{O}(\lambda^{\frac{2}{2-q}})$ for any point in a level set of the $\ell_{p,q}$ regularization problem, and by virtue of modern variational analysis techniques, we also provide a local analysis of recovery bound of order $\mathcal{O}(\lambda^2)$ for a path of local minima. In the algorithmic aspect, we apply the well-known proximal gradient method to solve the $\ell_{p,q}$ regularization problems, either by analytically solving some specific $\ell_{p,q}$ regularization subproblems, or by using the Newton method to solve general $\ell_{p,q}$ regularization subproblems. In particular, we establish a local linear convergence rate of the proximal gradient method for solving the $\ell_{1,q}$ regularization problem under some mild conditions and by first proving a second-order growth condition. As a consequence, the local linear convergence rate of proximal gradient method for solving the usual ℓ_q regularization problem $(0 < q < 1)$ is obtained. Finally in the aspect of application, we present some numerical results on both the simulated data and the real data in gene transcriptional regulation.

Keywords: group sparse optimization, lower-order regularization, nonconvex optimization, restricted eigenvalue condition, proximal gradient method, iterative thresholding algorithm, gene regulation network

*. Corresponding author.

©2017 Yaohua Hu, Chong Li, Kaiwen Meng, Jing Qin, and Xiaoqi Yang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-651.html.

1. Introduction

In recent years, a great amount of attention has been paid to sparse optimization, which is to find the sparse solutions of an underdetermined linear system. The sparse optimization problem arises in a wide range of fields, such as compressive sensing, machine learning, pattern analysis and graphical modeling; see Blumensath and Davies (2008); Candès et al. (2006b); Chen et al. (2001); Donoho (2006a); Fan and Li (2001); Tibshirani (1994) and references therein.

In many applications, the underlying data usually can be represented approximately by a linear system of the form

$$Ax = b + \varepsilon,$$

where $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$ are known, $\varepsilon \in \mathbb{R}^m$ is an unknown noise vector, and $x = (x_1, x_2, \ldots, x_n)^\top \in \mathbb{R}^n$ is the variable to be estimated. If $m \ll n$, the above linear system is seriously ill-conditioned and may have infinitely many solutions. The sparse optimization problem is to recover x from information b such that x is of a sparse structure. The sparsity of variable x has been measured by the ℓ_p norm $\|x\|_p$ ($p = 0$, see Blumensath and Davies (2008); $p = 1$, see Beck and Teboulle (2009); Chen et al. (2001); Daubechies et al. (2004); Donoho (2006a); Tibshirani (1994); Wright et al. (2009); Yang and Zhang (2011); and $p = 1/2$, see Chartrand and Staneva (2008); Xu et al. (2012)). The ℓ_p norm $\|x\|_p$ for $p > 0$ is defined by

$$\|x\|_p := \left(\sum_{i=1}^n |x_i|^p\right)^{1/p},$$

while the ℓ_0 norm $\|x\|_0$ is defined by the number of nonzero components of x. The sparse optimization problem can be modeled as

$$\begin{array}{ll} \min & \|Ax - b\|_2 \\ \text{s.t.} & \|x\|_0 \le s, \end{array}$$

where s is the given sparsity level.

For the sparse optimization problem, a popular and practical technique is the regularization method, which is to transform the sparse optimization problem into an unconstrained optimization problem, called the regularization problem. For example, the ℓ_0 regularization problem is

$$\min_{x\in\mathbb{R}^n} \|Ax - b\|_2^2 + \lambda\|x\|_0,$$

where $\lambda > 0$ is the regularization parameter, providing a tradeoff between accuracy and sparsity. However, the ℓ_0 regularization problem is nonconvex and non-Lipschitz, and thus it is generally intractable to solve it directly (indeed, it is NP-hard; see Natarajan, 1995).

To overcome this difficulty, two typical relaxations of the ℓ_0 regularization problem are introduced, which are the ℓ_1 regularization problem

$$\min_{x\in\mathbb{R}^n} \|Ax-b\|_2^2+\lambda\|x\|_1 \tag{1}$$

and the ℓ_q regularization problem $(0<q<1)$

$$\min_{x\in\mathbb{R}^n} \|Ax-b\|_2^2+\lambda\|x\|_q^q. \tag{2}$$

1.1 ℓ_q Regularization Problems

The ℓ_1 regularization problem (1), also called Lasso (Tibshirani, 1994) or Basis Pursuit (Chen et al., 2001), has attracted much attention and has been accepted as one of the most useful tools for sparse optimization. Since the ℓ_1 regularization problem is a convex optimization problem, many exclusive and efficient algorithms have been proposed and developed for solving problem (1); see Beck and Teboulle (2009); Combettes and Wajs (2005); Daubechies et al. (2004); Hu et al. (2016); Nesterov (2012, 2013); Xiao and Zhang (2013); Yang and Zhang (2011). However, the ℓ_1 regularization problem (1) suffers some frustrations in practical applications. It was revealed by extensive empirical studies that the solutions obtained from the ℓ_1 regularization problem are much less sparse than the true sparse solution, that it cannot recover a signal or image with the least measurements when applied to compressed sensing, and that it often leads to sub-optimal sparsity in reality; see Chartrand (2007); Xu et al. (2012); Zhang (2010).

Recently, to overcome these drawbacks of ℓ_1 regularization, the lower-order regularization technique (that is, the ℓ_q regularization with $0<q<1$) is proposed to improve the performance of sparsity recovery of the ℓ_1 regularization problem. Chartrand and Staneva (2008) claimed that a weaker restricted isometry property is sufficient to guarantee perfect recovery in the ℓ_q regularization, and that it can recover sparse signals from fewer linear measurements than that required by the ℓ_1 regularization. Xu et al. (2012) showed that the $\ell_{1/2}$ regularization admits a significantly stronger sparsity promoting capability than the ℓ_1 regularization in the sense that it allows to obtain a more sparse solution or predict a sparse signal from a smaller amount of samplings. Qin et al. (2014) exhibited that the $\ell_{1/2}$ regularization achieves a more reliable solution in biological sense than the ℓ_1 regularization when applied to infer gene regulatory network from gene expression data of mouse embryonic stem cell. However, the ℓ_q regularization problem is nonconvex, nonsmooth and non-Lipschitz, and thus it is difficult in general to design efficient algorithms for solving it. It was presented in Ge et al. (2011) that finding the global minimal value of the ℓ_q regularization problem (2) is strongly NP-hard; while fortunately, computing a local minimum could be done in polynomial time. Some effective and efficient algorithms have been proposed to find a local minimum of problem (2), such as interior-point potential reduction algorithm (Ge et al., 2011), smoothing methods (Chen, 2012; Chen et al., 2010), splitting methods (Li and Pong, 2015a,b) and iterative reweighted minimization methods (Lai and Wang, 2011; Lai et al., 2013; Lu, 2014).

The ℓ_q regularization problem (2) is a variant of lower-order penalty problems, investigated in Huang and Yang (2003); Luo et al. (1996); Yang and Huang (2001), for a constrained optimization problem. The main advantage of the lower-order penalty functions

over the classical ℓ_1 penalty function is that they require weaker conditions to guarantee an exact penalization property and that their least exact penalty parameter is smaller; see Huang and Yang (2003). It was reported in Yang and Huang (2001) that the first- and second-order necessary optimality conditions of lower-order penalty problems converge to that of the original constrained optimization problem under a linearly independent constraint qualification.

Besides the preceding numerical algorithms, one of the most widely studied methods for solving the sparse optimization problem is the class of the iterative thresholding algorithms, which is studied in a unified framework of proximal gradient methods; see Beck and Teboulle (2009); Blumensath and Davies (2008); Combettes and Wajs (2005); Daubechies et al. (2004); Gong et al. (2013); Nesterov (2013); Xu et al. (2012) and references therein. It is convergent fast and of very low computational complexity. Benefitting from its simple formulation and low storage requirement, it is very efficient and applicable for large-scale sparse optimization problems. In particular, the iterative hard (resp. soft, half) thresholding algorithm for the ℓ_0 (resp. ℓ_1, $\ell_{1/2}$) regularization problem was studied in Blumensath and Davies (2008) (resp. Daubechies et al., 2004; Xu et al., 2012).

1.2 Global Recovery Bound

To estimate how far is the solution of regularization problems from that of the linear system, the global recovery bound (also called the ℓ_2 consistency) of the ℓ_1 regularization problem has been investigated in the literature. More specifically, under some mild conditions on A, such as the restricted isometry property (RIP, Candès and Tao, 2005) or restricted eigenvalue condition (REC, Bickel et al., 2009), van de Geer and Bühlmann (2009) established a deterministic recovery bound for the (convex) ℓ_1 regularization problem:

$$\|x^*(\ell_1)-\bar{x}\|_2^2=\mathcal{O}(\lambda^2 s), \tag{3}$$

where $x^*(\ell_1)$ is a solution of problem (1), $\bar{x}$ is a solution of the linear system $Ax=b$, and sparsity $s:=\|\bar{x}\|_0$. In the statistics literature, Bickel et al. (2009); Bunea et al. (2007); Meinshausen and Yu (2009); Zhang (2009) provided the recovery bound in a high probability for the ℓ_1 regularization problem when the size of the variable tends to infinity, under REC/RIP or some relevant conditions. However, to the best of our knowledge, the recovery bound for the general (nonconvex) ℓ_p regularization problem is still undiscovered. We will establish such a deterministic property in section 2.

1.3 Group Sparse Optimization

In applications, a wide class of problems usually have certain special structures, and recently, enhancing the recoverability due to the special structures has become an active topic in sparse optimization. One of the most popular structures is the group sparsity structure, that is, the solution has a natural grouping of its components, and the components within each group are likely to be either all zeros or all nonzeros. In general, the grouping information can be an arbitrary partition of x, and it is usually pre-defined based on prior knowledge of specific problems. Let $x:=(x_{\mathcal{G}_1}^\top,\cdots,x_{\mathcal{G}_r}^\top)^\top$ represent the group structure of x. The group

sparsity of x with such a group structure can be measured by an $\ell_{p,q}$ norm, defined by

$$\|x\|_{p,q} := \left(\sum_{i=1}^{r} \|x_{\mathcal{G}_i}\|_p^q\right)^{1/q}.$$

Exploiting the group sparsity structure can reduce the degrees of freedom in the solution, thereby leading to better recovery performance. Benefitting from these advantages, the group sparse optimization model has been applied in birthweight prediction (Bach, 2008; Yuan and Lin, 2006), dynamic MRI (Usman et al., 2011) and gene finding (Meier et al., 2008; Yang et al., 2010) with the $\ell_{2,1}$ norm. More specifically, the $\ell_{2,1}$ regularization problem

$$\min_{x\in\mathbb{R}^n} \|Ax-b\|_2^2 + \lambda\|x\|_{2,1}$$

was introduced by Yuan and Lin (2006) to study the grouped variable selection in statistics under the name of group Lasso. The $\ell_{2,1}$ regularization, an important extension of the ℓ_1 regularization, proposes an ℓ_2 regularization for each group and ultimately yields the sparsity in a group manner. Since the $\ell_{2,1}$ regularization problem is a convex optimization problem, some effective algorithms have been proposed, such as, the spectral projected gradient method (van den Berg et al., 2008), SpaRSA (Wright et al., 2009) and the alternating direction method (Deng et al., 2011).

1.4 The Aim of This Paper

In this paper, we will investigate the group sparse optimization via $\ell_{p,q}$ regularization ($p \geq 1, 0 \leq q \leq 1$), also called the $\ell_{p,q}$ regularization problem

$$\min_{x\in\mathbb{R}^n} F(x) := \|Ax-b\|_2^2 + \lambda\|x\|_{p,q}^q. \tag{4}$$

We will investigate the oracle property and recovery bound for the $\ell_{p,q}$ regularization problem, which extends the existing results in two ways: one is the lower-order regularization, including the ℓ_q regularization problem ($q<1$); the other is the group sparse optimization, including the $\ell_{2,1}$ regularization problem (group Lasso) as a special case. To this end, we will introduce the weaker notions of REC: the lower-order REC and the group REC (GREC). We will further establish the relationships between the new notions with the classical one: the lower-order REC is weaker than the classical REC, but the reverse is not true (see Example 1); and the GREC is weaker than the REC. Under the lower-order GREC, we will provide the oracle property and the global recovery bound for the $\ell_{p,q}$ regularization problem (see Theorem 9). Furthermore, we will conduct a local analysis of recovery bound for the $\ell_{p,q}$ regularization problem by virtue of modern variational analysis techniques (Rockafellar and Wets, 1998). More precisely, we assume that any nonzero group of $\bar{x}$ is active and the columns of A corresponding to the active components of $\bar{x}$ (a solution of $Ax=b$) are linearly independent, which matches the nature of the group sparsity structure. This leads us to the application of implicit function theorem and thus guarantees the existence of a local path around $\bar{x}$ which satisfies a second-order growth condition. As such, in the local recovery bound, we will establish a uniform recovery bound $\mathcal{O}(\lambda^2 S)$ for all the $\ell_{p,q}$ regularization problems; see Theorem 2.2.

The proximal gradient method is one of the most popular and practical methods for the sparse optimization problems, either convex or nonconvex problems. We will apply the proximal gradient method to solve the $\ell_{p,q}$ regularization problem (4). The advantage of the proximal gradient method is that the proximal optimization subproblems of some specific regularization have the analytical solutions, and the resulting algorithm is thus practically attractive. In the general cases when the analytical solutions of the proximal optimization subproblems seem not available, we will employ the Newton method to solve them. Furthermore, we will investigate a local linear convergence rate of the proximal gradient method for solving the $\ell_{p,q}$ regularization problem when $p=1$ and $0<q<1$ under the assumption that any nonzero group of a local minimum is active. Problem (4) of the case $p=1$ and $0<q<1$ possesses the properties that the regularization term $\|\cdot\|_{p,q}^q$ is concave near a local minimum and that the objective function $F(\cdot)$ of (4) satisfies a second-order growth condition, which plays an important role in the establishment of the local linear convergence rate. To the best of our knowledge, this is the first attempt to study the local linear convergence rate of proximal gradient method for solving the lower-order optimization problems. As a consequence of this result, we will obtain the local linear convergence rate of proximal gradient method for solving ℓ_q regularization problem ($0<q<1$), which includes the iterative half thresholding algorithm ($q=1/2$) proposed in Xu et al. (2012) as a special case. The result on local linear convergence rate of proximal gradient method for solving the ℓ_q regularization problem is still new, as far as we know.

In the aspect of application, we will conduct some numerical experiments on both simulated data and real data in gene transcriptional regulation to demonstrate the performance of the proposed proximal gradient method. From the numerical results, it is demonstrated that the $\ell_{p,1/2}$ regularization is the best one among the $\ell_{p,q}$ regularizations for $q\in[0,1]$, and it outperforms the $\ell_{p,1}$ and $\ell_{p,0}$ regularizations on both accuracy and robustness. This observation is consistent with several previous numerical studies on the ℓ_p regularization problem; see Chartrand and Staneva (2008); Xu et al. (2012). It is also illustrated from the numerical results that the proximal gradient method ($\ell_{2,1/2}$) outperforms most solvers in group sparse learning, such as OMP (Cai and Wang, 2011), FoBa (Zhang, 2011), ℓ_1-Magic (Candès et al., 2006a), ISTA (Daubechies et al., 2004), YALL1 (Yang and Zhang, 2011) etc. The R package of the proximal gradient method for solving group sparse optimization, named GSparO in CRAN, is available at https://CRAN.R-project.org/package=GSparO

1.5 Main Contributions

This paper is to investigate the group sparse optimization under a unified framework of the $\ell_{p,q}$ regularization problem (4). In this paper, we establish the oracle property and recovery bound, design an efficient numerical method for problem (4), and apply the proposed method to solve the problem of gene transcriptional regulation. The main contributions are presented as follows.

(i) We establish the following global recovery bound for the $\ell_{p,q}$ regularization problem (4) under the (p,q)-GREC:

$$\|x^*-\bar{x}\|_2^2 \leq \begin{cases} \mathcal{O}(\lambda^{\frac{2}{2-q}} S), & 2^{K-1}q = 1, \\ \mathcal{O}(\lambda^{\frac{2}{2-q}} S^{\frac{3-q}{2-q}}), & 2^{K-1}q > 1, \end{cases} \tag{5}$$

where $\bar{x}$ is a solution of $Ax = b$, $S := \|\bar{x}\|_{p,0}$ is the group sparsity, $0 < q \le 1 \le p \le 2$, x^* is any point in the level set $\mathrm{lev}_F(\bar{x})$ of problem (4), and K is the smallest integer such that $2^{K-1}q \ge 1$.

(ii) By virtue of the variational analysis technique, for all the $\ell_{p,q}$ regularization problems, we establish a uniform local recovery bound

$$\|x^*_{p,q}(\lambda) - \bar{x}\|_2^2 \le \mathcal{O}(\lambda^2 S) \quad \text{for small } \lambda,$$

where $0 < q < 1 \le p$ and $x^*_{p,q}(\lambda)$ is a local optimal solution of problem (4) (near $\bar{x}$).

(iii) We present the analytical formulae for the proximal optimization subproblems of specific $\ell_{p,q}$ regularizations when $p = 1, 2$ and $q = 0, 1/2, 2/3, 1$. Moreover, we prove that any sequence $\{x^k\}$, generated by proximal gradient method for solving the $\ell_{1,q}$ regularization problem, linearly converges to a local minimum x^* under some mild conditions, that is, there exist $N \in \mathbb{N}$, $C > 0$ and $\eta \in (0, 1)$ such that

$$F(x^k) - F(x^*) \le C\eta^k \quad \text{and} \quad \|x^k - x^*\|_2 \le C\eta^k \quad \text{for any } k \ge N.$$

(iv) Our numerical experiments show that, measured by the biological golden standards, the accuracy of the gene regulation networks forecasting can be improved by exploiting the group structure of TF complexes. The successful application of group sparse optimization to gene transcriptional regulation will facilitate biologists to study the gene regulation of higher model organisms in a genome-wide scale.

1.6 The Organization of This Paper

This paper is organized as follows. In section 2, we introduce the notions of q-REC and GREC, and establish the oracle property and (global and local) recovery bounds for the $\ell_{p,q}$ regularization problem. In section 3, we apply the proximal gradient method to solve the group sparse optimization using different types of $\ell_{p,q}$ regularization, and investigate the local linear convergence rate of the resulting proximal gradient method. Finally, section 4 exhibits the numerical results on both simulated data and real data in gene transcriptional regulation.

2. Global and Local Recovery Bounds

This section is devoted to the study of the oracle property and (global and local) recovery bounds for the $\ell_{p,q}$ regularization problem (4). To this end, we first present some basic inequalities of ℓ_p norm and introduce the notions of RECs, as well as their relationships.

The notations adopted in this paper are described as follows. We let the lowercase letters x, y, z denote the vectors, calligraphic letters $\mathcal{I}$, $\mathcal{T}$, $\mathcal{S}$, $\mathcal{J}$, $\mathcal{N}$ denote the index sets, capital letters N, S denote the numbers of groups in the index sets. In particular, we use $\mathcal{G}_i$ to denote the index set corresponding to the i-th group and $\mathcal{G}_\mathcal{S}$ to denote the index set $\{\mathcal{G}_i : i \in \mathcal{S}\}$. For $x \in \mathbb{R}^n$ and $\mathcal{T} \subseteq \{1, \ldots, n\}$, we use $x_\mathcal{T}$ to denote the subvector of x corresponding to $\mathcal{T}$. We use $\mathrm{sign} : \mathbb{R} \to \mathbb{R}$ to denote the signum function, defined by

$$\mathrm{sign}(t) = \begin{cases} 1, & t > 0, \\ 0, & t = 0, \\ -1, & t < 0. \end{cases}$$

Throughout this paper, we assume that the group sparse optimization problem is of the group structure described as follows. Let $x := (x_{\mathcal{G}_1}^\top, \cdots, x_{\mathcal{G}_r}^\top)^\top$ represent the group structure of x, where $\{x_{\mathcal{G}_i} \in \mathbb{R}^{n_i} : i = 1, \cdots, r\}$ is the grouping of x, $\sum_{i=1}^r n_i = n$ and $n_{\max} := \max\{n_i : i \in \{1, \ldots, r\}\}$. For a group $x_{\mathcal{G}_i}$, we use $x_{\mathcal{G}_i} = 0$ (reps. $x_{\mathcal{G}_i} \ne 0$, $x_{\mathcal{G}_i} \neq_{\mathbf{a}} 0$) to denote a zero (reps. nonzero, active) group, where $x_{\mathcal{G}_i} = 0$ means that $x_j = 0$ for all $j \in \mathcal{G}_i$; $x_{\mathcal{G}_i} \ne 0$ means that $x_j \ne 0$ for some $j \in \mathcal{G}_i$; and $x_{\mathcal{G}_i} \neq_{\mathbf{a}} 0$ means that $x_j \ne 0$ for all $j \in \mathcal{G}_i$. It is trivial to see that

$$x_{\mathcal{G}_i} \neq_{\mathbf{a}} 0 \quad \Rightarrow \quad x_{\mathcal{G}_i} \ne 0.$$

For this group structure and $p > 0$, the $\ell_{p,q}$ norm of x is defined by

$$\|x\|_{p,q} = \begin{cases} (\sum_{i=1}^r \|x_{\mathcal{G}_i}\|_p^q)^{1/q}, & q > 0, \\ \sum_{i=1}^r \|x_{\mathcal{G}_i}\|_p^0, & q = 0, \end{cases} \tag{6}$$

which proposes the ℓ_p norm for each group and then processes the ℓ_q norm for the resulting vector. When $p = q$, the $\ell_{p,q}$ norm coincides with the ℓ_p norm, that is, $\|x\|_{p,p} = \|x\|_p$. Furthermore, all $\ell_{p,0}$ norms share the same formula, that is, $\|x\|_{p,0} = \|x\|_{2,0}$ for all $p > 0$. In particular, when the grouping structure is degenerated to the individual feature level, that is, if $n_{\max} = 1$ or $n = r$, we have $\|x\|_{p,q} = \|x\|_q$ for all $p > 0$ and $q > 0$.

Moreover, we assume that A and b in (4) are related by a linear model (noiseless)

$$b = A\bar{x}.$$

Let $\mathcal{S} := \{i \in \{1, \ldots, r\} : \bar{x}_{\mathcal{G}_i} \ne 0\}$ be the index set of nonzero groups of $\bar{x}$, $\mathcal{S}^c := \{1, \ldots, r\} \setminus \mathcal{S}$ be the complement of $\mathcal{S}$, $S := |\mathcal{S}|$ be the group sparsity of $\bar{x}$, and $n_\mathbf{a} := \sum_{i \in \mathcal{S}} n_i$.

2.1 Inequalities of $\ell_{p,q}$ Norm

We begin with some basic inequalities of the ℓ_p and $\ell_{p,q}$ norms, which will be useful in the later discussion of RECs and recovery bounds. First, we recall the following well-known inequality

$$\left(\sum_{i=1}^n |x_i|^{\gamma_2}\right)^{1/\gamma_2} \le \left(\sum_{i=1}^n |x_i|^{\gamma_1}\right)^{1/\gamma_1} \quad \text{if } 0 < \gamma_1 \le \gamma_2, \tag{7}$$

or equivalently ($x = (x_1, x_2, \ldots, x_n)^\top$),

$$\|x\|_{\gamma_2} \le \|x\|_{\gamma_1} \quad \text{if } 0 < \gamma_1 \le \gamma_2.$$

The following lemma improves Huang and Yang (2003, lem. 4.1) and extends to the $\ell_{p,q}$ norm. It will be useful in providing a shaper global recovery bound (see Theorem 9 later).

Lemma 1 *Let $0 < q \le p \le 2$, $x \in \mathbb{R}^n$ and K be the smallest integer such that $2^{K-1}q \ge 1$. Then the following relations hold.*

(i) $\|x\|_q^q \le n^{1-2^{-K}} \|x\|_2^q$.

(ii) $\|x\|_{p,q}^q \le r^{1-2^{-K}} \|x\|_{p,2}^q$.

Proof (i) Repeatedly using the property that $\|x\|_1 \le \sqrt{n}\|x\|_2$, one has that

$$\|x\|_q^q \le \sqrt{n}\left(\sum_{i=1}^n |x_i|^{2q}\right)^{1/2} \le \cdots \le n^{\frac{1}{2}+\cdots+\frac{1}{2^K}}\left(\sum_{i=1}^n |x_i|^{2^K q}\right)^{2^{-K}}.$$

Since $2^{K-1}q \ge 1$, by (7), we obtain that

$$\left(\sum_{i=1}^n |x_i|^{2^K q}\right)^{2^{-K}} = \left(\sum_{i=1}^n (|x_i|^2)^{2^{K-1}q}\right)^{\frac{1}{2^{K-1}q}\frac{q}{2}} \le \left(\sum_{i=1}^n |x_i|^2\right)^{q/2} = \|x\|_2^q.$$

Therefore, we arrive at the conclusion that

$$\|x\|_q^q \le n^{1-2^{-K}}\|x\|_2^q.$$

(ii) By (6), it is a direct consequence of (i). ■

For example, if $q = 1$, then $K = 1$; if $q = \frac{1}{2}$ or $\frac{2}{3}$, then $K = 2$. The following lemma describes the triangle inequality of $\|\cdot\|_{p,q}^q$.

Lemma 2 *Let $0 < q \le 1 \le p$ and $x, y \in \mathbb{R}^n$. Then*

$$\|x\|_{p,q}^q - \|y\|_{p,q}^q \le \|x - y\|_{p,q}^q.$$

Proof By the subadditivity of the ℓ_p norm and (7), it is easy to see that

$$\|x_{\mathcal{G}_i}\|_p^q - \|y_{\mathcal{G}_i}\|_p^q \le \|x_{\mathcal{G}_i} - y_{\mathcal{G}_i}\|_p^q, \quad \text{for } i = 1, \ldots, r.$$

Consequently, the conclusion directly follows from (6). ■

The following lemma will be beneficial to studying properties of the lower-order REC in Proposition 5 later.

Lemma 3 *Let $\gamma \ge 1$, and two finite sequences $\{y_i : i \in \mathcal{I}\}$ and $\{x_j : j \in \mathcal{J}\}$ satisfy that $y_i \ge x_j \ge 0$ for all $(i, j) \in \mathcal{I} \times \mathcal{J}$. If $\sum_{i\in\mathcal{I}} y_i \ge \sum_{j\in\mathcal{J}} x_j$, then $\sum_{i\in\mathcal{I}} y_i^\gamma \ge \sum_{j\in\mathcal{J}} x_j^\gamma$.*

Proof Set $\bar{y} := \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} y_i$ and $\alpha := \min_{i\in\mathcal{I}} y_i$. By Huang and Yang (2003, lem. 4.1), one has that

$$\sum_{i\in\mathcal{I}} y_i^\gamma \ge \frac{1}{|\mathcal{I}|^{\gamma-1}}\left(\sum_{i\in\mathcal{I}} y_i\right)^\gamma = |\mathcal{I}|\bar{y}^\gamma. \tag{8}$$

On the other hand, let $M \in \mathbb{N}$ and $\beta \in [0, \alpha)$ be such that $\sum_{j\in\mathcal{J}} x_j = M\alpha + \beta$. Observing that $\gamma \ge 1$ and $0 \le x_j \le \alpha$ for all $j \in \mathcal{J}$, we obtain that $x_j^\gamma \le x_j\alpha^{\gamma-1}$, and thus, $\sum_{j\in\mathcal{J}} x_j^\gamma \le M\alpha^\gamma + \alpha^{\gamma-1}\beta$. By (8), it remains to show that

$$|\mathcal{I}|\bar{y}^\gamma \ge M\alpha^\gamma + \alpha^{\gamma-1}\beta. \tag{9}$$

If $|\mathcal{I}| > M$, the relation (9) is trivial since $\bar{y} \ge \alpha > \beta$; otherwise, $|\mathcal{I}| \le M$, from the facts that $|\mathcal{I}|\bar{y} \ge M\alpha + \beta$ (that is, $\sum_{i\in\mathcal{I}} y_i \ge \sum_{j\in\mathcal{J}} x_j$) and that $\gamma \ge 1$, it follows that

$$|\mathcal{I}|\bar{y}^\gamma \ge M^{1-\gamma}(M\alpha + \beta)^\gamma \ge M^{1-\gamma}(M^\gamma\alpha^\gamma + \gamma M^{\gamma-1}\alpha^{\gamma-1}\beta) \ge M\alpha^\gamma + \alpha^{\gamma-1}\beta.$$

Therefore, we verify the relation (9), and the proof is complete. ■

2.2 Group Restricted Eigenvalue Conditions

This subsection aims at the development of the critical conditions on the matrix A to guarantee the oracle property and the global recovery bound of the $\ell_{p,q}$ regularization problem (4). In particular, we will focus on the restricted eigenvalue condition (REC), and extend it to the lower-order setting and equip it with the group structure.

In the scenario of sparse optimization, given the sparsity level s, it is always assumed that the $2s$-sparse minimal eigenvalue of $A^\top A$ is positive (see Bickel et al., 2009; Bunea et al., 2007; Meinshausen and Yu, 2009), that is,

$$\phi_{\min}(2s) := \min_{\|x\|_0\le 2s} \frac{x^\top A^\top Ax}{x^\top x} > 0, \tag{10}$$

which is the minimal eigenvalue of any $2s \times 2s$ dimensional submatrix. It is well-known that the solution at sparsity level s of the linear system $Ax = b$ is unique if the condition (10) is satisfied; otherwise, assume that there are two distinct vectors $\hat{x}$ and $\tilde{x}$ such that $A\hat{x} = A\tilde{x}$ and $\|\hat{x}\|_0 = \|\tilde{x}\|_0 = s$. Then $x := \hat{x} - \tilde{x}$ is a vector such that $Ax = 0$ and $\|x\|_0 \le 2s$, and thus $\phi_{\min}(2s) = 0$, which is contradict with (10). Therefore, if the $2s$-sparse minimal eigenvalue of $A^\top A$ is zero (that is, $\phi_{\min}(2s) = 0$), one has no hope of recovering the true sparse solution from noisy observations.

However, only condition (10) is not enough and some further condition is required to maintain the nice recovery of regularization problems; see Bickel et al. (2009); Bunea et al. (2007); Meinshausen and Yu (2009); van de Geer and Bühlmann (2009); Zhang (2009) and references therein. For example, the REC was introduced in Bickel et al. (2009) to investigate the ℓ_2 consistency of the ℓ_1 regularization problem (Lasso), where the minimum in (10) is replaced by a minimum over a restricted set of vectors measured by an ℓ_1 norm inequality and the denominator is replaced by the ℓ_2 norm of only a part of x.

We now introduce the notion of the lower-order REC. Note that the residual $\hat{x} := x^*(\ell_q) - \bar{x}$, where $x^*(\ell_q)$ is an optimal solution of the ℓ_q regularization problem and $\bar{x}$ is a sparse solution of $Ax = b$, of the ℓ_q regularization problem always satisfies

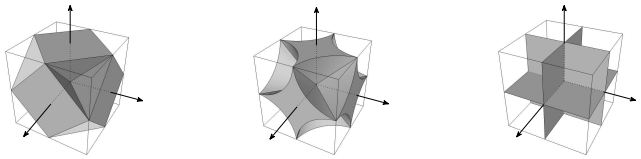
$$\|\hat{x}_{\mathcal{S}^c}\|_q \le \|\hat{x}_{\mathcal{S}}\|_q, \tag{11}$$

where $\mathcal{S}$ is the support of $\bar{x}$. Thus we introduce a lower-order REC, where the minimum is taken over a restricted set measured by an ℓ_q norm inequality such as (11), for establishing the global recovery bound of the ℓ_q regularization problem. Given $s \le t \ll n$, $x \in \mathbb{R}^n$ and $\mathcal{I} \subseteq \{1, \ldots, n\}$, we denote by $\mathcal{I}(x; t)$ the subset of $\{1, \ldots, n\}$ corresponding to the first t largest coordinates in absolute value of x in $\mathcal{I}^c$.

Definition 4 *Let $0 \le q \le 1$. The q-restricted eigenvalue condition relative to (s, t) (q-REC(s, t)) is said to be satisfied if*

$$\phi_q(s, t) := \min\left\{\frac{\|Ax\|_2}{\|x_{\mathcal{T}}\|_2} : |\mathcal{I}| \le s, \|x_{\mathcal{I}^c}\|_q \le \|x_{\mathcal{I}}\|_q, \mathcal{T} = \mathcal{I}(x; t) \cup \mathcal{I}\right\} > 0.$$

The q-REC describes a kind of restricted positive definiteness of $A^\top A$, which is valid only for the vectors satisfying the relation measured by an ℓ_q norm. The q-REC presents a unified framework of the REC-type conditions whenever $q \in [0, 1]$. In particular, we note by



(a) REC (b) 1/2-REC (c) 0-REC

Figure 1: The geometric interpretation of the RECs: the gray regions show the feasible sets $C_q(s)$ $(q = 1, 1/2, 0)$. The q-REC holds if and only if the null space of A does not intersect the gray region.

definition that 1-REC reduces to the classical REC (Bickel et al., 2009), and that $\phi_{\min}(2s) = \phi_0^2(s,s)$, and thus

$$(10) \quad \Leftrightarrow \quad 0\text{-REC}(s,s) \text{ is satisfied.}$$

It is well-known in the literature that the 1-REC is a stronger condition than the 0-REC (10). A natural question arises what are the relationships between the general q-RECs. To answer this question, associated with the q-REC, we consider the feasible set

$$C_q(s) := \{x \in \mathbb{R}^n : \|x_{\mathcal{I}^c}\|_q \le \|x_{\mathcal{I}}\|_q \text{ for some } |\mathcal{I}| \le s\},$$

which is a cone. Since the objective function associated with the q-REC is homogeneous, the q-REC(s,t) says that the null space of A does not cross over $C_q(s)$. Figure 1 presents the geometric interpretation of the q-RECs. It is shown in Figure 1 that $C_0(s) \subseteq C_{1/2}(s) \subseteq C_1(s)$, and thus

$$1\text{-REC} \quad \Rightarrow \quad 1/2\text{-REC} \quad \Rightarrow \quad 0\text{-REC}.$$

It is also observed from Figure 1 that the gap between the 1-REC and 1/2-REC and that between 1/2-REC and 0-REC are the matrices whose null spaces fall in the cones of $C_1(s) \setminus C_{1/2}(s)$ and $C_{1/2}(s) \setminus C_0(s)$, respectively.

We now provide a rigorous proof in the following proposition to identify the relationship between the feasible sets $C_q(s)$ and between the general q-RECs: the lower the q, the smaller the cone $C_q(s)$, and the weaker the q-REC.

Proposition 5 *Let $0 \le q_1 \le q_2 \le 1$ and $1 \le s \le t \ll n$. Then the following statements are true:*

(i) *$C_{q_1}(s) \subseteq C_{q_2}(s)$, and*

(ii) *if the q_2-REC(s,t) holds, then the q_1-REC(s,t) holds.*

Proof (i) Fix $x \in C_{q_1}(s)$. We use $\mathcal{I}_*$ to denote the index set of the first s largest coordinates in absolute value of x. Since $x \in C_{q_1}(s)$, it follows that $\|x_{\mathcal{I}_*^c}\|_{q_1} \le \|x_{\mathcal{I}_*}\|_{q_1}$ ($|\mathcal{I}_*| \le s$ due to the construction of $\mathcal{I}_*$). By Lemma 3 (taking $\gamma = q_2/q_1$), one has that

$$\|x_{\mathcal{I}_*^c}\|_{q_2} \le \|x_{\mathcal{I}_*}\|_{q_2},$$

that is, $x \in C_{q_2}(s)$. Hence it follows that $C_{q_1}(s) \subseteq C_{q_2}(s)$.
(ii) As proved by (i) that $C_{q_1}(s) \subseteq C_{q_2}(s)$, by the definition of q-REC, it follows that

$$\phi_{q_1}(s,t) \ge \phi_{q_2}(s,t) > 0.$$

The proof is complete. ■

To the best of our knowledge, this is the first work on introducing the lower-order REC and establishing the relationship of the lower-order RECs. In the following, we provide a counter example to show that the reverse of Proposition 5 is not true.

Example 1 (A matrix satisfying 1/2-REC but not REC) *Consider the matrix*

$$A := \begin{pmatrix} a & a+c & a-c \\ \tilde{a} & \tilde{a}-\tilde{c} & \tilde{a}+\tilde{c} \end{pmatrix} \in \mathbb{R}^{2\times 3},$$

where $a > c > 0$ and $\tilde{a} > \tilde{c} > 0$. This matrix A does not satisfy the REC$(1,1)$. Indeed, by letting $\mathcal{J} = \{1\}$ and $x = (2,-1,-1)^\top$, we have $Ax = 0$ and thus $\phi(1,1) = 0$.

Below, we claim that A satisfies the 1/2-REC$(1,1)$. It suffices to show that $\phi_{1/2}(1,1) > 0$. Let $x = (x_1,x_2,x_3)^\top$ satisfy the constraint associated with 1/2-REC$(1,1)$. As $s = 1$, the deduction is divided into the following three cases.

(i) *$\mathcal{J} = \{1\}$. Then*

$$|x_1| \ge \|x_{\mathcal{J}^c}\|_{1/2} = |x_2| + |x_3| + 2|x_2|^{1/2}|x_3|^{1/2}. \tag{12}$$

Without loss of generality, we assume $|x_1| \ge |x_2| \ge |x_3|$. Hence, $\mathcal{T} = \{1,2\}$ and

$$\frac{\|Ax\|_2}{\|x_{\mathcal{T}}\|_2} \ge \frac{\min\{a,\tilde{a}\}|x_1 + x_2 + x_3| + \min\{c,\tilde{c}\}|x_2 - x_3|}{|x_1| + |x_2|}. \tag{13}$$

If $|x_2| \le \frac{1}{3}|x_1|$, (13) reduces to

$$\frac{\|Ax\|_2}{\|x_{\mathcal{T}}\|_2} \ge \frac{\frac{\min\{a,\tilde{a}\}}{3}|x_1|}{\frac{4}{3}|x_1|} = \frac{\min\{a,\tilde{a}\}}{4}. \tag{14}$$

If $|x_2| \ge \frac{1}{3}|x_1|$, substituting (12) into (13), one has that

$$\frac{\|Ax\|_2}{\|x_{\mathcal{T}}\|_2} \ge \begin{cases} \frac{\min\{c,\tilde{c}\}}{8}, & |x_3| \le \frac{1}{2}|x_2|, \\ \frac{\min\{a,\tilde{a}\}}{4}, & |x_3| \ge \frac{1}{2}|x_2|. \end{cases} \tag{15}$$

(ii) *$\mathcal{J} = \{2\}$. Since $\mathcal{T} = \{2,1\}$ or $\{2,3\}$, it follows from Huang and Yang (2003, lem. 4.1) that*

$$|x_2| \ge \|x_{\mathcal{J}^c}\|_{1/2} \ge |x_1| + |x_3|. \tag{16}$$

Thus, it is easy to verify that $\|x_{\mathcal{T}}\|_2 \le 2|x_2|$ and that

$$\frac{\|Ax\|_2}{\|x_{\mathcal{T}}\|_2} \ge \frac{|ax_1 + (a+c)x_2 + (a-c)x_3|}{2|x_2|} = \frac{|a(x_1 + x_2 + \frac{a-c}{a}x_3) + cx_2|}{2|x_2|} \ge \frac{c}{2}, \tag{17}$$

where the last inequality follows from (16) and the fact that $a > c$.

(iii) $\mathcal{J}=\{3\}$. *Similar to the deduction of* (ii)*, one has that*

$$\frac{\|Ax\|_2}{\|x_\mathcal{T}\|_2}\geq\frac{|\tilde{a}x_1+(\tilde{a}-\tilde{c})x_2+(\tilde{a}+\tilde{c})x_3|}{2|x_3|}\geq\frac{\tilde{c}}{2}. \tag{18}$$

Therefore, by (14)-(15) *and* (17)-(18)*, we conclude that* $\phi_{1/2}(1,1)\geq\frac{1}{8}\min\{c,\tilde{c}\}>0$*, and thus, the matrix* A *satisfies the* $1/2$*-REC*$(1,1)$.

In order to establish the oracle property and the global recovery bound for the $\ell_{p,q}$ regularization problem, we further introduce the notion of group restricted eigenvalue condition (GREC). Given $S\leq N\ll r$, $x\in\mathbb{R}^n$ and $\mathcal{J}\subseteq\{1,\ldots,r\}$, we use $\mathrm{rank}_i(x)$ to denote the rank of $\|x_{\mathcal{G}_i}\|_p$ among $\{\|x_{\mathcal{G}_j}\|_p: j\in\mathcal{J}^c\}$ (in a decreasing order), $\mathcal{J}(x;N)$ to denote the index set of the first N largest groups in the value of $\|x_{\mathcal{G}_i}\|_p$ among $\{\|x_{\mathcal{G}_j}\|_p: j\in\mathcal{J}^c\}$, that is,

$$\mathcal{J}(x;N):=\{i\in\mathcal{J}^c:\mathrm{rank}_i(x)\in\{1,\ldots,N\}\}.$$

Furthermore, by letting $R:=\lceil\frac{r-|\mathcal{J}|}{N}\rceil$, we denote

$$\mathcal{J}_k(x;N):=\begin{cases}\{i\in\mathcal{J}^c:\mathrm{rank}_i(x)\in\{kN+1,\ldots,(k+1)N\}\}, & k=1,\ldots,R-1,\\ \{i\in\mathcal{J}^c:\mathrm{rank}_i(x)\in\{RN+1,\ldots,r-|\mathcal{J}|\}\}, & k=R.\end{cases} \tag{19}$$

Note that the residual $\hat{x}:=x^*(\ell_{p,q})-\bar{x}$ of the $\ell_{p,q}$ regularization problem always satisfies $\|\hat{x}_{\mathcal{G}_{\mathcal{S}^c}}\|_{p,q}\leq\|\hat{x}_{\mathcal{G}_\mathcal{S}}\|_{p,q}$. Thus we introduce the notion of GREC, where the minimum is taken over a restricted set measured by an $\ell_{p,q}$ norm inequality, as follows.

Definition 6 *Let* $0<q\leq p\leq 2$*. The* (p,q)*-group restricted eigenvalue condition relative to* (S,N) *(*(p,q)*-GREC*(S,N)*) is said to be satisfied if*

$$\phi_{p,q}(S,N):=\min\left\{\frac{\|Ax\|_2}{\|x_{\mathcal{G}_\mathcal{N}}\|_{p,2}}:|\mathcal{J}|\leq S,\|x_{\mathcal{G}_{\mathcal{J}^c}}\|_{p,q}\leq\|x_{\mathcal{G}_\mathcal{J}}\|_{p,q},\mathcal{N}=\mathcal{J}(x;N)\cup\mathcal{J}\right\}>0.$$

The (p,q)-GREC extends the q-REC to the setting equipping with a pre-defined group structure. Handling the components in each group as one element, the (p,q)-GREC admits the fewer degree of freedom, which is S (about $s/n_{\max}$), on its associated constraint than that of the q-REC, and thus it characterizes a weaker condition than the q-REC. For example, the 0-REC(s,s) is to indicate the restricted positive definiteness of $A^\top A$, which is valid only for the vectors whose cardinality is less than $2s$; while the $(p,0)$-GREC(S,S) is to describe the restricted positive definiteness of $A^\top A$ on any $2S$-group support, whose degree of freedom is much less than the $2s$-support. Thus the $(p,0)$-GREC(S,S) provides a broader condition than the 0-REC(s,s). Similar to the proof of Proposition 5, we can show that if $0\leq q_1\leq q_2\leq 1\leq p\leq 2$ and the (p,q_2)-GREC(S,N) holds, then the (p,q_1)-GREC(S,N) also holds.

We end this subsection by providing the following lemma, which will be useful in establishing the global recovery bound for the $\ell_{p,q}$ regularization problem in Theorem 9.

Lemma 7 *Let* $0<q\leq 1\leq p$*,* $\tau\geq 1$ *and* $x\in\mathbb{R}^n$*,* $\mathcal{N}:=\mathcal{J}(x;N)\cup\mathcal{J}$ *and* $\mathcal{J}_k:=\mathcal{J}_k(x;N)$ *for* $k=1,\ldots,R$*. Then the following inequalities hold*

$$\|x_{\mathcal{G}_{\mathcal{N}^c}}\|_{p,\tau}\leq\sum_{k=1}^R\|x_{\mathcal{G}_{\mathcal{J}_k}}\|_{p,\tau}\leq N^{\frac{1}{\tau}-\frac{1}{q}}\|x_{\mathcal{G}_{\mathcal{J}^c}}\|_{p,q}.$$

Proof By the definition of $\mathcal{J}_k$ (19), for each $j\in\mathcal{J}_k$, one has that

$$\|x_{\mathcal{G}_j}\|_p\leq\|x_{\mathcal{G}_i}\|_p,\quad\text{for each } i\in\mathcal{J}_{k-1},$$

and thus

$$\|x_{\mathcal{G}_j}\|_p^q\leq\frac{1}{N}\sum_{i\in\mathcal{J}_{k-1}}\|x_{\mathcal{G}_i}\|_p^q=\frac{1}{N}\|x_{\mathcal{G}_{\mathcal{J}_{k-1}}}\|_{p,q}^q.$$

Consequently, we obtain that

$$\|x_{\mathcal{G}_{\mathcal{J}_k}}\|_{p,\tau}^\tau=\sum_{i\in\mathcal{J}_k}\|x_{\mathcal{G}_i}\|_p^\tau\leq N^{1-\tau/q}\|x_{\mathcal{G}_{\mathcal{J}_{k-1}}}\|_{p,q}^\tau.$$

Further by Huang and Yang (2003, lem. 4.1) ($\tau\geq 1$ and $q\leq 1$), it follows that

$$\begin{aligned}\|x_{\mathcal{G}_{\mathcal{N}^c}}\|_{p,\tau}&=\left(\textstyle\sum_{k=1}^R\sum_{i\in\mathcal{J}_k}\|x_{\mathcal{G}_i}\|_p^\tau\right)^{1/\tau}\\&\leq\textstyle\sum_{k=1}^R\|x_{\mathcal{G}_{\mathcal{J}_k}}\|_{p,\tau}\\&\leq N^{\frac{1}{\tau}-\frac{1}{q}}\textstyle\sum_{k=1}^R\|x_{\mathcal{G}_{\mathcal{J}_{k-1}}}\|_{p,q}\\&\leq N^{\frac{1}{\tau}-\frac{1}{q}}\|x_{\mathcal{G}_{\mathcal{J}^c}}\|_{p,q}.\end{aligned}$$

The proof is complete. ■

2.3 Global Recovery Bound

In recent years, many articles have been devoted to establishing the oracle property and the global recovery bound for the ℓ_1 regularization problem (1) under the RIP or REC; see Bickel et al. (2009); Meinshausen and Yu (2009); van de Geer and Bühlmann (2009); Zhang (2009). However, to the best of our knowledge, there is few paper devoted to investigating these properties for the lower-order regularization problem.

In the preceding subsections, we have introduced the general notion of (p,q)-GREC. Under the (p,q)-GREC(S,S), the solution of $Ax=b$ with group sparsity being S is unique. In this subsection, we will present the oracle property and the global recovery bound for the $\ell_{p,q}$ regularization problem (4) under the (p,q)-GREC. The oracle property provides an upper bound on the squares error of the linear system and the violation of the true nonzero groups for each point in the level set of the objective function of problem (4)

$$\mathrm{lev}_F(\bar{x}):=\{x\in\mathbb{R}^n:\|Ax-b\|_2^2+\lambda\|x\|_{p,q}^q\leq\lambda\|\bar{x}\|_{p,q}^q\}.$$

Proposition 8 *Let* $0<q\leq 1\leq p$*,* $S>0$ *and let the* (p,q)*-GREC*(S,S) *hold. Let* $\bar{x}$ *be the unique solution of* $Ax=b$ *at a group sparsity level* S*, and* $\mathcal{S}$ *be the index set of nonzero groups of* $\bar{x}$*. Let* K *be the smallest integer such that* $2^{K-1}q\geq 1$*. Then, for any* $x^*\in\mathrm{lev}_F(\bar{x})$*, the following oracle inequality holds*

$$\|Ax^*-A\bar{x}\|_2^2+\lambda\|x^*_{\mathcal{G}_{\mathcal{S}^c}}\|_{p,q}^q\leq\lambda^{\frac{2}{2-q}}S^{\left(1-2^{-K}\right)\frac{2}{2-q}}/\phi_{p,q}^{\frac{2q}{2-q}}(S,S). \tag{20}$$

Moreover, letting $\mathcal{N}_*:=\mathcal{S}\cup\mathcal{S}(x^*;S)$*, we have*

$$\|x^*_{\mathcal{G}_{\mathcal{N}_*}}-\bar{x}_{\mathcal{G}_{\mathcal{N}_*}}\|_{p,2}^2\leq\lambda^{\frac{2}{2-q}}S^{\left(1-2^{-K}\right)\frac{2}{2-q}}/\phi_{p,q}^{\frac{4}{2-q}}(S,S).$$

Proof Let $x^* \in \mathrm{lev}_F(\bar{x})$. That is, $\|Ax^* - b\|_2^2 + \lambda\|x^*\|_{p,q}^q \le \lambda\|\bar{x}\|_{p,q}^q$. By Lemmas 1(ii) and 2, one has that

$$\begin{aligned} \|Ax^* - A\bar{x}\|_2^2 + \lambda\|x^*_{\mathcal{G}_{\mathcal{S}^c}}\|_{p,q}^q &\le \lambda\|\bar{x}_{\mathcal{G}_{\mathcal{S}}}\|_{p,q}^q - \lambda\|x^*_{\mathcal{G}_{\mathcal{S}}}\|_{p,q}^q \\ &\le \lambda\|\bar{x}_{\mathcal{G}_{\mathcal{S}}} - x^*_{\mathcal{G}_{\mathcal{S}}}\|_{p,q}^q \\ &\le \lambda S^{1-2^{-K}}\|\bar{x}_{\mathcal{G}_{\mathcal{S}}} - x^*_{\mathcal{G}_{\mathcal{S}}}\|_{p,2}^q. \end{aligned} \tag{21}$$

Noting that

$$\|x^*_{\mathcal{G}_{\mathcal{S}^c}} - \bar{x}_{\mathcal{G}_{\mathcal{S}^c}}\|_{p,q}^q - \|x^*_{\mathcal{G}_{\mathcal{S}}} - \bar{x}_{\mathcal{G}_{\mathcal{S}}}\|_{p,q}^q \le \|x^*_{\mathcal{G}_{\mathcal{S}^c}}\|_{p,q}^q - (\|\bar{x}_{\mathcal{G}_{\mathcal{S}}}\|_{p,q}^q - \|x^*_{\mathcal{G}_{\mathcal{S}}}\|_{p,q}^q) = \|x^*\|_{p,q}^q - \|\bar{x}\|_{p,q}^q \le 0.$$

Then the (p,q)-GREC(S,S) implies that

$$\|\bar{x}_{\mathcal{G}_{\mathcal{S}}} - x^*_{\mathcal{G}_{\mathcal{S}}}\|_{p,2} \le \|Ax^* - A\bar{x}\|_2/\phi_{p,q}(S,S).$$

This, together with (21), yields that

$$\|Ax^* - A\bar{x}\|_2^2 + \lambda\|x^*_{\mathcal{G}_{\mathcal{S}^c}}\|_{p,q}^q \le \lambda S^{1-2^{-K}}\|Ax^* - A\bar{x}\|_2^q/\phi_{p,q}^q(S,S), \tag{22}$$

and consequently,

$$\|Ax^* - A\bar{x}\|_2 \le \lambda^{\frac{1}{2-q}} S^{\left(1-2^{-K}\right)/(2-q)}/\phi_{p,q}^{\frac{q}{2-q}}(S,S). \tag{23}$$

Therefore, by (22) and (23), we arrive at the oracle inequality (20). Furthermore, by the definition of $\mathcal{N}_*$, the (p,q)-GREC(S,S) implies that

$$\|x^*_{\mathcal{G}_{\mathcal{N}_*}} - \bar{x}_{\mathcal{G}_{\mathcal{N}_*}}\|_{p,2}^2 \le \|Ax^* - A\bar{x}\|_2^2/\phi_{p,q}^2(S,S) \le \lambda^{\frac{2}{2-q}} S^{\left(1-2^{-K}\right)\frac{2}{2-q}}/\phi_{p,q}^{\frac{4}{2-q}}(S,S).$$

The proof is complete. ■

One of the main results of this section is presented as follows, where we establish the global recovery bound for the $\ell_{p,q}$ regularization problem under the (p,q)-GREC. We will apply oracle inequality (20) and Lemma 7 in our proof.

Theorem 9 *Let $0 < q \le 1 \le p \le 2$, $S > 0$ and let the (p,q)-GREC(S,S) hold. Let $\bar{x}$ be the unique solution of $Ax = b$ at a group sparsity level S, and $\mathcal{S}$ be the index set of nonzero groups of $\bar{x}$. Let K be the smallest integer such that $2^{K-1}q \ge 1$. Then, for any $x^* \in \mathrm{lev}_F(\bar{x})$, the following global recovery bound for problem* (4) *holds*

$$\|x^* - \bar{x}\|_2^2 \le 2\lambda^{\frac{2}{2-q}} S^{\frac{q-2}{q} + \left(1-2^{-K}\right)\frac{4}{q(2-q)}}/\phi_{p,q}^{\frac{4}{2-q}}(S,S). \tag{24}$$

More precisely,

$$\|x^* - \bar{x}\|_2^2 \le \begin{cases} \mathcal{O}(\lambda^{\frac{2}{2-q}} S), & 2^{K-1}q = 1, \\ \mathcal{O}(\lambda^{\frac{2}{2-q}} S^{\frac{3-q}{2-q}}), & 2^{K-1}q > 1. \end{cases} \tag{25}$$

Proof Let $\mathcal{N}_* := \mathcal{S} \cup \mathcal{S}(x^*; S)$ as in Proposition 8. Since $p \le 2$, it follows from Lemma 7 and Proposition 8 that

$$\|x^*_{\mathcal{G}_{\mathcal{N}_*^c}}\|_2^2 \le \|x^*_{\mathcal{G}_{\mathcal{N}_*^c}}\|_{p,2}^2 \le S^{1-2/q}\|x^*_{\mathcal{G}_{\mathcal{S}^c}}\|_{p,q}^2 \le \lambda^{\frac{2}{2-q}} S^{\frac{q-2}{q} + \left(1-2^{-K}\right)\frac{4}{q(2-q)}}/\phi_{p,q}^{\frac{4}{2-q}}(S,S).$$

Then by Proposition 8, one has that

$$\begin{aligned} \|x^* - \bar{x}\|_2^2 &= \|x^*_{\mathcal{G}_{\mathcal{N}_*}} - \bar{x}_{\mathcal{G}_{\mathcal{N}_*}}\|_2^2 + \|x^*_{\mathcal{G}_{\mathcal{N}_*^c}}\|_2^2 \\ &\le \lambda^{\frac{2}{2-q}} S^{\left(1-2^{-K}\right)\frac{2}{2-q}}/\phi_{p,q}^{\frac{4}{2-q}}(S,S) + \lambda^{\frac{2}{2-q}} S^{\frac{q-2}{q} + \left(1-2^{-K}\right)\frac{4}{q(2-q)}}/\phi_{p,q}^{\frac{4}{2-q}}(S,S) \\ &\le 2\lambda^{\frac{2}{2-q}} S^{\frac{q-2}{q} + \left(1-2^{-K}\right)\frac{4}{q(2-q)}}/\phi_{p,q}^{\frac{4}{2-q}}(S,S), \end{aligned}$$

where the last inequality follows from the fact that $2^{K-1}q \ge 1$. This proves (24). In particular, if $2^{K-1}q = 1$, then $\frac{q-2}{q} + \left(1-2^{-K}\right)\frac{4}{q(2-q)} = 1$ and thus

$$\|x^* - \bar{x}\|_2^2 \le \mathcal{O}(\lambda^{\frac{2}{2-q}} S).$$

If $2^{K-1}q > 1$, then $2^{K-2}q < 1$. Hence, $\frac{q-2}{q} + \left(1-2^{-K}\right)\frac{4}{q(2-q)} < \frac{3-q}{2-q}$, and consequently

$$\|x^* - \bar{x}\|_2^2 \le \mathcal{O}(\lambda^{\frac{2}{2-q}} S^{\frac{3-q}{2-q}}).$$

Hence (25) is obtained. The proof is complete. ■

The global recovery bound (25) is deduced under general (p,q)-GREC(S,S), which is weaker than the REC as used in van de Geer and Bühlmann (2009). It shows that the sparse solution $\bar{x}$ may be recovered by any point x^* in the level set $\mathrm{lev}_F(\bar{x})$, in particular, when x^* is a global optimal solution of problem (4), as long as λ is sufficiently small. It is well-known that when $p = 2$ and $q = 1$, convex regularization problem (4) is numerically intractable for finding the sparse solution and that when $q < 1$ finding a point in the nonconvex level set $\mathrm{lev}_F(\bar{x})$ is equivalent to finding a global minimum of minimizing the indicator function of the nonconvex level set, which is NP-hard. Thus Theorem 9 is only a theoretical result and does not provide any insight for the numerical computation of a sparse solution by virtue of problem (4). We will design a proximal gradient method in section 3, test its numerical efficiency and provide some general guidance on which q is more attractive in practical applications in section 4.

To conclude this subsection, we illustrate by an example in which (24) does not hold when $q = 1$, but it does and is also tight when $q = \frac{1}{2}$. We will testify the recovery bound $\mathcal{O}(\lambda^{4/3}S)$ in (24) when $q = \frac{1}{2}$ by using a global optimization method.

Example 2 *By letting $a = \tilde{a} = 2$ and $c = \tilde{c} = 1$ in Example 1, we consider the following matrix:*

$$A := \begin{pmatrix} 2 & 3 & 1 \\ 2 & 1 & 3 \end{pmatrix}.$$

We set $b := (2,2)^\top$ and then a true solution of $Ax = b$ is $\bar{x} := (1,0,0)^\top$. Denoting $x := (x_1, x_2, x_3)^\top$, the objective function associated with the ℓ_1 regularization problem (1) *is*

$$\begin{aligned} F(x) &:= \|Ax - b\|_2^2 + \lambda\|x\|_1 \\ &= (2x_1 + 3x_2 + x_3 - 2)^2 + (2x_1 + x_2 + 3x_3 - 2)^2 + \lambda(|x_1| + |x_2| + |x_3|). \end{aligned}$$

Let $x^*(\ell_1) := (x_1^*, x_2^*, x_3^*)^\top$ *be an optimal solution of problem* (1). *Without loss of generality, we assume* $\lambda \le 1$. *The necessary condition of* $x^*(\ell_1)$ *being an optimal solution of problem* (1) *is* $0 \in \partial F(x^*(\ell_1))$, *that is,*

$$0 \in 16x_1^* + 16x_2^* + 16x_3^* - 16 + \lambda\partial|x_1^*|, \tag{26a}$$
$$0 \in 16x_1^* + 20x_2^* + 12x_3^* - 16 + \lambda\partial|x_2^*|, \tag{26b}$$
$$0 \in 16x_1^* + 12x_2^* + 20x_3^* - 16 + \lambda\partial|x_3^*|, \tag{26c}$$

where $\partial|\mu| := \begin{cases} sign(\mu), & \mu \neq 0, \\ [-1,1], & \mu = 0. \end{cases}$

We first show that $x_i^* \ge 0$ *for* $i = 1, 2, 3$ *by contradiction. Indeed, if* $x_1^* < 0$, (26a) *reduces to*

$$16x_1^* + 16x_2^* + 16x_3^* - 16 = \lambda.$$

Summing (26b) *and* (26c), *we further have*

$$\lambda = 16x_1^* + 16x_2^* + 16x_3^* - 16 \in -\frac{\lambda}{2}(\partial|x_2^*| + \partial|x_3^*|),$$

which implies that $x_2^* \le 0$ *and* $x_3^* \le 0$. *Hence, it follows that* $F(x^*) > F(0)$, *which indicates that* x^* *is not an optimal solution of problem* (1), *and thus,* $x_1^* < 0$ *is impossible. Similarly, we can show that* $x_2^* \ge 0$ *and* $x_3^* \ge 0$.

Next, we find the optimal solution $x^*(\ell_1)$ *by only considering* $x^*(\ell_1) \ge 0$. *It is easy to obtain that the solution of* (26) *and the corresponding objective value associated with problem* (1) *can be represented respectively by*

$$x_1^* = 1 - \frac{\lambda}{16} - 2x_3^*, \quad x_2^* = x_3^* \left(0 \le x_3^* \le \frac{1}{2} - \frac{\lambda}{32}\right), \quad \text{and} \quad F\left(x^*(\ell_1)\right) = \lambda - \frac{\lambda^2}{32}.$$

Hence, $x^*(\ell_1) := \left(0, \frac{1}{2} - \frac{\lambda}{32}, \frac{1}{2} - \frac{\lambda}{32}\right)^\top$ *is an optimal solution of problem* (1). *The estimated error for this* $x^*(\ell_1)$ *is*

$$\|x^*(\ell_1) - \bar{x}\|_2^2 = 1 + \frac{1}{2}\left(1 - \frac{\lambda}{16}\right)^2 > 1,$$

which does not meet the recovery bound (25) *for any* $\lambda \le 1$.

It is revealed from Example 1 that this matrix A *satisfies the* $1/2$*-REC*$(1,1)$. *Then the hypothesis of Theorem 9 is verified, and thus, Theorem 9 is applicable to establishing the recovery bound* (25) *for the* $\ell_{1/2}$ *regularization problem. Although we cannot obtain the closed-form solution of this nonconvex* $\ell_{1/2}$ *regularization problem, as it is of only 3-dimensions, we can apply a global optimization method, the filled function method (Ge, 1990), to find the global optimal solution* $x^*(\ell_{1/2})$ *and thus to testify the recovery bound* (25). *This is done by computing the* $\ell_{1/2}$ *regularization problem for many* λ *to plot the curve* $\|x^*(\ell_{1/2}) - \bar{x}\|_2^2$. *Figure 2 illustrates the variation of the estimated error* $\|x^*(\ell_{1/2}) - \bar{x}\|_2^2$ *and the bound* $2\lambda^{4/3}$ *(that is the right-hand side of* (24), *where* $S = 1$ *and* $\phi_{1/2}(1,1) \le 1$ *(see Example 1)), when varying the regularization parameter* λ *from* 10^{-8} *to* 1. *It is illustrated from Figure 2 the recovery bound* (25) *is satisfied, and it is indeed tight, for this example.*

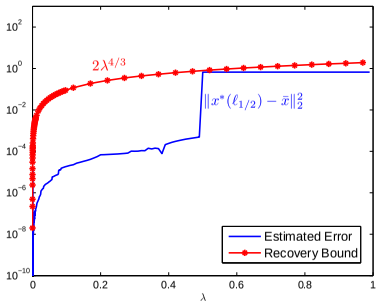


Figure 2: The illustration of the recovery bound (24) and estimated error.

2.4 Local Recovery Bound

In the preceding subsection, we provided the global analysis of the recovery bound for the $\ell_{p,q}$ regularization problem under the (p,q)-GREC; see Theorem 9. One can also observe from Figure 2 that the global recovery bound (25) is tight for the $\ell_{1/2}$ regularization problem as the curves come together at $\lambda \simeq 0.5$, but there is still a big gap for the improvement when λ is small.

This subsection is devoted to providing a local analysis of the recovery bound for the $\ell_{p,q}$ regularization problem by virtue of the variational analysis technique (Rockafellar and Wets, 1998). For $x \in \mathbb{R}^n$ and $\delta \in \mathbb{R}_+$, we use $\mathbf{B}(x,\delta)$ to denote the open ball of radius δ centered at x. For a lower semi-continuous (lsc) function $f : \mathbb{R}^n \to \mathbb{R}$ and $x, w \in \mathbb{R}^n$, the subderivative of f at x along the direction w is defined by

$$df(\bar{x})(w) := \liminf_{\tau \downarrow 0,\, w' \to w} \frac{f(\bar{x} + \tau w') - f(\bar{x})}{\tau}.$$

To begin with, we show in the following lemma a significant advantage of lower-order regularization over the ℓ_1 regularization: the lower-order regularization term can easily induce the sparsity of the local minimum.

Lemma 10 *Let* $0 < q < 1 \le p$. *Let* $f : \mathbb{R}^n \to \mathbb{R}$ *be a lsc function satisfying* $df(0)(0) = 0$. *Then the function* $F := f + \lambda\|\cdot\|_{p,q}^q$ *has a local minimum at* 0 *with the first-order growth condition being fulfilled, that is, there exist some* $\epsilon > 0$ *and* $\delta > 0$ *such that*

$$F(x) \ge F(0) + \epsilon\|x\|_2 \quad \text{for any } x \in \mathbf{B}(0,\delta).$$

Proof Let $\varphi := \lambda\|\cdot\|_{p,q}^q$ and then $F = f + \varphi$. Since φ is grouped separable, by Rockafellar and Wets (1998, prop. 10.5), it follows from the definition that $d\varphi(0) = \delta_{\{0\}}$, where δ_X is the indicator function of X. Applying Rockafellar and Wets (1998, prop. 10.9), it follows that

$$dF(0) \ge df(0) + d\varphi(0) = df(0) + \delta_{\{0\}}. \tag{27}$$

By the assumption that f is finite and $df(0)(0) = 0$, its subderivative $df(0)$ is proper (see Rockafellar and Wets, 1998, ex. 3.19). Noting that $df(0)(0) = 0$, we obtain that $df(0) + \delta_{\{0\}} = \delta_{\{0\}}$. This, together with (27), yields that $dF(0) \geq \delta_{\{0\}}$. Therefore, by definition, there exist some $\epsilon > 0$ and $\delta > 0$ such that

$$F(x) \geq F(0) + \epsilon\|x\|_2 \quad \text{for any } x \in \mathbf{B}(0, \delta).$$

The proof is complete. ■

With the help of the above lemma, we can present in the following a local version of the recovery bound. This is done by constructing a path of local minima depending on the regularization parameter λ for the regularization problem, which starts from a sparse solution of the original problem and shares the same support as this sparse solution has, resulting in a sharper bound in terms of λ^2.

Theorem 11 *Let $\bar{x}$ be a solution of $Ax = b$, S be the group sparsity of $\bar{x}$, and B be a submatrix of A consisting of its columns corresponding to the active components of $\bar{x}$. Suppose that any nonzero group of $\bar{x}$ is active, and that the columns of A corresponding to the active components of $\bar{x}$ are linearly independent. Let $0 < q < 1 \leq p$. Then there exist $\kappa > 0$ and a path of local minima of problem (4), $x^*(\lambda)$, such that*

$$\|x^*(\lambda) - \bar{x}\|_2^2 \leq \lambda^2 q^2 S \|(B^\top B)^{-1}\|^2 \max_{\bar{x}_{\mathcal{G}_i} \neq 0} \left(\|\bar{x}_{\mathcal{G}_i}\|_p^{2(q-p)} \|\bar{x}_{\mathcal{G}_i}\|_{2p-2}^{2p-2} \right) \quad \textit{for any } \lambda < \kappa.$$

Proof Without loss of generality, we let $\bar{x}$ be of structure $\bar{x} = (\bar{z}^\top, 0)^\top$ with

$$\bar{z} = (\bar{x}_{\mathcal{G}_1}^\top, \ldots, \bar{x}_{\mathcal{G}_S}^\top)^\top \text{ and } \bar{x}_{\mathcal{G}_i} \neq_{\mathbf{a}} 0 \text{ for } i = 1, \ldots, S,$$

and let s be the sparsity of $\bar{x}$. Let $A = (B, D)$ with B being the submatrix involving the first s columns of A (corresponding to the active components of $\bar{x}$). By the assumption, we have that B is of full column rank and thus $B^\top B$ is invertible. In this setting, the linear relation $A\bar{x} = b$ reduces to $B\bar{z} = b$. The proof of this theorem is divided into the following three steps:

(a) construct a smooth path from $\bar{x}$ by the implicit function theorem;

(b) validate that every point of the constructed path is a local minimum of (4); and

(c) establish the recovery bound for the constructed path.

First, to show (a), we define $H : \mathbb{R}^{s+1} \to \mathbb{R}^s$ by

$$H(z, \lambda) = 2B^\top(Bz - b) + \lambda q \begin{pmatrix} \|z_{\mathcal{G}_1}\|_p^{q-p} \sigma(z_{\mathcal{G}_1}) \\ \vdots \\ \|z_{\mathcal{G}_S}\|_p^{q-p} \sigma(z_{\mathcal{G}_S}) \end{pmatrix},$$

where $\sigma(z_{\mathcal{G}_i}) = \text{vector}\left(|z_j|^{p-1}\text{sign}(z_j)\right)_{\mathcal{G}_i}$, denoting a vector consisting of $|z_j|^{p-1}\text{sign}(z_j)$ for all $j \in \mathcal{G}_i$. Let $\bar{\delta} > 0$ be sufficiently small such that $\text{sign}(z) = \text{sign}(\bar{z})$ for any $z \in \mathbf{B}(\bar{z}, \bar{\delta})$

and thus H is smooth on $\mathbf{B}(\bar{z}, \bar{\delta}) \times \mathbb{R}$. Note that $H(\bar{z}, 0) = 0$ and $\frac{\partial H}{\partial z}(\bar{z}, 0) = 2B^\top B$. By the implicit function theorem (Rudin, 1976), there exist some $\kappa > 0$, $\delta \in (0, \bar{\delta})$ and a unique smooth function $\xi : (-\kappa, \kappa) \to \mathbf{B}(\bar{z}, \delta)$ such that

$$\{(z, \lambda) \in \mathbf{B}(\bar{z}, \bar{\delta}) \times (-\kappa, \kappa) : H(z, \lambda) = 0\} = \{(\xi(\lambda), \lambda) : \lambda \in (-\kappa, \kappa)\}, \tag{28}$$

and

$$\frac{d\xi}{d\lambda} = -q \left(2B^\top B + \lambda q \begin{pmatrix} M_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & M_{\mathcal{S}} \end{pmatrix} \right)^{-1} \begin{pmatrix} \|\xi(\lambda)_{\mathcal{G}_1}\|_p^{q-p} \sigma(\xi(\lambda)_{\mathcal{G}_1}) \\ \vdots \\ \|\xi(\lambda)_{\mathcal{G}_S}\|_p^{q-p} \sigma(\xi(\lambda)_{\mathcal{G}_S}) \end{pmatrix}, \tag{29}$$

where M_i for each $i = 1, \ldots, S$ is denoted by

$$M_i = (q-p)\|\xi(\lambda)_{\mathcal{G}_i}\|_p^{q-2p} (\sigma(\xi(\lambda)_{\mathcal{G}_i}))(\sigma(\xi(\lambda)_{\mathcal{G}_i}))^\top + (p-1)\|\xi(\lambda)_{\mathcal{G}_i}\|_p^{q-p} \text{diag}\left(|\xi(\lambda)_j|^{p-2}\right),$$

and $\text{diag}\left(|\xi(\lambda)_j|^{p-2}\right)$ denotes a diagonal matrix generated by vector $\left(|\xi(\lambda)_j|^{p-2}\right)$. Thus, by (28) and (29), we have constructed a smooth path $\xi(\lambda)$ near $\bar{z}$, $\lambda \in (-\kappa, \kappa)$, such that

$$2B^\top(B\xi(\lambda) - b) + \lambda q \begin{pmatrix} \|\xi(\lambda)_{\mathcal{G}_1}\|_p^{q-p} \sigma(\xi(\lambda)_{\mathcal{G}_1}) \\ \vdots \\ \|\xi(\lambda)_{\mathcal{G}_S}\|_p^{q-p} \sigma(\xi(\lambda)_{\mathcal{G}_S}) \end{pmatrix} = 0 \tag{30}$$

and

$$2B^\top B + \lambda q \begin{pmatrix} M_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & M_{\mathcal{S}} \end{pmatrix} \succ 0. \tag{31}$$

This shows that (a) is done as desired.

For fixed $\lambda \in (-\kappa, \kappa)$, let $x^*(\lambda) := (\xi(\lambda)^\top, 0)^\top$. To verify (b), we prove that $x^*(\lambda)$, with $\xi(\lambda)$ satisfying (30) and (31), is a local minimum of problem (4). Let $h : \mathbb{R}^s \to \mathbb{R}$ be a function with $h(z) := \|Bz - b\|_2^2 + \lambda\|z\|_{p,q}^q$ for any $z \in \mathbb{R}^s$. Note that $h(\xi(\lambda)) = \|Ax^*(\lambda) - b\|_2^2 + \lambda\|x^*(\lambda)\|_{p,q}^q$ and that h is smooth around $\xi(\lambda)$. By noting that $\xi(\lambda)$ satisfies (30) and (31) (the first- and second- derivative of h at $\xi(\lambda)$), one has that h satisfies the second-order growth condition at $\xi(\lambda)$, that is, there exist $\epsilon_\lambda > 0$ and $\delta_\lambda > 0$ such that

$$h(z) \geq h(\xi(\lambda)) + 2\epsilon_\lambda\|z - \xi(\lambda)\|_2^2 \quad \text{for any } z \in \mathbf{B}(\xi(\lambda), \delta_\lambda). \tag{32}$$

In what follows, let $\epsilon_\lambda > 0$ and $\delta_\lambda > 0$ be given as above, and select $\epsilon_0 > 0$ such that

$$\sqrt{\epsilon_\lambda \epsilon_0} - \|B\|\|D\| > 0. \tag{33}$$

According to Lemma 10 (with $\|D \cdot\|_2^2 + 2\langle B\xi(\lambda) - b, D\cdot\rangle - 2\epsilon_0\|\cdot\|_2^2$ in place of f), there exists $\delta_0 > 0$ such that

$$\|Dy\|_2^2 + 2\langle B\xi(\lambda) - b, Dy\rangle - 2\epsilon_0\|y\|_2^2 + \lambda\|y\|_{p,q}^q \geq 0 \quad \text{for any } y \in \mathbf{B}(0, \delta_0). \tag{34}$$

Thus, for each $x := (z, y) \in \mathbf{B}(\xi(\lambda), \delta_\lambda) \times \mathbf{B}(0, \delta_0)$, it follows that

$$\begin{aligned}
&\|Ax - b\|_2^2 + \lambda \|x\|_{p,q}^q \\
&= \|Bz - b + Dy\|_2^2 + \lambda \|z\|_{p,q}^q + \lambda \|y\|_{p,q}^q \\
&= \|Bz - b\|_2^2 + \lambda \|z\|_{p,q}^q + \|Dy\|_2^2 + 2\langle Bz - b, Dy\rangle + \lambda \|y\|_{p,q}^q \\
&= h(z) + \|Dy\|_2^2 + 2\langle B\xi(\lambda) - b, Dy\rangle + \lambda \|y\|_{p,q}^q + 2\langle B(z - \xi(\lambda)), Dy\rangle.
\end{aligned}$$

By (32) and (34), it yields that

$$\begin{aligned}
&\|Ax - b\|_2^2 + \lambda \|x\|_{p,q}^q \\
&\geq h(\xi(\lambda)) + 2\epsilon_\lambda \|z - \xi(\lambda)\|_2^2 + 2\epsilon_0 \|y\|_2^2 + 2\langle B(z - \xi(\lambda)), Dy\rangle \\
&\geq h(\xi(\lambda)) + 4\sqrt{\epsilon_\lambda \epsilon_0} \|z - \xi(\lambda)\|_2 \|y\|_2 - 2\|B\|\|D\|\|z - \xi(\lambda)\|_2 \|y\|_2 \\
&= \|Ax^*(\lambda) - b\|_2^2 + \lambda \|x^*(\lambda)\|_{p,q}^q + 2(2\sqrt{\epsilon_\lambda \epsilon_0} - \|B\|\|D\|) \|z - \xi(\lambda)\|_2 \|y\|_2 \\
&\geq \|Ax^*(\lambda) - b\|_2^2 + \lambda \|x^*(\lambda)\|_{p,q}^q,
\end{aligned}$$

where the last inequality follows from (33). Hence $x^*(\lambda)$ is a local minimum of problem (4), and (b) is verified.

Finally, we check (c) by providing an upper bound on the distance from $\xi(\lambda)$ to $\bar{z}$. By (30), one has that

$$\xi(\lambda) - \bar{z} = -\frac{\lambda q}{2}((B^\top B)^{-1}) \begin{pmatrix} \|\xi(\lambda)_{\mathcal{G}_1}\|_p^{q-p} \sigma(\xi(\lambda)_{\mathcal{G}_1}) \\ \vdots \\ \|\xi(\lambda)_{\mathcal{G}_S}\|_p^{q-p} \sigma(\xi(\lambda)_{\mathcal{G}_S}) \end{pmatrix}. \tag{35}$$

Noting that $\{\xi(\lambda) : \lambda \in (-\kappa, \kappa)\} \subseteq \mathbf{B}(\bar{z}, \bar{\delta})$, without loss of generality, we assume for any $\lambda < \kappa$ that

$$\|\xi(\lambda)_{\mathcal{G}_i}\|_p^{2(q-p)} \leq 2\|\bar{z}_{\mathcal{G}_i}\|_p^{2(q-p)} \quad \text{and} \quad \|\xi(\lambda)_{\mathcal{G}_i}\|_{2p-2}^{2p-2} \leq 2\|\bar{z}_{\mathcal{G}_i}\|_{2p-2}^{2p-2} \quad \text{for } i = 1, \ldots, S$$

(otherwise, we choose a smaller $\bar{\delta}$). Recall that $\sigma(\xi(\lambda)_{\mathcal{G}_i}) = \text{vector}\left(|\xi(\lambda)_j|^{p-1}\text{sign}(\xi(\lambda)_j)\right)_{\mathcal{G}_i}$. We obtain from (35) that

$$\begin{aligned}
\|\xi(\lambda) - \bar{z}\|_2^2 &\leq \tfrac{\lambda^2 q^2}{4} \|(B^\top B)^{-1}\|^2 \textstyle\sum_{i=1}^S \left(\|\xi(\lambda)_{\mathcal{G}_i}\|_p^{2(q-p)} \sum_{j \in \mathcal{G}_i} |\xi(\lambda)_j|^{2p-2} \right) \\
&= \tfrac{\lambda^2 q^2}{4} \|(B^\top B)^{-1}\|^2 \textstyle\sum_{i=1}^S \left(\|\xi(\lambda)_{\mathcal{G}_i}\|_p^{2(q-p)} \|\xi(\lambda)_{\mathcal{G}_i}\|_{2p-2}^{2p-2} \right) \\
&\leq \tfrac{\lambda^2 q^2}{4} \|(B^\top B)^{-1}\|^2 S \max_{i=1,\ldots,S} \left(\|\xi(\lambda)_{\mathcal{G}_i}\|_p^{2(q-p)} \|\xi(\lambda)_{\mathcal{G}_i}\|_{2p-2}^{2p-2} \right) \\
&\leq \lambda^2 q^2 S \|(B^\top B)^{-1}\|^2 \max_{i=1,\ldots,S} \left(\|\bar{z}_{\mathcal{G}_i}\|_p^{2(q-p)} \|\bar{z}_{\mathcal{G}_i}\|_{2p-2}^{2p-2} \right).
\end{aligned}$$

Hence we arrive at that

$$\|x^*(\lambda) - \bar{x}\|_2^2 = \|\xi(\lambda) - \bar{z}\|_2^2 \leq \lambda^2 q^2 S \|(B^\top B)^{-1}\|^2 \max_{\bar{x}_{\mathcal{G}_i} \neq 0} \left(\|\bar{x}_{\mathcal{G}_i}\|_p^{2(q-p)} \|\bar{x}_{\mathcal{G}_i}\|_{2p-2}^{2p-2} \right)$$

for any $\lambda < \kappa$, and the proof is complete. ■

Theorem 11 provides a uniform local recovery bound for all the $\ell_{p,q}$ regularization problems ($0 < q < 1 \leq p$), which is

$$\|x_{p,q}^*(\lambda) - \bar{x}\|_2^2 \leq \mathcal{O}(\lambda^2 S),$$

where $x_{p,q}^*(\lambda)$ is a local optimal solution of problem (4) (near $\bar{x}$). This bound improves the global recovery bound given in Theorem 9 (of order $\mathcal{O}(\lambda^{\frac{2}{2-q}})$) and shares the same one with the $\ell_{p,1}$ regularization problem (group Lasso); see Blumensath and Davies (2008); van de Geer and Bühlmann (2009). It is worth noting that our proof technique is not working when $q = 1$ as Lemma 10 fails in this case.

3. Proximal Gradient Method for Group Sparse Optimization

Many efficient algorithms have been proposed to solve sparse optimization problems, and one of the most popular optimization algorithms is the proximal gradient method (PGM); see Beck and Teboulle (2009); Combettes and Wajs (2005); Xiao and Zhang (2013) and references therein. It was reported in Combettes and Wajs (2005) that the PGM for solving the ℓ_1 regularization problem (1) reduces to the well-known iterative soft thresholding algorithm (ISTA), and that the ISTA has a local linear convergence rate under some assumptions; see Bredies and Lorenz (2008); Hale et al. (2008); Tao et al. (2016). Recently, the global convergence of the PGM for solving some types of nonconvex regularization problems have been studied under the framework of the Kurdyka-Łojasiewicz theory (Attouch et al., 2010; Bolte et al., 2013), the majorization-minimization scheme (Mairal, 2013), the coordinate gradient descent method (Tseng and Yun, 2009), the general iterative shrinkage and thresholding (Gong et al., 2013) and the successive upper-bound minimization approach (Razaviyayn et al., 2013).

In this section, we apply the PGM to solve the group sparse optimization problem (4) (PGM-GSO), which is stated as follows.

Algorithm 1 (PGM-GSO) *Select a stepsize v, start with an initial point $x_0 \in \mathbb{R}^n$, and generate a sequence $\{x^k\} \subseteq \mathbb{R}^n$ via the iteration*

$$z^k = x^k - 2vA^\top(Ax^k - b), \tag{36}$$

$$x^{k+1} \in \arg\min_{x \in \mathbb{R}^n} \left\{ \lambda \|x\|_{p,q}^q + \frac{1}{2v} \|x - z^k\|_2^2 \right\}. \tag{37}$$

Global convergence of the PGM-GSO falls in the framework of the Kurdyka-Łojasiewicz theory (see Attouch et al., 2010). In particular, following from Bolte et al. (2013, prop. 3), the sequence generated by the PGM-GSO converges to a critical point, especially a global minimum when $q \geq 1$ and a local minimum when $q = 0$ (inspired by the idea in Blumensath and Davies, 2008), as summarized as follows.

Theorem 12 *Let $p \geq 1$, and let $\{x^k\}$ be a sequence generated by the PGM-GSO with $v < \frac{1}{2}\|A\|_2^{-2}$. Then the following statements hold:*

(i) *if $q \geq 1$, then $\{x^k\}$ converges to a global minimum of problem (4),*

(ii) *if $q = 0$, then $\{x^k\}$ converges to a local minimum of problem* (4)*, and*

(iii) *if $0 < q < 1$, then $\{x^k\}$ converges to a critical point* [1] *of problem* (4).

Although the global convergence of the PGM-GSO has been provided in Theorem 12, some important issues of the PGM-GSO have not been discovered yet. The section is to continue the development of the PGM-GSO, concentrating on its efficiency and applicability. In particular, we will establish the local convergence rate of the PGM-GSO under some mild conditions, and derive the analytical solutions of subproblem (37) for some specific p and q.

3.1 Local Linear Convergence Rate

In this subsection, we establish the local linear convergence rate of the PGM-GSO for the case when $p = 1$ and $0 < q < 1$. For the reminder of this subsection, we always assume that $p = 1$ and $0 < q < 1$.

To begin with, by virtue of the second-order necessary condition of subproblem (37), the following lemma provides a lower bound for nonzero groups of sequence $\{x^k\}$ generated by the PGM-GSO and shows that the index set of nonzero groups of $\{x^k\}$ maintains constant for large k.

Lemma 13 *Let $K = (v\lambda q(1-q))^{\frac{1}{2-q}}$, and let $\{x^k\}$ be a sequence generated by the PGM-GSO with $v < \frac{1}{2}\|A\|_2^{-2}$. Then the following statements hold:*

(i) *For any i and k, if $x^k_{\mathcal{G}_i} \neq 0$, then $\|x^k_{\mathcal{G}_i}\|_1 \geq K$.*

(ii) *x^k shares the same index set of nonzero groups for large k, that is, there exist $N \in \mathbb{N}$ and $\mathcal{I} \subseteq \{1, \ldots, r\}$ such that*

$$\begin{cases} x^k_{\mathcal{G}_i} \neq 0, & i \in \mathcal{I}, \\ x^k_{\mathcal{G}_i} = 0, & i \notin \mathcal{I}, \end{cases} \quad \text{for all } k \geq N.$$

Proof (i) For each group $x^k_{\mathcal{G}_i}$, by (37), one has that

$$x^k_{\mathcal{G}_i} \in \underset{x \in \mathbb{R}^{n_i}}{\arg\min} \left\{ \lambda \|x\|_1^q + \frac{1}{2v}\|x - z^{k-1}_{\mathcal{G}_i}\|_2^2 \right\}. \tag{38}$$

If $x^k_{\mathcal{G}_i} \neq 0$, we define $\mathcal{A}_i^k := \{j \in \mathcal{G}_i : x_j^k \neq 0\}$ and $a_i^k := |\mathcal{A}_i^k|$. Without loss of generality, we assume that the first a_i^k components of $x^k_{\mathcal{G}_i}$ are nonzeros. Then (38) implies that

$$x^k_{\mathcal{G}_i} \in \underset{x \in \mathbb{R}^{a_i^k} \times \{0\}}{\arg\min} \left\{ \lambda \|x\|_1^q + \frac{1}{2v}\|x - z^{k-1}_{\mathcal{G}_i}\|_2^2 \right\}.$$

Its second-order necessary condition says that

$$\frac{1}{v} I_i^k + \lambda q(q-1) M_i^k \succeq 0,$$

1. A point x is said to be a critical point of F if 0 belongs to its limiting subdifferential at x; see Mordukhovich (2006).

where I_i^k is the identity matrix in $R^{a_i^k \times a_i^k}$ and $M_i^k = \|x^k_{\mathcal{A}_i^k}\|_1^{q-2}(\mathrm{sign}(x^k_{\mathcal{A}_i^k}))(\mathrm{sign}(x^k_{\mathcal{A}_i^k}))^\top$. Let e be the first column of I_i^k. Then, we obtain that

$$\frac{1}{v} e^\top I_i^k e + \lambda q(q-1) e^\top M_i^k e \geq 0,$$

that is,

$$\frac{1}{v} + \lambda q(q-1)\|x^k_{\mathcal{A}_i^k}\|_1^{q-2} \geq 0.$$

Consequently, it implies that

$$\|x^k_{\mathcal{G}_i}\|_1 = \|x^k_{\mathcal{A}_i^k}\|_1 \geq (v\lambda q(1-q))^{\frac{1}{2-q}} = K.$$

Hence, it completes the proof of (i).
(ii) Recall from Theorem 12 that $\{x^k\}$ converges to a critical point x^*. Then there exists $N \in \mathbb{N}$ such that $\|x^k - x^*\|_2 < \frac{K}{2\sqrt{n}}$, and thus,

$$\|x^{k+1} - x^k\|_2 \leq \|x^{k+1} - x^*\|_2 + \|x^k - x^*\|_2 < \frac{K}{\sqrt{n}}, \tag{39}$$

for any $k \geq N$. Proving by contradiction, without loss of generality, we assume that there exist $k \geq N$ and $i \in \{1, \ldots, r\}$ such that $x^{k+1}_{\mathcal{G}_i} \neq 0$ and $x^k_{\mathcal{G}_i} = 0$. Then it follows from (i) that

$$\|x^{k+1} - x^k\|_2 \geq \frac{1}{\sqrt{n}}\|x^{k+1} - x^k\|_1 \geq \frac{1}{\sqrt{n}}\|x^{k+1}_{\mathcal{G}_i} - x^k_{\mathcal{G}_i}\|_1 \geq \frac{K}{\sqrt{n}},$$

which yields a contradiction with (39). The proof is complete. ■

Let $x^* \in \mathbb{R}^n$, and let

$$\mathcal{S} := \left\{ i \in \{1, \ldots, r\} : x^*_{\mathcal{G}_i} \neq 0 \right\} \quad \text{and} \quad B := (A_{\cdot j})_{j \in \mathcal{G}_\mathcal{S}}.$$

Consider the following restricted problem

$$\min_{y \in \mathbb{R}^{n_\mathbf{a}}} \quad f(y) + \varphi(y), \tag{40}$$

where $n_\mathbf{a} := \sum_{i \in \mathcal{S}} n_i$, and

$$f : \mathbb{R}^{n_\mathbf{a}} \to \mathbb{R} \quad \text{with} \quad f(y) := \|By - b\|_2^2 \quad \text{for any } y \in \mathbb{R}^{n_\mathbf{a}},$$

$$\varphi : \mathbb{R}^{n_\mathbf{a}} \to \mathbb{R} \quad \text{with} \quad \varphi(y) := \lambda\|y\|_{1,q}^q \quad \text{for any } y \in \mathbb{R}^{n_\mathbf{a}}.$$

The following lemma provides the first- and second-order conditions for a local minimum of the $\ell_{1,q}$ regularization problem, and shows a second-order growth condition for the restricted problem (40), which is useful for establishing the local linear convergence rate of the PGM-GSO.

Lemma 14 *Assume that x^* is a local minimum of problem (4). Suppose that any nonzero group of x^* is active, and the columns of B are linearly independent* [2]*. Then the following statements are true:*

(i) *The following first- and second-order conditions hold*

$$2B^\top(By^*-b)+\lambda q\begin{pmatrix}\|y^*_{\mathcal{G}_1}\|_1^{q-1}\mathrm{sign}(y^*_{\mathcal{G}_1})\\ \vdots\\ \|y^*_{\mathcal{G}_S}\|_1^{q-1}\mathrm{sign}(y^*_{\mathcal{G}_S})\end{pmatrix}=0, \tag{41}$$

and

$$2B^\top B+\lambda q(q-1)\begin{pmatrix}M_1^* & 0 & 0\\ 0 & \ddots & 0\\ 0 & 0 & M_S^*\end{pmatrix}\succ 0, \tag{42}$$

where

$$M_i^*=\|y^*_{\mathcal{G}_i}\|_1^{q-2}\left(\mathrm{sign}(y^*_{\mathcal{G}_i})\right)\left(\mathrm{sign}(y^*_{\mathcal{G}_i})\right)^\top.$$

(ii) *The second-order growth condition holds at y^* for problem (40), that is, there exist $\varepsilon>0$ and $\delta>0$ such that*

$$(f+\varphi)(y)\geq(f+\varphi)(y^*)+\varepsilon\|y-y^*\|_2^2 \quad \textit{for any } y\in\mathbf{B}(y^*,\delta). \tag{43}$$

Proof Without loss of generality, we assume that $\mathcal{S}:=\{1,\ldots,S\}$. By assumption, x^* is of structure $x^*:=(y^{*\top},0)^\top$ with

$$y^*:=(x^{*\top}_{\mathcal{G}_1},\ldots,x^{*\top}_{\mathcal{G}_S})^\top \text{ and } x^*_{\mathcal{G}_i}\neq_{\mathbf{a}}0 \text{ for } i=1,\ldots,S. \tag{44}$$

(i) By (44), one has that $\varphi(\cdot)$ is smooth around y^* with its first- and second-derivatives being

$$\varphi'(y^*)=\lambda q\begin{pmatrix}\|y^*_{\mathcal{G}_1}\|_1^{q-1}\mathrm{sign}(y^*_{\mathcal{G}_1})\\ \vdots\\ \|y^*_{\mathcal{G}_S}\|_1^{q-1}\mathrm{sign}(y^*_{\mathcal{G}_S})\end{pmatrix},$$

and

$$\varphi''(y^*)=\lambda q(q-1)\begin{pmatrix}M_1^* & 0 & 0\\ 0 & \ddots & 0\\ 0 & 0 & M_S^*\end{pmatrix};$$

hence $(f+\varphi)(\cdot)$ is also smooth around y^*. Therefore, we obtain the following first- and second-order necessary conditions of problem (40)

$$f'(y^*)+\varphi'(y^*)=0 \quad\text{and}\quad f''(y^*)+\varphi''(y^*)\succeq 0,$$

2. This assumption is mild, and it holds automatically for the case when $n_{\max}=1$ (see Chen et al., 2010, thm. 2.1).

which are (41) and

$$2B^\top B+\lambda q(q-1)\begin{pmatrix}M_1^* & 0 & 0\\ 0 & \ddots & 0\\ 0 & 0 & M_S^*\end{pmatrix}\succeq 0,$$

respectively. Proving by contradiction, we assume that (42) does not hold, that is, there exists some $w\neq 0$ such that

$$2w^\top B^\top Bw+\lambda q(q-1)\sum_{i=1}^S\left(\|y^*_{\mathcal{G}_i}\|_1^{q-2}\cdot\left(\sum_{j\in\mathcal{G}_i}w_j\mathrm{sign}(y^*_j)\right)^2\right)=0. \tag{45}$$

By assumption, one has that $B^\top B\succ 0$, and thus it follows from (45) that

$$\left(\sum_{j\in\mathcal{G}_i}w_j\mathrm{sign}(y^*_j)\right)^2>0 \quad \text{for some } i\in\{1,\ldots,S\}. \tag{46}$$

Let $h:\mathbb{R}\to\mathbb{R}$ with $h(t):=\|B(y^*+tw)-b\|_2^2+\lambda\|y^*+tw\|_p^p$ for any $t\in\mathbb{R}$. Clearly, $h(\cdot)$ has a local minimum at 0, and $h(\cdot)$ is smooth around 0 with its derivatives being

$$h'(0)=2w^\top B^\top(By^*-b)+\lambda q\sum_{i=1}^S\left(\|y^*_{\mathcal{G}_i}\|_1^{q-1}\cdot\sum_{j\in\mathcal{G}_i}w_j\mathrm{sign}(y^*_j)\right)=0,$$

$$h''(0)=2w^\top B^\top Bw+\lambda q(q-1)\sum_{i=1}^S\left(\|y^*_{\mathcal{G}_i}\|_1^{q-2}\cdot\left(\sum_{j\in\mathcal{G}_i}w_j\mathrm{sign}(y^*_j)\right)^2\right)=0 \text{ (by (45))},$$

$$h^{(3)}(0)=\lambda q(q-1)(q-2)\sum_{i=1}^S\left(\|y^*_{\mathcal{G}_i}\|_1^{q-3}\cdot\left(\sum_{j\in\mathcal{G}_i}w_j\mathrm{sign}(y^*_j)\right)^3\right)=0,$$

$$h^{(4)}(0)=\lambda q(q-1)(q-2)(q-3)\sum_{i=1}^S\left(\|y^*_{\mathcal{G}_i}\|_1^{q-4}\cdot\left(\sum_{j\in\mathcal{G}_i}w_j\mathrm{sign}(y^*_j)\right)^4\right)<0; \tag{47}$$

due to (46). However, by elementary of calculus, it is clear that $h^{(4)}(0)$ must be nonnegative (since $h(\cdot)$ obtains a local minimum at 0), which yields a contradiction to (47). Therefore, we proved (42).

(ii) By the structure of y^* (44), $\varphi(\cdot)$ is smooth around y^*, and thus, $(f+\varphi)(\cdot)$ is also smooth around y^* with its derivatives being

$$f'(y^*)+\varphi'(y^*)=0 \quad\text{and}\quad f''(y^*)+\varphi''(y^*)\succ 0;$$

due to (41) and (42). Hence (43) follows from Rockafellar and Wets (1998, thm. 13.24). This completes the proof. ■

The key for the study of local convergence rate of the PGM-GSO is the descent property of the function $f+\varphi$ in each iteration step. The following lemma states some basic properties of active groups of sequence $\{x^k\}$ generated by the PGM-GSO.

Lemma 15 *Let $\{x^k\}$ be a sequence generated by the PGM-GSO with $v < \frac{1}{2}\|A\|_2^{-2}$, which converges to x^* (by Theorem 12). Assume that the assumptions in Lemma 14 are satisfied. We define*

$$\alpha := \|B\|_2^2, \quad L := 2\|A\|_2^2 \quad \text{and} \quad D_k := \varphi(y^k) - \varphi(y^{k+1}) + \langle f'(y^k), y^k - y^{k+1}\rangle.$$

Then there exist $\delta > 0$ and $N \in \mathbb{N}$ such that the following inequalities hold for any $k \geq N$:

$$D_k \geq \left(\frac{1}{v} - \alpha\right)\|y^k - y^{k+1}\|_2^2, \tag{48}$$

and

$$(f+\varphi)(y^{k+1}) \leq (f+\varphi)(y^k) - \left(1 - \frac{Lv}{2(1-v\alpha)}\right) D_k. \tag{49}$$

Proof By Lemma 13(ii) and the fact that $\{x^k\}$ converges to x^*, one has that x^k shares the same index set of nonzero groups with that of x^* for large k; further by the structure of y^* (44), we obtain that all components in nonzero groups of y^k are nonzero for large k. In another word, we have

$$\text{there exists } N \in \mathbb{N} \text{ such that } y^k \neq_{\mathbf{a}} 0 \text{ and } x^k_{\mathcal{G}_{\mathcal{S}^c}} = 0 \text{ for any } k \geq N; \tag{50}$$

hence $\varphi(\cdot)$ is smooth around y^k for any $k \geq N$.

In view of PGM-GSO and the decomposition of $x = \left(y^\top, z^\top\right)^\top$, one has that

$$y^{k+1} \in \arg\min\left\{\varphi(y) + \frac{1}{2v}\left\|y - \left(y^k - vf'(y^k)\right)\right\|_2^2\right\}.$$

Its first-order necessary condition is

$$\varphi'(y^{k+1}) = \frac{1}{v}\left(y^k - vf'(y^k) - y^{k+1}\right). \tag{51}$$

Recall from (42) that $\varphi''(y^*) \succ -2B^\top B$. Since $\varphi(\cdot)$ is smooth around y^*, then there exists $\delta > 0$ such that $\varphi''(w) \succ -2B^\top B$ for any $w \in \mathbf{B}(y^*, \delta)$. Noting that $\{y^k\}$ converges to y^*, without loss of generality, we assume that $\|y^k - y^*\| < \delta$ for any $k \geq N$ (otherwise, we can choose a larger N). Therefore, one has that $\varphi''(y^k) \succ -2B^\top B$ for any $k \geq N$. Then by Taylor expansion, we can assume without loss of generality that the following inequality holds for any $k \geq N$ and any $w \in \mathbf{B}(y^*, \delta)$ (otherwise, we can choose a smaller δ):

$$\varphi(w) > \varphi(y^{k+1}) + \langle\varphi'(y^{k+1}), w - y^{k+1}\rangle - \alpha\|w - y^{k+1}\|_2^2.$$

Hence, by (51), it follows that

$$\varphi(w) - \varphi(y^{k+1}) > \frac{1}{v}\langle y^k - vf'(y^k) - y^{k+1}, w - y^{k+1}\rangle - \alpha\|w - y^{k+1}\|_2^2. \tag{52}$$

Then (48) follows by setting $w = y^k$. Furthermore, by the definition of $f(\cdot)$, it is of class $C_L^{1,1}$ and it follows from Bertsekas (1999, prop. A.24) that

$$\|f(y) - f(x) - f'(x)(y-x)\| \leq \frac{L}{2}\|y - x\|^2 \quad \text{for any } x, y.$$

Then, by the definition of D_k, it follows that

$$\begin{aligned}(f+\varphi)(y^{k+1}) - (f+\varphi)(y^k) + D_k &= f(y^{k+1}) - f(y^k) + \langle f'(y^k), y^k - y^{k+1}\rangle \\ &\leq \tfrac{L}{2}\|y^k - y^{k+1}\|_2^2 \\ &\leq \tfrac{Lv}{2(1-v\alpha)} D_k,\end{aligned}$$

where the last inequality follows from (48), and thus, (49) is proved. ■

The main result of this subsection is presented as follows, in which we establish the local linear convergence rate of the PGM-GSO to a local minimum for the case when $p = 1$ and $0 < q < 1$ under some mild assumptions.

Theorem 16 *Let $\{x^k\}$ be a sequence generated by the PGM-GSO with $v < \frac{1}{2}\|A\|_2^{-2}$. Then $\{x^k\}$ converges to a critical point x^* of problem (4). Assume that x^* is a local minimum of problem (4). Suppose that any nonzero group of x^* is active, and the columns of B are linearly independent. Then there exist $N \in \mathbb{N}$, $C > 0$ and $\eta \in (0,1)$ such that*

$$F(x^k) - F(x^*) \leq C\eta^k \quad \text{and} \quad \|x^k - x^*\|_2 \leq C\eta^k \quad \textit{for any } k \geq N. \tag{53}$$

Proof The convergence of $\{x^k\}$ to a critical point x^* of problem (4) directly follows from Theorem 12. Let D_k, N and δ be defined as in Lemma 15, and let

$$r_k := F(x^k) - F(x^*).$$

Note in (50) that $y^k \neq_{\mathbf{a}} 0$ and $x^k_{\mathcal{G}^c_{\mathcal{S}}} = 0$ for any $k \geq N$. Thus

$$r_k = (f+\varphi)(y^k) - (f+\varphi)(y^*) \quad \text{for any } k \geq N.$$

It is trivial to see that $\varphi(\cdot)$ is smooth around y^* (as it is active) and that

$$\varphi''(y^*) = \lambda q(q-1)\begin{pmatrix} M_1^* & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & M_S^* \end{pmatrix} \prec 0, \quad f''(y^*) + \varphi''(y^*) \succ 0;$$

as shown in (42). This shows that $\varphi(\cdot)$ is concave around y^*, while $(f+\varphi)(\cdot)$ is convex around y^*. Without loss of generality, we assume that $\varphi(\cdot)$ is concave and $(f+\varphi)(\cdot)$ is convex in $\mathbf{B}(y^*, \delta)$ and that $y^k \in \mathbf{B}(y^*, \delta)$ for any $k \geq N$ (since $\{y^k\}$ converges to y^*).

By the convexity of $(f+\varphi)(\cdot)$ in $\mathbf{B}(y^*, \delta)$, it follows that for any $k \geq N$

$$\begin{aligned} r_k &= (f+\varphi)(y^k) - (f+\varphi)(y^*) \\ &\leq \langle f'(y^k) + \varphi'(y^k), y^k - y^*\rangle \\ &= \langle f'(y^k) + \varphi'(y^k), y^k - y^{k+1}\rangle + \langle f'(y^k) + \varphi'(y^k), y^{k+1} - y^*\rangle \\ &= D_k - \varphi(y^k) + \varphi(y^{k+1}) + \langle\varphi'(y^k), y^k - y^{k+1}\rangle + \langle f'(y^k) + \varphi'(y^k), y^{k+1} - y^*\rangle. \end{aligned} \tag{54}$$

Noting that $\varphi(\cdot)$ is concave in $\mathbf{B}(y^*, \delta)$, it follows that

$$\varphi(y^k) - \varphi(y^{k+1}) \geq \langle\varphi'(y^k), y^k - y^{k+1}\rangle.$$

Consequently, (54) is reduced to

$$\begin{aligned} r_k &\le D_k + \langle f'(y^k) + \varphi'(y^k), y^{k+1} - y^* \rangle \\ &= D_k + \langle \varphi'(y^k) - \varphi'(y^{k+1}), y^{k+1} - y^* \rangle + \langle f'(y^k) + \varphi'(y^{k+1}), y^{k+1} - y^* \rangle \\ &\le D_k + \left(L_\varphi + \tfrac{1}{v}\right) \|y^k - y^{k+1}\|_2 \|y^{k+1} - y^*\|_2, \end{aligned} \tag{55}$$

where the last inequality follows from the smoothness of φ on $\mathbf{B}(y^*, \delta)$ and (51), and L_φ is the Lipschitz constant of $\varphi'(\cdot)$ on $\mathbf{B}(y^*, \delta)$. Let $\beta := 1 - \frac{Lv}{2(1-v\alpha)} \in (0,1)$ (due to the assumption $v < \frac{1}{L}$). Then, (49) is reduced to

$$r_k - r_{k+1} = (f+\varphi)(y^k) - (f+\varphi)(y^{k+1}) \ge \beta D_k > 0,$$

and thus, it follows from (55) and (48) that

$$\begin{aligned} \beta r_k &\le \beta D_k + \beta \left(L_\varphi + \tfrac{1}{v}\right) \|y^k - y^{k+1}\|_2 \|y^{k+1} - y^*\|_2 \\ &\le r_k - r_{k+1} + \beta \left(L_\varphi + \tfrac{1}{v}\right) \|y^{k+1} - y^*\|_2 \sqrt{\tfrac{v}{1-v\alpha} D_k} \\ &\le r_k - r_{k+1} + \left(L_\varphi + \tfrac{1}{v}\right) \sqrt{\tfrac{v\beta}{1-v\alpha}} \|y^{k+1} - y^*\|_2 \sqrt{r_k - r_{k+1}}. \end{aligned} \tag{56}$$

Recall from Lemma 14(ii) that there exists $c > 0$ such that

$$\|y - y^*\|_2^2 \le c\left((f+\varphi)(y) - (f+\varphi)(y^*)\right) \quad \text{for any } y \in \mathbf{B}(y^*, \delta).$$

Thus, it follows that

$$\|y^{k+1} - y^*\|_2^2 \le c r_{k+1} \le c r_k \quad \text{for any } k \ge N. \tag{57}$$

Let $\epsilon := \frac{c}{\beta}\left(L_\varphi + \frac{1}{v}\right)^2$. By Young's inequality, (56) yields that

$$\begin{aligned} \beta r_k &\le r_k - r_{k+1} + \tfrac{1}{2\epsilon} \|y^{k+1} - y^*\|_2^2 \left(L_\varphi + \tfrac{1}{v}\right)^2 + \tfrac{ev\beta}{2(1-v\alpha)}(r_k - r_{k+1}) \\ &\le r_k - r_{k+1} + \tfrac{\beta}{2} r_k + \tfrac{cv}{2(1-v\alpha)} \left(L_\varphi + \tfrac{1}{v}\right)^2 (r_k - r_{k+1}). \end{aligned} \tag{58}$$

Let $\gamma := \frac{cv}{2(1-v\alpha)}\left(L_\varphi + \frac{1}{v}\right)^2 > 0$. Then, (58) is reduced to

$$r_{k+1} \le \frac{1 + \gamma - \frac{\beta}{2}}{1+\gamma} r_k = \eta_1 r_k,$$

where $\eta_1 := \frac{1+\gamma-\frac{\beta}{2}}{1+\gamma} \in (0,1)$. Thus, by letting $C_1 := r_N \eta_1^{-N}$, it follows that

$$r_k \le \eta_1^{k-N} r_N = C_1 \eta_1^k \quad \text{for any } k \ge N.$$

By letting $\eta_2 = \sqrt{\eta_1}$ and $C_2 = \sqrt{cC_1}$, it follows from (57) that

$$\|x^k - x^*\|_2 = \|y^k - y^*\|_2 \le (cr_k)^{1/2} \le C_2 \eta_2^k \quad \text{for any } k \ge N.$$

Letting $C := \max\{C_1, C_2\}$ and $\eta := \max\{\eta_1, \eta_2\}$, we obtain (53). The proof is complete. ■

Theorem 16 establishes the linear convergence rate of the PGM for solving the $\ell_{1,q}$ regularization problem under two assumptions: (i) the critical point x^* of the sequence produced by the PGM is a local minimum of problem (4), and (ii) any nonzero group of the local minimum is an active one. The assumption (i) is important by which we are able to establish a second-order growth property, which plays a crucial role in our analysis. Note that the assumption (ii) is satisfied automatically for the sparse optimization problem ($n_{max} = 1$). Hence, when $n_{max} = 1$, we obtain the linear convergence rate of the PGM for solving ℓ_q regularization problem ($0 < q < 1$). This result is stated below as a corollary.

Corollary 17 *Let $0 < q < 1$, and let $\{x^k\}$ be a sequence generated by the PGM for solving the following ℓ_q regularization problem*

$$\min_{x \in \mathbb{R}^n} F(x) := \|Ax - b\|_2^2 + \lambda \|x\|_q^q \tag{59}$$

with $v < \frac{1}{2}\|A\|_2^{-2}$. Then $\{x^k\}$ converges to a critical point x^* of problem (59). Further assume that x^* is a local minimum of problem (59). Then there exist $N \in \mathbb{N}$, $C > 0$ and $\eta \in (0,1)$ such that

$$F(x^k) - F(x^*) \le C\eta^k \quad \text{and} \quad \|x^k - x^*\|_2 \le C\eta^k \quad \text{for any } k \ge N.$$

While we are carrying out the revision of our manuscript, we have found that the local linear convergence rate of the PGM has been studied in the literature. On one hand, the local linear convergence rate of the PGM for solving the ℓ_1 regularization problem (ISTA) has been established under some assumptions in Bredies and Lorenz (2008); Hale et al. (2008); Tao et al. (2016), and, under the framework of the so-called KL theory, it is established that the sequence generated by the PGM linearly converges to a critical point of a KL function if its KL exponent is in $(0, \frac{1}{2}]$; see Frankel et al. (2015); Li and Pong (2016); Xu and Yin (2013). However, the KL exponent of the ℓ_q regularized function is still unknown, and thus the linear convergence result in these references cannot directly be applied to the ℓ_q regularization problem. On the other hand, Zeng et al. (2015) has obtained the linear convergence rate of the ℓ_q regularization problem under the framework of a restricted KL property. However, it seems that their result is restrictive as it is assumed that the stepsize v and the regularization component q satisfy

$$\frac{q}{2} < \frac{\lambda_{\min}(A_{\mathcal{S}}^T A_{\mathcal{S}})}{\|A\|_2^2} \quad \text{and} \quad \frac{q}{4\lambda_{\min}(A_{\mathcal{S}}^T A_{\mathcal{S}})} < v < \frac{1}{2\|A\|_2^2},$$

where $\mathcal{S}$ is the active index of the limiting point x^*, while our result in Corollary 17 holds for all the stepsize $v < \frac{1}{2}\|A\|_2^{-2}$ and the regularization component $0 < q < 1$.

3.2 Analytical Solutions of Proximal Subproblems

Since the main computation of the PGM is the proximal step (37), it is significant to investigate the solutions of subproblem (37) for the specific applications so as to spread the application of the PGM. Note that $\|x\|_{p,q}^q$ and $\|x - z^k\|_2^2$ are both grouped separable. Then

the proximal step (37) can be achieved parallelly in each group, and is equivalent to solve a cycle of low dimensional proximal optimization subproblems

$$x_{\mathcal{G}_i}^{k+1} \in \arg\min_{x\in\mathbb{R}^{n_i}} \left\{ \lambda\|x_{\mathcal{G}_i}\|_p^q + \frac{1}{2v}\|x_{\mathcal{G}_i} - z_{\mathcal{G}_i}^k\|_2^2 \right\} \quad \text{for } i = 1, \cdots, r. \tag{60}$$

When p and q are given as some specific numbers, such as $p = 1, 2$ and $q = 0, 1/2, 2/3, 1$, the solution of subproblem (60) of each group can be given explicitly by an analytical formula, as shown in the following proposition.

Proposition 18 *Let $z \in \mathbb{R}^l$, $v > 0$ and the proximal regularization be*

$$Q_{p,q}(x) := \lambda\|x\|_p^q + \frac{1}{2v}\|x - z\|_2^2 \quad \textit{for any } x \in \mathbb{R}^l.$$

Then the proximal operator

$$P_{p,q}(z) \in \arg\min_{x\in\mathbb{R}^l} \{Q_{p,q}(x)\}$$

has the following analytical formula:

(i) *if $p = 2$ and $q = 1$, then*

$$P_{2,1}(z) = \begin{cases} \left(1 - \frac{v\lambda}{\|z\|_2}\right) z, & \|z\|_2 > v\lambda, \\ 0, & \text{otherwise}, \end{cases}$$

(ii) *if $p = 2$ and $q = 0$, then*

$$P_{p,0}(z) = \begin{cases} z, & \|z\|_2 > \sqrt{2v\lambda}, \\ 0 \text{ or } z, & \|z\|_2 = \sqrt{2v\lambda}, \\ 0, & \|z\|_2 < \sqrt{2v\lambda}, \end{cases}$$

(iii) *if $p = 2$ and $q = 1/2$, then*

$$P_{2,1/2}(z) = \begin{cases} \frac{16\|z\|_2^{3/2}\cos^3\left(\frac{\pi}{3} - \frac{\psi(z)}{3}\right)}{3\sqrt{3}v\lambda + 16\|z\|_2^{3/2}\cos^3\left(\frac{\pi}{3} - \frac{\psi(z)}{3}\right)} z, & \|z\|_2 > \frac{3}{2}(v\lambda)^{2/3}, \\ 0 \text{ or } \frac{16\|z\|_2^{3/2}\cos^3\left(\frac{\pi}{3} - \frac{\psi(z)}{3}\right)}{3\sqrt{3}v\lambda + 16\|z\|_2^{3/2}\cos^3\left(\frac{\pi}{3} - \frac{\psi(z)}{3}\right)} z, & \|z\|_2 = \frac{3}{2}(v\lambda)^{2/3}, \\ 0, & \|z\|_2 < \frac{3}{2}(v\lambda)^{2/3}, \end{cases} \tag{61}$$

with

$$\psi(z) = \arccos\left(\frac{v\lambda}{4}\left(\frac{3}{\|z\|_2}\right)^{3/2}\right), \tag{62}$$

(iv) *if $p = 1$ and $q = 1/2$, then*

$$P_{1,1/2}(z) = \begin{cases} \tilde{z}, & Q_{1,1/2}(\tilde{z}) < Q_{1,1/2}(0), \\ 0 \text{ or } \tilde{z}, & Q_{1,1/2}(\tilde{z}) = Q_{1,1/2}(0), \\ 0, & Q_{1,1/2}(\tilde{z}) > Q_{1,1/2}(0), \end{cases}$$

with

$$\tilde{z} = z - \frac{\sqrt{3}v\lambda}{4\sqrt{\|z\|_1}\cos\left(\frac{\pi}{3} - \frac{\xi(z)}{3}\right)}\text{sign}(z), \quad \xi(z) = \arccos\left(\frac{v\lambda l}{4}\left(\frac{3}{\|z\|_1}\right)^{3/2}\right),$$

(v) *if $p = 2$ and $q = 2/3$, then*

$$P_{2,2/3}(z) = \begin{cases} \frac{3\left(a^{3/2} + \sqrt{2\|z\|_2 - a^3}\right)}{32v\lambda a^2 + 3\left(a^{3/2} + \sqrt{2\|z\|_2 - a^3}\right)} z, & \|z\|_2 > 2\left(\frac{2}{3}v\lambda\right)^{3/4}, \\ 0 \text{ or } \frac{3\left(a^{3/2} + \sqrt{2\|z\|_2 - a^3}\right)}{32v\lambda a^2 + 3\left(a^{3/2} + \sqrt{2\|z\|_2 - a^3}\right)} z, & \|z\|_2 = 2\left(\frac{2}{3}v\lambda\right)^{3/4}, \\ 0, & \|z\|_2 < 2\left(\frac{2}{3}v\lambda\right)^{3/4}, \end{cases} \tag{63}$$

with

$$a = \frac{2}{\sqrt{3}}(2v\lambda)^{1/4}\left(\cosh\left(\frac{\varphi(z)}{3}\right)\right)^{1/2}, \quad \varphi(z) = \text{arccosh}\left(\frac{27\|z\|_2^2}{16(2v\lambda)^{3/2}}\right), \tag{64}$$

(vi) *if $p = 1$ and $q = 2/3$, then*

$$P_{1,2/3}(z) = \begin{cases} \bar{z}, & Q_{1,2/3}(\bar{z}) < Q_{1,2/3}(0), \\ 0 \text{ or } \bar{z}, & Q_{1,2/3}(\bar{z}) = Q_{1,2/3}(0), \\ 0, & Q_{1,2/3}(\bar{z}) > Q_{1,2/3}(0), \end{cases}$$

with

$$\bar{z} = z - \frac{4v\lambda\bar{a}^{1/2}}{3\left(\bar{a}^{3/2} + \sqrt{2\|z\|_1 - \bar{a}^3}\right)}\text{sign}(z),$$

and

$$\bar{a} = \frac{2}{\sqrt{3}}(2v\lambda l)^{1/4}\left(\cosh\left(\frac{\zeta(z)}{3}\right)\right)^{1/2}, \quad \zeta(z) = \text{arccosh}\left(\frac{27\|z\|_1^2}{16(2v\lambda l)^{3/2}}\right).$$

Proof Since the proximal regularization $Q_{p,q}(\cdot) := \lambda\|\cdot\|_p^q + \frac{1}{2v}\|\cdot - z\|_2^2$ is non-differentiable only at 0, $P_{p,q}(z)$ must be 0 or some point $\tilde{x}(\neq 0)$ satisfying the first-order optimality condition

$$\lambda q\|\tilde{x}\|_p^{q-p}\begin{pmatrix} |\tilde{x}_1|^{p-1}\text{sign}(\tilde{x}_1) \\ \vdots \\ |\tilde{x}_l|^{p-1}\text{sign}(\tilde{x}_l) \end{pmatrix} + \frac{1}{v}(\tilde{x} - z) = 0. \tag{65}$$

Thus, to derive the analytical formula of the proximal operator $P_{p,q}(z)$, we just need to calculate such $\tilde{x}$ via (65), and then compare the objective function values $Q_{p,q}(\tilde{x})$ and $Q_{p,q}(0)$ to obtain the solution inducing a smaller value. The proofs of the six statements follow in the above routine, and we only provide the detailed proofs of (iii) and (v) as samples.

(iii) When $p = 2$ and $q = 1/2$, (65) reduces to

$$\frac{\lambda\tilde{x}}{2\|\tilde{x}\|_2^{3/2}} + \frac{1}{v}(\tilde{x} - z) = 0, \tag{66}$$

and consequently,

$$\|\tilde{x}\|_2^{3/2} - \|z\|_2\|\tilde{x}\|_2^{1/2} + \frac{1}{2}v\lambda = 0. \tag{67}$$

Denote $\eta = \|\tilde{x}\|_2^{1/2} > 0$. The equation (67) can be transformed into the following cubic algebraic equation

$$\eta^3 - \|z\|_2\eta + \frac{1}{2}v\lambda = 0. \tag{68}$$

Due to the hyperbolic solution of the cubic equation (see Short, 1937), by denoting

$$r = 2\sqrt{\frac{\|z\|_2}{3}},\ \alpha = \arccos\left(\frac{v\lambda}{4}\left(\frac{3}{\|z\|_2}\right)^{3/2}\right) \text{ and } \beta = \operatorname{arccosh}\left(-\frac{v\lambda}{4}\left(\frac{3}{\|z\|_2}\right)^{3/2}\right),$$

the solution of (68) can be expressed as the follows.

(1) If $0 \le \|z\|_2 \le 3\left(\frac{v\lambda}{4}\right)^{2/3}$, then the three roots of (68) are given by

$$\eta_1 = r\cosh\frac{\beta}{3},\ \eta_2 = -\frac{r}{2}\cosh\frac{\beta}{3} + i\frac{\sqrt{3}r}{2}\sinh\frac{\beta}{3},\ \eta_3 = -\frac{r}{2}\cosh\frac{\beta}{3} - i\frac{\sqrt{3}r}{2}\sinh\frac{\beta}{3},$$

where i denotes the imaginary unit. However, this β does not exist since the value of hyperbolic cosine must be positive. Thus, in this case, $P_{2,1/2}(z) = 0$.

(2) If $\|z\|_2 > 3\left(\frac{v\lambda}{4}\right)^{2/3}$, then the three roots of (68) are

$$\eta_1 = r\cos\left(\frac{\pi}{3} - \frac{\alpha}{3}\right),\ \eta_2 = -r\sin\left(\frac{\pi}{2} - \frac{\alpha}{3}\right),\ \eta_3 = -r\cos\left(\frac{2\pi}{3} - \frac{\alpha}{3}\right).$$

The unique positive solution of (68) is $\|\tilde{x}\|_2^{1/2} = \eta_1$, and thus, the unique solution of (66) is given by

$$\tilde{x} = \frac{2\eta_1^3}{v\lambda + 2\eta_1^3}z = \frac{16\|z\|_2^{3/2}\cos^3\left(\frac{\pi}{3} - \frac{\psi(z)}{3}\right)}{3\sqrt{3}v\lambda + 16\|z\|_2^{3/2}\cos^3\left(\frac{\pi}{3} - \frac{\psi(z)}{3}\right)}z.$$

Finally, we compare the objective function values $Q_{2,1/2}(\tilde{x})$ and $Q_{2,1/2}(0)$. For this purpose, when $\|z\|_2 > 3\left(\frac{v\lambda}{4}\right)^{2/3}$, we define

$$\begin{aligned} H(\|z\|_2) &:= \tfrac{v}{\|\tilde{x}\|_2}\left(Q_{2,1/2}(0) - Q_{2,1/2}(\tilde{x})\right) \\ &= \tfrac{v}{\|\tilde{x}\|_2}\left(\tfrac{1}{2v}\|z\|_2^2 - \lambda\|\tilde{x}\|_2^{1/2} - \tfrac{1}{2v}\|\tilde{x} - z\|_2^2\right) \\ &= \|z\|_2 - \tfrac{\|\tilde{x}\|_2^2 + 2v\lambda\|\tilde{x}\|_2^{1/2}}{2\|\tilde{x}\|_2} \\ &= \tfrac{1}{2}\|z\|_2 - \tfrac{3}{4}v\lambda\|\tilde{x}\|_2^{-1/2}, \end{aligned}$$

where the third equality holds since that $\tilde{x}$ is proportional to z, and fourth equality follows from (67). Since both $\|z\|_2$ and $\|\tilde{x}\|_2$ are strictly increasing on $\|z\|_2$, $H(\|z\|_2)$ is also strictly increasing when $\|z\|_2 > 3\left(\frac{v\lambda}{4}\right)^{2/3}$. Thus the unique solution of $H(\|z\|_2) = 0$ satisfies

$$\|z\|_2\|\tilde{x}\|_2^{1/2} = \frac{3}{2}v\lambda,$$

and further, (67) implies that the solution of $H(\|z\|_2) = 0$ is

$$\|z\|_2 = \frac{3}{2}(v\lambda)^{2/3}.$$

Therefore, we arrive at the formulae (61) and (62).
(v) When $p = 2$ and $q = 2/3$, (65) reduces to

$$\frac{2\lambda\tilde{x}}{3\|\tilde{x}\|_2^{4/3}} + \frac{1}{v}(\tilde{x} - z) = 0, \tag{69}$$

and consequently,

$$\|\tilde{x}\|_2^{4/3} - \|z\|_2\|\tilde{x}\|_2^{1/3} + \frac{2}{3}v\lambda = 0. \tag{70}$$

Denote $\eta = \|\tilde{x}\|_2^{1/3} > 0$ and $h(t) = t^4 - \|z\|_2 t + \frac{2}{3}v\lambda$ for any $t \in \mathbb{R}$. Thus, η is the positive solution of $h(t) = 0$. Next, we seek η by the method of undetermined coefficients. Assume that

$$h(t) = t^4 - \|z\|_2 t + \frac{2}{3}v\lambda = (t^2 + at + b)(t^2 + ct + d), \quad \text{where } a, b, c, d \in \mathbb{R}. \tag{71}$$

By expansion and comparison, we have that

$$a + c = 0, \quad b + d + ac = 0, \quad ad + bc = -\|z\|_2, \quad bd = \frac{2}{3}v\lambda,$$

and thus,

$$c = -a, b = \frac{1}{2}\left(a^2 + \frac{\|z\|_2}{a}\right), d = \frac{1}{2}\left(a^2 - \frac{\|z\|_2}{a}\right), bd = \frac{1}{4}\left(a^4 - \frac{\|z\|_2^2}{a^2}\right) = \frac{2}{3}v\lambda. \tag{72}$$

By letting $M = a^2$, the last one of the above equalities reduces to the following cubic algebraic equation

$$M^3 - \frac{8}{3}v\lambda M - \|z\|_2^2 = 0. \tag{73}$$

According to the Cardano formula for the cubic equation, the root of (73) can be represented by

$$a^2 = M = \left(\frac{\|z\|_2^2}{2} + \sqrt{\frac{\|z\|_2^4}{4} - \left(\frac{8}{9}v\lambda\right)^3}\right)^{1/3} + \left(\frac{\|z\|_2^2}{2} - \sqrt{\frac{\|z\|_2^4}{4} - \left(\frac{8}{9}v\lambda\right)^3}\right)^{1/3},$$

which can also be reformulated in the following hyperbolic form (see Short, 1937)

$$a^2 = M = \frac{4}{3}\sqrt{2v\lambda}\cosh\left(\frac{\varphi(z)}{3}\right), \tag{74}$$

where $\varphi(z)$ is given by (64). By (71) and (72), we have that η, the positive root of $h(t) = 0$, satisfies

$$\eta^2 + a\eta + \frac{1}{2}\left(a^2 + \frac{\|z\|_2}{a}\right) = 0 \quad \text{or} \quad \eta^2 - a\eta + \frac{1}{2}\left(a^2 - \frac{\|z\|_2}{a}\right) = 0.$$

Hence, the real roots of the above equations, that is, the real roots of $h(t) = 0$, are

$$\eta_1 = \frac{1}{2}\left(|a| + \sqrt{\frac{2\|z\|_2}{|a|} - a^2}\right) \quad \text{and} \quad \eta_2 = \frac{1}{2}\left(|a| - \sqrt{\frac{2\|z\|_2}{|a|} - a^2}\right). \tag{75}$$

It is easy to see that $\eta_1 > \eta_2$ and that η_2 should be discarded as it induces the saddle point rather than a minimum (since $h(t) > 0$ when $t < \eta_2$). Thus, by (69), (74) and (75), one has

$$\tilde{x} = \frac{3\eta_1^4}{2v\lambda + 3\eta_1^4} z = \frac{3\left(a^{3/2} + \sqrt{2\|z\|_2 - a^3}\right)}{32v\lambda a^2 + 3\left(a^{3/2} + \sqrt{2\|z\|_2 - a^3}\right)} z,$$

where a is given by (64). Finally, we compare the objective function values $Q_{2,2/3}(\tilde{x})$ and $Q_{2,2/3}(0)$. For this purpose, we define

$$\begin{aligned} H(\|z\|_2) &:= \tfrac{v}{\|\tilde{x}\|_2}\left(Q_{2,2/3}(0) - Q_{2,2/3}(\tilde{x})\right) \\ &= \tfrac{v}{\|\tilde{x}\|_2}\left(\tfrac{1}{2v}\|z\|_2^2 - \lambda\|\tilde{x}\|_2^{2/3} - \tfrac{1}{2v}\|\tilde{x} - z\|_2^2\right) \\ &= \|z\|_2 - \tfrac{\|\tilde{x}\|_2^2 + 2v\lambda\|\tilde{x}\|_2^{2/3}}{2\|\tilde{x}\|_2} \\ &= \tfrac{1}{2}\|z\|_2 - \tfrac{2}{3}v\lambda\|\tilde{x}\|_2^{-1/3}, \end{aligned}$$

where the third equality holds since that $\tilde{x}$ is proportional to z, and fourth equality follows from (70). Since both $\|z\|_2$ and $\|\tilde{x}\|_2$ are strictly increasing on $\|z\|_2$, $H(\|z\|_2)$ is also strictly increasing when $\|z\|_2 > 4(\frac{2}{9}v\lambda)^{3/4}$. Thus the unique solution of $H(\|z\|_2) = 0$ satisfies

$$\|z\|_2\|\tilde{x}\|_2^{1/3} = \frac{4}{3}v\lambda,$$

and further, (70) implies that the solution of $H(\|z\|_2) = 0$ is

$$\|z\|_2 = 2\left(\frac{2}{3}v\lambda\right)^{3/4}.$$

Therefore, we arrive at the formulae (63) and (64). The proof is complete. ■

Remark 19 *Note from Proposition 18 that the solutions of the proximal optimization subproblems might not be unique when $Q_{p,q}(\tilde{x}) = Q_{p,q}(0)$. To avoid this obstacle in numerical computations, we select the solution $P_{p,q}(z) = 0$ whenever $Q_{p,q}(\tilde{x}) = Q_{p,q}(0)$, which achieves a more sparse solution, in the definition of the proximal operator to guarantee a unique update.*

Remark 20 *By Proposition 18, one sees that the PGM-GSO meets the group sparsity structure, since the components of each iterate within each group are likely to be either all zeros or all nonzeros. When $n_{\max} = 1$, the data do not form any group structure in the feature space, and the sparsity is achieved only on the individual feature level. In this case, the proximal operators $P_{2,1}(z)$, $P_{2,0}(z)$, and $P_{2,1/2}(z)$ and $P_{1,1/2}(z)$ reduce to the soft thresholding function in Daubechies et al. (2004), the hard thresholding function in Blumensath and Davies (2008) and the half thresholding function in Xu et al. (2012), respectively.*

Remark 21 *Proposition 18 presents the analytical solution of the proximal optimization subproblems* (60) *when $q = 0, 1/2, 2/3, 1$. However, in other cases, the analytical solution of problem* (60) *seems not available, since the algebraic equation* (65) *does not have an analytical solution (it is difficult to find an analytical solution for the algebraic equation whose order is larger than four). Thus, in the general cases of $q \in (0, 1)$, we alternatively use the Newton method to solve the nonlinear equation* (65)*, which is the optimality condition of the proximal optimization subproblem. The numerical simulation in Figure 5 of section 4 shows that the Newton method works in solving the proximal optimization subproblems* (60) *for the general q, while the $\ell_{p,1/2}$ regularization is the best one among the $\ell_{p,q}$ regularizations for $q \in [0, 1]$.*

4. Numerical Experiments

The purpose of this section is to carry out the numerical experiments of the proposed PGM for the $\ell_{p,q}$ regularization problem. We illustrate the performance of the PGM-GSO among different types of $\ell_{p,q}$ regularization, in particular, when $(p, q) =$(2,1), (2,0), (2,1/2), (1,1/2), (2,2/3) and (1,2/3), by comparing them with several state-of-the-art algorithms for simulated data and applying them to infer gene regulatory network from gene expression data of mouse embryonic stem cell. All numerical experiments are implemented in Matlab R2013b and executed on a personal desktop (Intel Core Duo E8500, 3.16 GHz, 4.00 GB of RAM). The R package of the PGM for solving group sparse optimization, named GSparO in CRAN, is available at https://CRAN.R-project.org/package=GSparO

4.1 Simulated Data

In the numerical experiments on simulated data, the numerical data are generated as follows. We first randomly generate an i.i.d. Gaussian ensemble $A \in \mathbb{R}^{m\times n}$ satisfying $A^\top A = I$. Then we generate a group sparse solution $\bar{x} \in \mathbb{R}^n$ via randomly splitting its components into r groups and randomly picking k of them as active groups, whose entries are also randomly generated as i.i.d. Gaussian, while the remaining groups are all set to be zeros. We generate the data b by the Matlab script

$$b = A * \bar{x} + sigma * randn(m, 1),$$

where *sigma* is the standard deviation of additive Gaussian noise. The problem size is set to $n = 1024$ and $m = 256$, and we test on the noisy measurement data with *sigma* $= 0.1\%$. Assuming the group sparsity level S is predefined, the regularization parameter λ is iteratively updated by obeying the rule: we set the iterative threshold to be the S-th largest value of $\|z^k_{\mathcal{G}_i}\|_2$ and solve the λ by virtue of Proposition 18.

For each given sparsity level, which is k/r, we randomly generate the data A, $\bar{x}$, b (as above) 500 times, run the algorithm, and average the 500 numerical results to illustrate the performance of the algorithm. We choose the stepsize $v = 1/2$ in the tests of the PGM-GSO. Two key criteria to characterize the performance are the relative error $\|x - \bar{x}\|_2/\|\bar{x}\|_2$ and the successful recovery rate, where the recovery is defined as *success* when the relative error between the recovered data and the true data is smaller than 0.5%; otherwise, it is regarded as *failure*.

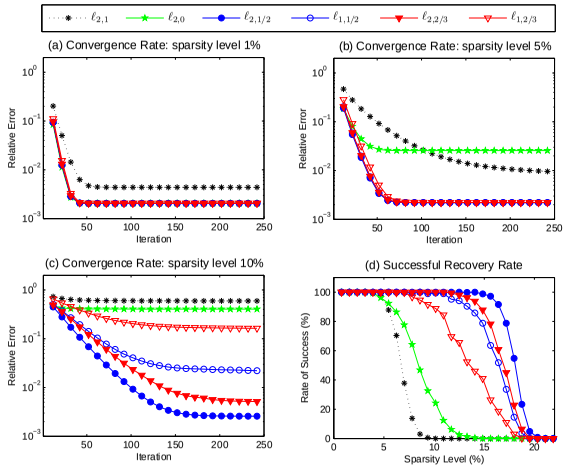


Figure 3: Convergence results and recovery rates for different sparsity levels.

We carry out six experiments with the initial point $x_0 = 0$ (unless otherwise specified). In the first experiment, setting $r = 128$ (so group size $G = 1024/128 = 8$), we compare the convergence rates and the successful recovery rates of the PGM-GSO with $(p, q) = (2, 1), (2, 0), (2, 1/2), (1, 1/2), (2, 2/3)$ and $(1, 2/3)$ for different sparsity levels. In Figure 3, (a), (b), and (c) illustrate the convergence rate results on sparsity level 1%, 5%, and 10%, respectively, while (d) plots the successful recovery rates on different sparsity levels. When the solution is of high sparse level, as shown in Figure 3(a), all $\ell_{p,q}$ regularization problems perform perfect and achieve a fast convergence rate. As demonstrated in Figure 3(b), when the sparsity level drops to 5%, $\ell_{p,1/2}$ and $\ell_{p,2/3}$ ($p = 1$ and 2) perform better and arrive at a more accurate level than $\ell_{2,1}$ and $\ell_{2,0}$. As illustrated in Figure 3(c), when the sparsity level is 10%, $\ell_{p,1/2}$ further outperforms $\ell_{p,2/3}$ ($p = 1$ or 2), and it surprises us that $\ell_{2,q}$ performs better than $\ell_{1,q}$ ($q = 1/2$ or $2/3$). From Figure 3(d), it is illustrated that $\ell_{p,1/2}$ achieves a better successful recovery rate than $\ell_{p,2/3}$ ($p = 1$ or 2), which outperforms $\ell_{2,0}$ and $\ell_{2,1}$. Moreover, we surprisingly see that $\ell_{2,q}$ also outperforms $\ell_{1,q}$ ($q = 1/2$ or $2/3$) on the successful recovery rate. In a word, $\ell_{2,1/2}$ performs as the best one of these six regularizations on both accuracy and robustness. In this experiment, we also note that the running times are at a same level, about 0.9 second per 500 iterations.

The second experiment is performed to show the sensitivity analysis on the group size ($G = 4, 8, 16, 32$) of the PGM-GSO with the six types of $\ell_{p,q}$ regularization. As shown in Figure 4, the six types of $\ell_{p,q}$ regularization reach a higher successful recovery rate for the larger group size. We also note that the larger the group size, the shorter the running time.

The third experiment is implemented to study the variation of the PGM-GSO when varying the regularization order q (fix $p = 2$). Recall from Theorem 18 that the analytical solu-

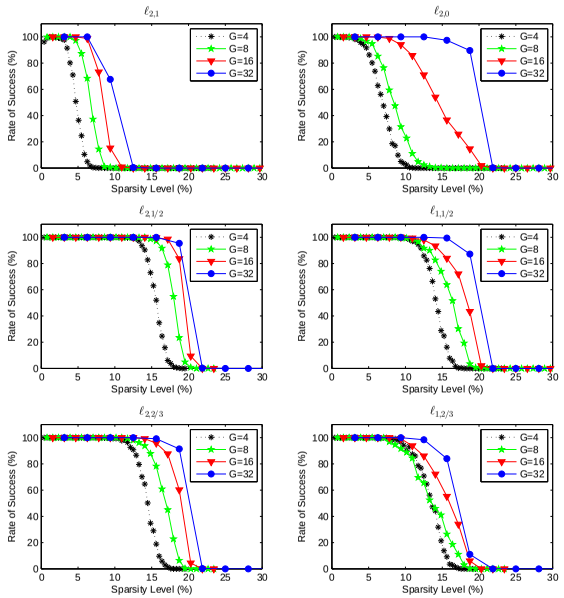


Figure 4: Sensitivity analysis on group size.

tion of the proximal optimization subproblem (60) can be obtained when $q = 0, 1/2, 2/3, 1$. However, in other cases, the analytical solution of subproblem (60) seems not available, and thus we apply the Newton method to solve the nonlinear equation (65), which is the optimality condition of the proximal optimization subproblem. Figure 5 shows the variation of successful recovery rates by decreasing the regularization order q from 1 to 0. It is illustrated that the PGM-GSO achieves the best successful recovery rate when $q = 1/2$, which arrives at the same conclusion as the first experiment. The farther the distance of q from 1/2, the lower the successful recovery rate.

The fourth experiment is to compare the PGM-GSO with several state-of-the-art algorithms in the field of sparse optimization, either convex or nonconvex algorithms, as listed in Table 1. All these algorithms, including PGM-GSO, can successfully recover the signal when the solution is of high sparse level. However, some of these algorithms fails to obtain the group sparsity structure along with the group sparsity level decreases. Figure 7 plots the signals estimated by these algorithms in a random trial at a group sparsity level of 15%. It is illustrated that the solutions of the MultiFoBa and the PGM-GSO type solvers are of group sparsity structure, while other algorithms do not obtain the group sparse solutions.

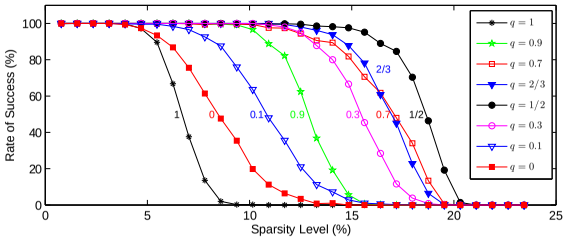


Figure 5: Variation of the PGM-GSO when varying the regularization order q.

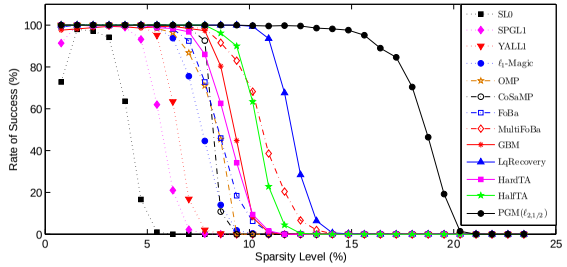


Figure 6: Comparison between the PGM-GSO and several state-of-the-art algorithms.

In these solvers, the MultiFoBa and the $\ell_{2,1}$ obtain the true active groups but inaccurate weights, the $\ell_{2,0}$ achieves the accurate weights but some incorrect active groups, while $\ell_{p,1/2}$ and $\ell_{p,2/3}$ recover perfect solutions in both true active groups and accurate weights. Figure 6 demonstrates the overall performance of these algorithms by plotting the successful recovery rates on different sparsity levels. It is indicated by Figure 6 that $\ell_{2,1/2}$ can achieve the higher successful recovery rate than other algorithms, by exploiting the group sparsity structure and lower-order regularization. From this experiment, it is demonstrated that the PGM-GSO ($\ell_{2,1/2}$) outperforms most solvers of sparse learning in solving sparse optimization problems with group structure.

The fifth experiment is devoted to the phase diagram study (see Donoho, 2006b) of the $\ell_{p,q}$ regularization problem, which further demonstrates the stronger sparsity promoting capability of $\ell_{p,1/2}$ and $\ell_{p,2/3}$ $(p = 1, 2)$ regularization over $\ell_{2,1}$ regularization. In this experiment, we consider a noiseless signal recovery problem with $n = 512$, $G = 4$, $r = 128$ and $sigma = 0$ as a prototype. More specifically, for each fixed m, we vary the number of active groups k from 1 to m/G, that is, the sparsity level varies from $1/r$ to $m/(Gr)$, and then, we increase m from G to n in the way such that 128 equidistributed values $m_j = jG$ are considered. For each specific problem size, we randomly generate the data 500 times and apply the PGM-GSO to solve the $\ell_{p,q}$ regularization problem. For these noiseless data,

SL0	SL0 is an algorithm for finding the sparsest solutions of an underdetermined system of linear equations based on Smoothed ℓ_0 norm (Mohimani et al., 2009). The package is available at http://ee.sharif.edu/~SLzero/
SPGL1	SPGL1 is a Matlab solver for large-scale sparse reconstruction (van den Berg and Friedlander, 2008). The package is available at http://www.cs.ubc.ca/~mpf/spgl1/
YALL1	YALL1 (Your ALgorithm for L1) is a package of Matlab solvers for the ℓ_1 sparse reconstruction, by virtue of the alternating direction method (Deng et al., 2011; Yang and Zhang, 2011). The package is available at http://yall1.blogs.rice.edu/
OMP	Orthogonal Matching Pursuit algorithm for the recovery of a high-dimensional sparse signal (Cai and Wang, 2011). The package is available at MathWorks.
CoSaMP	Compressive Sampling Matched Pursuit algorithms for the recovery of a high-dimensional sparse signal (Needell and Tropp, 2009). The packages are available at MathWorks.
FoBa	Adaptive forward-backward greedy algorithm for sparse learning (Zhang, 2011). The R package is available at https://CRAN.R-project.org/package=foba
MultiFoBa	MultiFoBa is group FoBa for multi-task learning (Tian et al., 2016).
ℓ_1-Magic	ℓ_1-Magic is a collection of Matlab routines for solving the convex optimization programs central to compressive sampling, based on standard interior-point methods (Candès et al., 2006a). The package is available at https://statweb.stanford.edu/~candes/l1magic/
ISTA	Iterative Shrinkage/Thresholding Algorithm (Daubechies et al., 2004) for solving the ℓ_1 regularization problem.
GBM	Gradient Based Method for solving the $\ell_{1/2}$ regularization problem (Wu et al., 2014). Suggested by the authors, we choose the initial point as the solution obtained by the ℓ_1-Magic.
LqRecovery	LqRecovery is an iterative algorithm for the ℓ_p norm minimization (Foucart and Lai, 2009). The code is available at http://www.math.drexel.edu/~foucart/software.htm
HardTA	Iterative Hard Thresholding Algorithm (Blumensath and Davies, 2008) for solving the ℓ_0 regularization problem.
HalfTA	Iterative Half Thresholding Algorithm (Xu et al., 2012) for solving the $\ell_{1/2}$ regularization problem.

Table 1: List of the state-of-the-art algorithms for sparse learning.

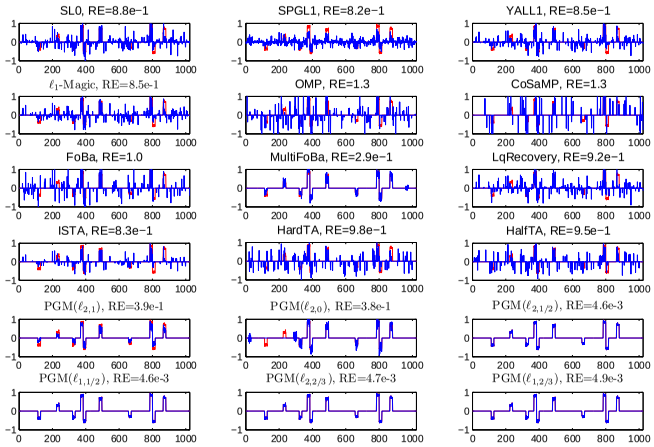


Figure 7: Simulation of the PGM-GSO and several state-of-the-art algorithms.

the recovery is defined as *success* whenever the relative error between the recovered data and the true data is smaller than 10^{-5}, otherwise, it is regarded as *failure*. Also, we embody a pixel blue whenever the point is in the case of *success*, otherwise, red when *failure*. In this way, a phase diagram of an algorithm is plotted in Figure 8, where the color of each cell reflects the empirical recovery rate (scaled between 0 and 1). It is illustrated in Figure 8 that the phase transition phenomenon does appear for the PGM-GSO with the six types of $\ell_{p,q}$ regularization. It is shown that $\ell_{p,1/2}$ and $\ell_{p,2/3}$ $(p = 1, 2)$ regularizations are more robust than that of $\ell_{2,0}$ and $\ell_{2,1}$ in the sense that they allow to achieve higher recovery rates and recover a sparse signal from a smaller amount of samples. It is also revealed by Figure 8 that $\ell_{2,1/2}$ regularization is the most robust one of these six regularizations, which achieves the same conclusion as the first and third experiments.

Even though some global optimization method, such as the filled function method (Ge, 1990), can find the global solution of the lower-order regularization problem as in Example 2, however, it does not work for the large-scale sparse optimization problems. Because, in the filled function method, all the directions need to be searched or compared in each iteration, which costs a large amount of time and hampers the efficiency for solving the large-scale problems.

4.2 Real Data in Gene Transcriptional Regulation

Gene transcriptional regulation is the process that a combination of transcription factors (TFs) act in concert to control the transcription of the target genes. Inferring gene regulatory network from high-throughput genome-wide data is still a major challenge in systems

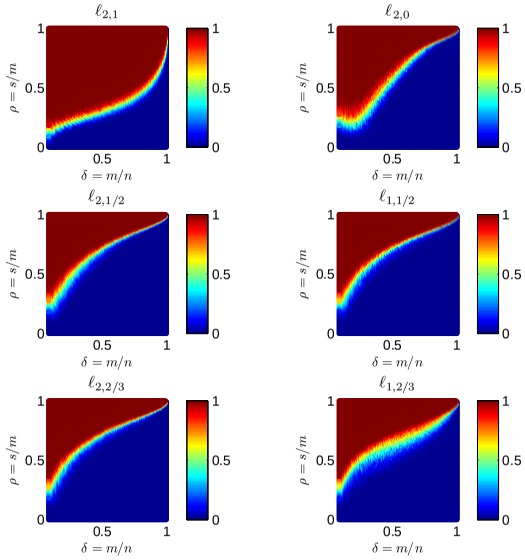


Figure 8: Phase diagram study of $\ell_{p,q}$ regularization of group sparse optimization. Blue denotes perfect recovery in all experiments, and red denotes failure for all experiments.

biology, especially when the number of genes is large but the number of experimental samples is small. In large genomes, such as human and mouse, the complexity of gene regulatory system dramatically increases. Thousands of TFs combine in different ways to regulate tens of thousands target genes in various tissues or biological processes. However, only a few TFs collaborate and usually form complexes (groups of cooperative TFs) to control the expression of a specific gene in a specific cell type or developmental stage. Thus, the prevalence of TF complex makes the solution of gene regulatory network have a group structure, and the gene regulatory network inference in such large genomes becomes a group sparse optimization problem, which is to search a small number of TF complexes (or TFs) from a pool of thousands of TF complexes (or TFs) for each target gene based on the dependencies between the expression of TF complexes (or TFs) and the targets. Even though TFs often work in the form of complexes (Xie et al., 2013), and TF complexes are very important in the control of cell identity and diseases (Hnisz et al., 2013), current methods to infer gene regulatory network usually consider each TF separately. To take the grouping information of TF complexes into consideration, we can apply the group sparse optimization to gene

regulatory network inference with the prior knowledge of TF complexes as the pre-defined grouping.

4.2.1 Materials

Chromatin immunoprecipitation (ChIP) coupled with next generation sequencing (ChIP-seq) identifies *in vivo* active and cell-specific binding sites of a TF. They are commonly used to infer TF complexes recently. Thus, we manually collect ChIP-seq data in mouse embryonic stem cells (mESCs), as shown in Table 2. Transcriptome is the gene expression profile of the whole genome that is measured by microarray or RNA-seq. The transcriptome data in mESCs for gene regulatory network inference are downloaded from Gene Expression Omnibus (GEO). 245 experiments under perturbations in mESC are collected from three papers Correa-Cerro et al. (2011); Nishiyama et al. (2009, 2013). Each experiment produced transcriptome data with or without overexpression or knockdown of a gene, in which the control and treatment have two replicates respectively. Gene expression fold changes between control samples and treatment samples of 12488 target genes in all experiments are $\log 2$ transformed and form matrix $B \in \mathbb{R}^{245\times 12488}$ (Figure 9A). The known TFs are collected from four TF databases, TRANSFAC, JASPAR, UniPROBE and TFCat, as well as literature. Let matrix $H \in \mathbb{R}^{245\times 939}$ be made up of the expression profiles of 939 known TFs, and matrix $Z \in \mathbb{R}^{939\times 12488}$ describe the connections between these TFs and targets. Then, the regulatory relationship between TFs and targets can be represented approximately by a linear system

$$HZ = B + \epsilon.$$

The TF-target connections defined by ChIP-seq data are converted into an initial matrix Z^0 (see Qin et al., 2014). Indeed, if TF i has a binding site around the gene j promoter within a defined distance (10 kbp), a non-zero number is assigned on Z^0_{ij} as a prior value.

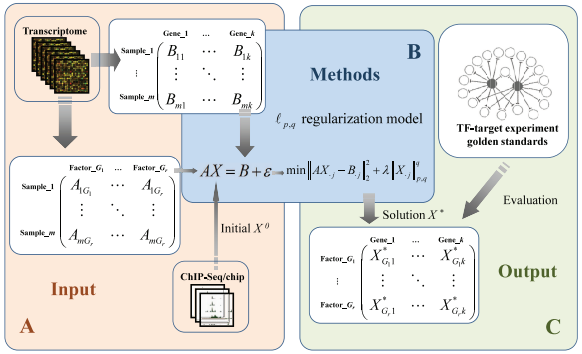


Figure 9: Workflow of gene regulatory network inference with $\ell_{p,q}$ regularization.

Now we add the grouping information (TF complexes) into this linear system. The TF complexes are inferred from ChIP-seq data (Table 2) via the method described in Giannopoulou and Elemento (2013). Let the group structure of Z be a matrix $W \in \mathbb{R}^{2257\times 939}$ (actually, the number of groups is 1439), whose Moore-Penrose pseudoinverse is denoted by W^+ (Horn and Johnson., 1985). We further let $A := HW^+$ and $X := WZ$. Then the linear system can be converted into

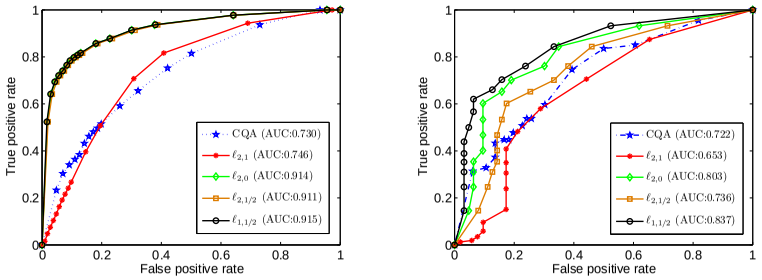
$$AX = B + \epsilon,$$

where A denotes expression profiles of TF complexes, and X represents connections between TF complexes and targets (Figure 9A).

Factor	GEO accession	Pubmed ID	Factor	GEO accession	Pubmed ID
Atf7ip	GSE26680	-	Rad21	GSE24029	21589869
Atrx	GSE22162	21029860	Rbbp5	GSE22934	21477851
Cdx2	GSE16375	19796622	Rcor1	GSE27844	22297846
Chd4	GSE27844	22297846	Rest	GSE26680	-
Ctcf	GSE11431	18555785	Rest	GSE27844	22297846
Ctcf	GSE28247	21685913	Rnf2	GSE13084	18974828
Ctr9	GSE20530	20434984	Rnf2	GSE26680	-
Dpy30	GSE26136	21335234	Rnf2	GSE34518	22305566
E2f1	GSE11431	18555785	Setdb1	GSE17642	19884257
Ep300	GSE11431	18555785	Smad1	GSE11431	18555785
Ep300	GSE28247	21685913	Smad2	GSE23581	21731500
Esrrb	GSE11431	18555785	Smarca4	GSE14344	19279218
Ezh2	GSE13084	18974828	Smc1a	GSE22562	20720539
Ezh2	GSE18776	20064375	Smc3	GSE22562	20720539
Jarid2	GSE18776	20064375	Sox2	GSE11431	18555785
Jarid2	GSE19365	20075857	Stat3	GSE11431	18555785
Kdm1a	GSE27844	22297846	Supt5h	GSE20530	20434984
Kdm5a	GSE18776	20064375	Suz12	GSE11431	18555785
Klf4	GSE11431	18555785	Suz12	GSE13084	18974828
Lmnb1	GSE28247	21685913	Suz12	GSE18776	20064375
Med1	GSE22562	20720539	Suz12	GSE19365	20075857
Med12	GSE22562	20720539	Taf1	GSE30959	21884934
Myc	GSE11431	18555785	Taf3	GSE30959	21884934
Mycn	GSE11431	18555785	Tbp	GSE30959	21884934
Nanog	GSE11431	18555785	Tbx3	GSE19219	20139965
Nipbl	GSE22562	20720539	Tcfcp2l1	GSE11431	18555785
Nr5a2	GSE19019	20096661	Tet1	GSE26832	21451524
Pou5f1	GSE11431	18555785	Wdr5	GSE22934	21477851
Pou5f1	GSE22934	21477851	Whsc2	GSE20530	20434984
Prdm14	GSE25409	21183938	Zfx	GSE11431	18555785

Table 2: ChIP-seq data for TF complex inference.

A literature-based golden standard (low-throughput golden standard) TF-target pair set from biological studies (Figure 9C), including 97 TF-target interactions between 23 TFs and 48 target genes, is downloaded from iScMiD (Integrated Stem Cell Molecular Interactions Database). Each TF-target pair in this golden standard data set has been verified by biological experiments. Another more comprehensive golden standard mESC network is constructed from high-throughput data (high-throughput golden standard) by ChIP-Array (Qin et al., 2011) using the methods described in Qin et al. (2014). It contains 40006 TF-target pairs between 13092 TFs or targets (Figure 9C). Basically, each TF-target pair in the network is evidenced by a cell-type specific binding site of the TF on the target's promoter and the expression change of the target in the perturbation experiment of the TF,



(a) Evaluation with high-throughput golden standard.

(b) Evaluation with literature-based low-throughput golden standard.

Figure 10: ROC curves and AUCs of the PGM-GSO on mESC gene regulatory network inference.

which is generally accepted as a true TF-target regulation. These two independent golden standards are both used to validate the accuracy of the inferred gene regulatory networks.

4.2.2 Numerical Results

We apply the PGM-GSO, starting from the initial matrix $X^0 := WZ^0$, to the gene regulatory network inference problem (Figure 9B), and compare with the CQ algorithm (CQA), which is shown in Wang et al. (2017) an efficient solver for gene regulatory network inference based on the group structure of TF complexes. The area under the curve (AUC) of a receiver operating characteristic (ROC) curve is widely recognized as an important index of the overall classification performance of an algorithm (see Fawcett, 2006). Here, we apply AUC to evaluate the performance of the PGM-GSO with four types of $\ell_{p,q}$ regularization, $(p,q) = (2,1), (2,0), (2,1/2)$ and $(1,1/2)$, as well as the CQA. A series of numbers of predictive TF complexes (or TFs), denoted by k, from 1 to 100 (that is, the sparsity level varies from about 0.07% to 7%) are tested. For each k and each pair of TF complex (or TF) i and target j, if the $X^{(k)}_{Gij}$ is non-zero, this TF complex (or TF) is regarded as a potential regulator of this target in this test. In biological sense, we only concern about whether the true TF is predicted, but not the weight of this TF. We also expect that the TF complexes (or TFs) which are predicted in a higher sparsity level should be more important than those that are only reported in a lower sparsity level. Thus, when calculating the AUC, a score Score_{ij} is applied as the predictor for TF i on target j:

$$\text{Score}_{ij} := \begin{cases} \max_k\{1/k\}, & X^{(k)}_{ij} \neq 0, \\ 0, & \text{otherwise.} \end{cases}$$

Both high-throughput and low-throughput golden standards are used to draw the ROC curves of the PGM-GSO with four types of $\ell_{p,q}$ regularization and the CQA in Figure 10

to compare their accuracy. When matched with the high-throughput golden standard, it is illustrated from Figure 10(a) that $\ell_{2,1/2}$, $\ell_{1,1/2}$ and $\ell_{2,0}$ perform almost the same (as indicated by the almost same AUC value), and significantly outperform $\ell_{2,1}$ and CQA. With the low-throughput golden standard, it is demonstrated from Figure 10(b) that $\ell_{1,1/2}$ is slightly better than $\ell_{2,1/2}$, $\ell_{2,0}$ and CQA, and these three regularizations perform much better than $\ell_{2,1}$. These results are basically consistent with the results from simulated data. Since the golden standards we use here are obtained from real biological experiments, which are well-accepted as true TF-target regulations, the higher AUC, the more biologically accurate the result gene regulatory network is. Thus, our results indicate that the $\ell_{p,1/2}$ and $\ell_{p,0}$ regularizations are applicable to gene regulatory network inference in biological researches that study higher organisms but generate transcriptome data for only a small number of samples, which facilitates biologists to analyze gene regulation in a system level.

Acknowledgments

We are grateful to the anonymous reviewers for their valuable suggestions and remarks which helped to improve the quality of the paper. We are also thankful to Professor Marc Teboulle for providing the reference Bolte et al. (2013) and the suggestion that the global convergence of the PGM-GSO can be established by using the so-called Kurdyka-Łojasiewicz theory, and Professor Junwen Wang for providing the gene expression data and biological golden standards for inferring gene regulatory network of mouse embryonic stem cell. Hu's work was supported in part by the National Natural Science Foundation of China (11601343), Natural Science Foundation of Guangdong (2016A030310038) and Foundation for Distinguished Young Talents in Higher Education of Guangdong (2015KQNCX145). Li's work was supported in part by the National Natural Science Foundation of China (11571308, 11371325, 91432302). Meng's work was supported in part by the National Natural Science Foundation of China (11671329, 31601066, 71601162). Qin's work was supported in part by the National Natural Science Foundation of China (41606143). Yang's work was supported in part by the Research Grants Council of Hong Kong (PolyU 152167/15E) and the National Natural Science Foundation of China (11431004).

References

H. Attouch, J. Bolte, P. Redont, and A. Soubeyran. Proximal alternating minimization and projection methods for nonconvex problems: An approach based on the Kurdyka-Łojasiewicz inequality. *Mathematics of Operations Research*, 35(2):438–457, 2010.

F. R. Bach. Consistency of the group Lasso and multiple kernel learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Cambridge, 1999.

P. J. Bickel, Y. Ritov, and A. B. Tsybakov. Simultaneous analysis of Lasso and Dantzig selector. *Annals of Statistics*, 37:1705–1732, 2009.

T. Blumensath and M. E. Davies. Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, 14:629–654, 2008.

J. Bolte, S. Sabach, and M. Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Mathematical Programming*, 146(1):459–494, 2013.

K. Bredies and D. A. Lorenz. Linear convergence of iterative soft-thresholding. *Journal of Fourier Analysis and Applications*, 14(5):813–837, 2008.

F. Bunea, A. Tsybakov, and M. Wegkamp. Sparsity oracle inequalities for the Lasso. *Electronic Journal of Statistics*, 1:169–194, 2007.

T. T. Cai and L. Wang. Orthogonal matching pursuit for sparse signal recovery with noise. *IEEE Transactions on Information Theory*, 57:4680–4688, 2011.

E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006a.

E. J. Candès and T. Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51:4203–4215, 2005.

E. J. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006b.

R. Chartrand. Exact reconstruction of sparse signals via nonconvex minimization. *IEEE Signal Processing Letters*, 14(10):707–710, 2007.

R. Chartrand and V. Staneva. Restricted isometry properties and nonconvex compressive sensing. *Inverse Problems*, 24:1–14, 2008.

S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Review*, 43:129–159, 2001.

X. Chen, F. Xu, and Y. Ye. Lower bound theory of nonzero entries in solutions of ℓ_2-ℓ_p minimization. *SIAM Journal on Scientific Computing*, 32(5):2832–2852, 2010.

X. Chen. Smoothing methods for nonsmooth, nonconvex minimization. *Mathematical Programming*, 134(1):71–99, 2012.

P. Combettes and V. Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, 4(4):1168–1200, 2005.

L. S. Correa-Cerro, Y. Piao, A. A. Sharov, A. Nishiyama, J. S. Cadet, H. Yu, L. V. Sharova, L. Xin, H. G. Hoang, M. Thomas, Y. Qian, D. B. Dudekula, E. Meyers, B. Y. Binder, G. Mowrer, U. Bassey, D. L. Longo, D. Schlessinger, and M. S. Ko. Generation of mouse ES cell lines engineered for the forced induction of transcription factors. *Scientific Reports*, 1:167, 2011.

I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57:1413–1457, 2004.

W. Deng, W. Yin, and Y. Zhang. Group sparse optimization by alternating direction method. Technical report, Rice University, 2011.

D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(8): 1289–1306, 2006a.

D. L. Donoho. High-dimensional centrally symmetric polytopes with neighborliness proportional to dimension. *Discrete and Computational Geometry*, 35(4):617–652, 2006b.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association*, 96(456):1348–1360, 2001.

T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

S. Foucart and M.-J. Lai. Sparsest solutions of underdetermined linear systems via ℓ_q-minimization for $0 < q \leq 1$. *Applied and Computational Harmonic Analysis*, 26(3): 395–407, 2009.

P. Frankel, G. Garrigos, and J. Peypouquet. Splitting methods with variable metric for Kurdyka–Lojasiewicz functions and general convergence rates. *Journal of Optimization Theory and Applications*, 165(3):874–900, 2015.

D. Ge, X. Jiang, and Y. Ye. A note on complexity of L_p minimization. *Mathematical Programming*, 129:285–299, 2011.

R. Ge. A filled function method for finding a global minimizer of a function of several variables. *Mathematical Programming*, 46(1):191–204, 1990.

E. G. Giannopoulou and O. Elemento. Inferring chromatin-bound protein complexes from genome-wide binding assays. *Genome Research*, 23(8):1295–1306, 2013.

P. Gong, C. Zhang, Z. Lu, J. Huang, and J. Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. *International Conference on Machine Learning (ICML)*, 2013.

E. T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for ℓ_1-minimization: Methodology and convergence. *SIAM Journal on Optimization*, 19(3):1107–1130, 2008.

D. Hnisz, B. J. Abraham, T. I. Lee, A. Lau, V. Saint-Andre, A. A. Sigova, H. A. Hoke, and R. A. Young. Super-enhancers in the control of cell identity and disease. *Cell*, 155(4): 934–947, 2013.

R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, New York, 1985.

Y. Hu, C. Li, and X. Yang. On convergence rates of linearized proximal algorithms for convex composite optimization with applications. *SIAM Journal on Optimization*, 26(2): 1207–1235, 2016.

X. Huang and X. Yang. A unified augmented Lagrangian approach to duality and exact penalization. *Mathematics of Operations Research*, 28(3):533–552, 2003.

M.-J. Lai and J. Wang. An unconstrained ℓ_q minimization with $0 < q \leq 1$ for sparse solution of underdetermined linear systems. *SIAM Journal on Optimization*, 21(1):82–101, 2011.

M.-J. Lai, Y. Xu, and W. Yin. Improved iteratively reweighted least squares for unconstrained smoothed ℓ_q minimization. *SIAM Journal on Numerical Analysis*, 51(2):927–957, 2013.

G. Li and T. K. Pong. Douglas–Rachford splitting for nonconvex optimization with application to nonconvex feasibility problems. *Mathematical Programming*, 159(1):1–31, 2015a.

G. Li and T. K. Pong. Global convergence of splitting methods for nonconvex composite optimization. *SIAM Journal on Optimization*, 25(4):2434–2460, 2015b.

G. Li and T. K. Pong. Calculus of the exponent of Kurdyka–Lojasiewicz inequality and its applications to linear convergence of first-order methods. *https://arXiv:1602.02915*, 2016.

Z. Lu. Iterative reweighted minimization methods for lp regularized unconstrained nonlinear programming. *Mathematical Programming*, 147(1):277–307, 2014.

Z.-Q. Luo, J. S. Pang, and D. Ralph. *Mathematical Programs with Equilibrium Constraints*. Cambridge University Press, Cambridge, 1996.

J. Mairal. Optimization with first-order surrogate functions. *International Conference on Machine Learning (ICML)*, 2013.

L. Meier, S. A. van de Geer, and P. Bühlmann. The group Lasso for logistic regression. *Journal of the Royal Statistical Society: Series B*, 70:53–71, 2008.

N. Meinshausen and B. Yu. Lasso-type recovery of sparse representations for high-dimensional data. *Annals of Statistics*, 37:246–270, 2009.

H. Mohimani, M. Babaie-Zadeh, and C. Jutten. A fast approach for overcomplete sparse decomposition based on smoothed l_0 norm. *IEEE Transactions on Signal Processing*, 57 (1):289–301, 2009.

B. S. Mordukhovich. *VariationalAnalysis and Generalized Differentiation I: Basic Theory*. Springer, Berlin, 2006.

B. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24(2):227–234, 1995.

D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2009.

Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

A. Nishiyama, L. Xin, A. A. Sharov, M. Thomas, G. Mowrer, E. Meyers, Y. Piao, S. Mehta, S. Yee, Y. Nakatake, C. Stagg, L. Sharova, L. S. Correa-Cerro, U. Bassey, H. Hoang, E. Kim, R. Tapnio, Y. Qian, D. Dudekula, M. Zalzman, M. Li, G. Falco, H. T. Yang, S. L. Lee, M. Monti, I. Stanghellini, M. N. Islam, R. Nagaraja, I. Goldberg, W. Wang, D. L. Longo, D. Schlessinger, and M. S. Ko. Uncovering early response of gene regulatory networks in ESCs by systematic induction of transcription factors. *Cell Stem Cell*, 5: 420–433, 2009.

A. Nishiyama, A. A. Sharov, Y. Piao, M. Amano, T. Amano, H. G. Hoang, B. Y. Binder, R. Tapnio, U. Bassey, J. N. Malinou, L. S. Correa-Cerro, H. Yu, L. Xin, E. Meyers, M. Zalzman, Y. Nakatake, C. Stagg, L. Sharova, Y. Qian, D. Dudekula, S. Sheer, J. S. Cadet, T. Hirata, H. T. Yang, I. Goldberg, M. K. Evans, D. L. Longo, D. Schlessinger, and M. S K.o. Systematic repression of transcription factors reveals limited patterns of gene expression changes in ES cells. *Scientific Reports*, 3:1390, 2013.

J. Qin, M. J. Li, P. Wang, M. Q. Zhang, and J. Wang. ChIP-Array: Combinatory analysis of ChIP-seq/chip and microarray gene expression data to discover direct/indirect targets of a transcription factor. *Nucleic Acids Research*, 39:W430–436, 2011.

J. Qin, Y. Hu, F. Xu, H. K. Yalamanchili, and J. Wang. Inferring gene regulatory networks by integrating ChIP-seq/chip and transcriptome data via LASSO-type regularization methods. *Methods*, 67(3):294–303, 2014.

M. Razaviyayn, M. Hong, and Z.-Q. Luo. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 23 (2):1126–1153, 2013.

R. T. Rockafellar and R. J.-B. Wets. *Variational Analysis*. Springer-Verlag, Berlin, 1998.

W. Rudin. *Principles of Mathematical Analysis*. McGraw-Hill, New York, 1976.

W. T. Short. Hyperbolic solution of the cubic equation. *National Mathematics Magazine*, 12(3):111–114, 1937.

S. Tao, D. Boley, and S. Zhang. Local linear convergence of ISTA and FISTA on the LASSO problem. *SIAM Journal on Optimization*, 26(1):313–336, 2016.

L. Tian, P. Xu, and Q. Gu. Forward backward greedy algorithms for multi-task learning with faster rates. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 735–744, 2016.

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society: Series B*, 58:267–288, 1994.

P. Tseng and S. Yun. A coordinate gradient descent method for nonsmooth separable minimization. *Mathematical Programming*, 117(1):387–423, 2009.

M. Usman, C. Prieto, T. Schaeffter, and P. G. Batchelor. *k-t* Group sparse: A method for accelerating dynamic MRI. *Magnetic Resonance in Medicine*, 66(4):1163–1176, 2011.

S. A. van de Geer and P. Bühlmann. On the conditions used to prove oracle results for the Lasso. *Electronic Journal of Statistics*, 3:1360–1392, 2009.

E. van den Berg and M. P. Friedlander. Probing the pareto frontier for basis pursuit solutions. *SIAM Journal on Scientific Computing*, 31(2):890–912, 2008.

E. van den Berg, M. Schmidt, M. P. Friedlander, and K. Murphy. Group sparsity via linear-time projection. Technical report, University of British Columbia, 2008.

J. Wang, Y. Hu, C. Li, and J.-C. Yao. Linear convergence of CQ algorithms and applications in gene regulatory network inference. *Inverse Problems*, to appear, 2017.

S. J. Wright, R. D. Nowak, and M. A T Figueiredo. Sparse reconstruction by separable approximation. *IEEE Transactions on Signal Processing*, 57(7):2479–2493, 2009.

L. Wu, Z. Sun, and D.-H. Li. A gradient based method for the l_2-$l_{1/2}$ minimization and application to compressive sensing. *Pacific Journal of Optimization*, 10(2):401–414, 2014.

L. Xiao and T. Zhang. A proximal-gradient homotopy method for the sparse least-squares problem. *SIAM Journal on Optimization*, 23(2):1062–1091, 2013.

D. Xie, A. P. Boyle, L. Wu, J. Zhai, T. Kawli, and M. Snyder. Dynamic trans-acting factor colocalization in human cells. *Cell*, 155(3):713–724, 2013.

Y. Xu and W. Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on Imaging Sciences*, 6(3):1758–1789, 2013.

Z. Xu, X. Chang, F. Xu, and H. Zhang. $L_{1/2}$ regularization: A thresholding representation theory and a fast solver. *IEEE Transactions on Neural Networks and Learning Systems*, 23:1013–1027, 2012.

H. Yang, Z. Xu, I. King, and M. R. Lyu. Online learning for group Lasso. *International Conference on Machine Learning (ICML)*, 2010.

J. Yang and Y. Zhang. Alternating direction algorithms for ℓ_1-problems in compressive sensing. *SIAM Journal on Scientific Computing*, 33(1):250–278, 2011.

X. Yang and X. Huang. A nonlinear Lagrangian approach to constrained optimization problems. *SIAM Journal on Optimization*, 11(4):1119–1144, 2001.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of The Royal Statistical Society: Series B*, 68:49–67, 2006.

J. Zeng, S. Lin, and Z. Xu. Sparse regularization: Convergence of iterative jumping thresholding algorithm. *https://arxiv.org/abs/1402.5744*, 2015.

T. Zhang. Adaptive forward-backward greedy algorithm for learning sparse representations. *IEEE Transactions on Information Theory*, 57(7):4689–4708, 2011.

T. Zhang. Some sharp performance bounds for least squares regression with L_1 regularization. *Annals of Statistics*, 37:2109–2144, 2009.

T. Zhang. Analysis of multi-stage convex relaxation for sparse regularization. *Journal of Machine Learning Research*, 11:1081–1107, 2010.

Journal of Machine Learning Research 18 (2017) 1-32 Submitted 9/16; Revised 2/17; Published 4/17

Preference-based Teaching

Ziyuan Gao GAO257@CS.UREGINA.CA
Department of Computer Science, University of Regina

Christoph Ries CHRISTOPH.RIES@RUB.DE
Department of Mathematics, Ruhr-University Bochum

Hans U. Simon HANS.SIMON@RUB.DE
Department of Mathematics, Ruhr-University Bochum

Sandra Zilles ZILLES@CS.UREGINA.CA
Department of Computer Science, University of Regina

Editor: Manfred Warmuth

Abstract

We introduce a new model of teaching named "preference-based teaching" and a corresponding complexity parameter—the preference-based teaching dimension (PBTD)—representing the worst-case number of examples needed to teach any concept in a given concept class. Although the PBTD coincides with the well-known recursive teaching dimension (RTD) on finite classes, it is radically different on infinite ones: the RTD becomes infinite already for trivial infinite classes (such as half-intervals) whereas the PBTD evaluates to reasonably small values for a wide collection of infinite classes including classes consisting of so-called closed sets w.r.t. a given closure operator, including various classes related to linear sets over $\mathbb{N}_0$ (whose RTD had been studied quite recently) and including the class of Euclidean half-spaces. On top of presenting these concrete results, we provide the reader with a theoretical framework (of a combinatorial flavor) which helps to derive bounds on the PBTD.

Keywords: teaching dimension, preference relation, recursive teaching dimension, learning half-spaces, linear sets

1. Introduction

The classical model of teaching (Shinohara and Miyano, 1991; Goldman and Kearns, 1995) formulates the following interaction protocol between a teacher and a student:

- Both of them agree on a "classification-rule system", formally given by a concept class $\mathcal{L}$.
- In order to teach a specific concept $L \in \mathcal{L}$, the teacher presents to the student a *teaching set*, i.e., a set T of labeled examples so that L is the only concept in $\mathcal{L}$ that is consistent with T.
- The student determines L as the unique concept in $\mathcal{L}$ that is consistent with T.

Goldman and Mathias (1996) pointed out that this model of teaching is not powerful enough, since the teacher is required to make *any* consistent learner successful. A challenge is to model powerful teacher/student interactions without enabling unfair "coding tricks". Intuitively, the term "coding trick" refers to any form of undesirable collusion between teacher and learner, which would reduce the learning process to a mere decoding of a code the teacher sent to the learner. There is no generally accepted definition of what constitutes a coding trick, in part because teaching an exact learner could always be considered coding to some extent: the teacher presents a set of examples which the learner "decodes" into a concept.

In this paper, we adopt the notion of "valid teacher/learner pair" introduced by Goldman and Mathias (1996). They consider their model to be intuitively free of coding tricks while it provably allows for a much broader class of interaction protocols than the original teaching model. In particular, teaching may thus become more efficient in terms of the number of examples in the teaching sets. Further definitions of how to avoid unfair coding tricks have been suggested (Zilles et al., 2011), but they were less stringent than the one proposed by Goldman and Mathias. The latter simply requests that, if the learner hypothesizes concept L upon seeing a sample set S of labeled examples, then the learner will still hypothesize L when presented with any sample set $S \cup S'$, where S' contains only examples labeled consistently with L. A coding trick would then be any form of exchange between the teacher and the learner that does not satisfy this definition of validity.

The model of recursive teaching (Zilles et al., 2011; Mazadi et al., 2014), which is free of coding tricks according to the Goldman-Mathias definition, has recently gained attention because its complexity parameter, the recursive teaching dimension (RTD), has shown relations to the VC-dimension and to sample compression (Chen et al., 2016; Doliwa et al., 2014; Moran et al., 2015; Simon and Zilles, 2015), when focusing on finite concept classes. Below though we will give examples of rather simple infinite concept classes with infinite RTD, suggesting that the RTD is inadequate for addressing the complexity of teaching infinite classes.

In this paper, we introduce a model called *preference-based teaching*, in which the teacher and the student do not only agree on a classification-rule system $\mathcal{L}$ but also on a preference relation (a strict partial order) imposed on $\mathcal{L}$. If the labeled examples presented by the teacher allow for several consistent explanations (= consistent concepts) in $\mathcal{L}$, the student will choose a concept $L \in \mathcal{L}$ that she prefers most. This gives more flexibility to the teacher than the classical model: the set of labeled examples need not distinguish a target concept L from any other concept in $\mathcal{L}$ but only from those concepts L' over which L is not preferred.[1] At the same time, preference-based teaching yields valid teacher/learner pairs according to Goldman and Mathias's definition. We will show that the new model, despite avoiding coding tricks, is quite powerful. Moreover, as we will see in the course of the paper, it often allows for a very natural design of teaching sets.

Assume teacher and student choose a preference relation that minimizes the worst-case number M of examples required for teaching any concept in the class $\mathcal{L}$. This number M is then called the preference-based teaching dimension (PBTD) of $\mathcal{L}$. In particular, we will show the following:

(i) Recursive teaching is a special case of preference-based teaching where the preference relation satisfies a so-called "finite-depth condition". It is precisely this additional condition that renders recursive teaching useless for many natural and apparently simple infinite concept classes. Preference-based teaching successfully addresses these shortcomings of recursive teaching, see Section 3. For finite classes, PBTD and RTD are equal.

(ii) A wide collection of geometric and algebraic concept classes with infinite RTD can be taught very efficiently, i.e., with low PBTD. To establish such results, we show in Section 4 that spanning sets can be used as preference-based teaching sets with positive examples only — a result that is very simple to obtain but quite useful.

1. Such a preference relation can be thought of as a kind of bias in learning: the student is "biased" towards concepts that are preferred over others, and the teacher, knowing the student's bias, selects teaching sets accordingly.

©2017 Ziyuan Gao, Christoph Ries, Hans U. Simon, and Sandra Zilles.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-460.html.

(iii) In the preference-based model, linear sets over $\mathbb{N}_0$ with origin 0 and at most k generators can be taught with k positive examples, while recursive teaching with a bounded number of positive examples was previously shown to be impossible and it is unknown whether recursive teaching with a bounded number of positive and negative examples is possible for $k \geq 4$. We also give some almost matching upper and lower bounds on the PBTD for other classes of linear sets, see Section 6.

(iv) The PBTD of halfspaces in $\mathbb{R}^d$ is upper-bounded by 6, independent of the dimensionality d (see Section 7), while its RTD is infinite.

(v) We give full characterizations of concept classes that can be taught with only one example (or with only one example, which is positive) in the preference-based model (see Section 8).

Based on our results and the naturalness of the teaching sets and preference relations used in their proofs, we claim that preference-based teaching is far more suitable to the study of infinite concept classes than recursive teaching.

Parts of this paper were published in a previous conference version (Gao et al., 2016).

2. Basic Definitions and Facts

$\mathbb{N}_0$ denotes the set of all non-negative integers and $\mathbb{N}$ denotes the set of all positive integers. A *concept class* $\mathcal{L}$ is a family of subsets over a universe $\mathcal{X}$, i.e., $\mathcal{L} \subseteq 2^{\mathcal{X}}$ where $2^{\mathcal{X}}$ denotes the powerset of $\mathcal{X}$. The elements of $\mathcal{L}$ are called *concepts*. A *labeled example* is an element of $\mathcal{X} \times \{-,+\}$. We slightly deviate from this notation in Section 7, where our treatment of halfspaces makes it more convenient to use $\{-1,1\}$ instead of $\{-,+\}$, and in Section 8, where we perform Boolean operations on the labels and therefore use $\{0,1\}$ instead of $\{-,+\}$. Elements of $\mathcal{X}$ are called *examples*. Suppose that T is a set of labeled examples. Let $T^+ = \{x \in \mathcal{X} : (x,+) \in T\}$ and $T^- = \{x \in \mathcal{X} : (x,-) \in T\}$. A set $L \subseteq \mathcal{X}$ is *consistent with* T if it includes all examples in T that are labeled "+" and excludes all examples in T that are labeled "−", i.e, if $T^+ \subseteq L$ and $T^- \cap L = \emptyset$. A set of labeled examples that is consistent with L but not with L' is said to *distinguish L from L'*. The classical model of teaching is then defined as follows.

Definition 1 (Shinohara and Miyano (1991); Goldman and Kearns (1995)) *A* teaching set *for a concept $L \in \mathcal{L}$ w.r.t. $\mathcal{L}$ is a set T of labeled examples such that L is the only concept in $\mathcal{L}$ that is consistent with T, i.e., T distinguishes L from any other concept in $\mathcal{L}$. Define* $\mathrm{TD}(L,\mathcal{L}) = \inf\{|T| : T$ *is a teaching set for L w.r.t. $\mathcal{L}\}$. i.e., $\mathrm{TD}(L,\mathcal{L})$ is the smallest possible size of a teaching set for L w.r.t. $\mathcal{L}$. If L has no finite teaching set w.r.t. $\mathcal{L}$, then $\mathrm{TD}(L,\mathcal{L}) = \infty$. The number $\mathrm{TD}(\mathcal{L}) = \sup_{L\in\mathcal{L}} \mathrm{TD}(L,\mathcal{L}) \in \mathbb{N}_0 \cup \{\infty\}$ is called the* teaching dimension of $\mathcal{L}$.

For technical reasons, we will occasionally deal with the number $\mathrm{TD}_{min}(\mathcal{L}) = \inf_{L\in\mathcal{L}} \mathrm{TD}(L, \mathcal{L})$, i.e., the number of examples needed to teach the concept from $\mathcal{L}$ that is easiest to teach.

In this paper, we will examine a teaching model in which the teacher and the student do not only agree on a classification-rule system $\mathcal{L}$ but also on a preference relation, denoted as $\prec$, imposed on $\mathcal{L}$. We assume that $\prec$ is a *strict partial order* on $\mathcal{L}$, i.e., $\prec$ is asymmetric and transitive. The partial order that makes every pair $L \neq L' \in \mathcal{L}$ incomparable is denoted by $\prec_\emptyset$. For every $L \in \mathcal{L}$, let

$$\mathcal{L}_{\prec L} = \{L' \in \mathcal{L} : L' \prec L\}$$

be the set of concepts over which L is strictly preferred. Note that $\mathcal{L}_{\prec_\emptyset L} = \emptyset$ for every $L \in \mathcal{L}$.

As already noted above, a teaching set T of L w.r.t. $\mathcal{L}$ distinguishes L from any other concept in $\mathcal{L}$. If a preference relation comes into play, then T will be exempted from the obligation to distinguish L from the concepts in $\mathcal{L}_{\prec L}$ because L is strictly preferred over them anyway.

Definition 2 *A* teaching set for $L \subseteq X$ w.r.t. $(\mathcal{L},\prec)$ *is defined as a teaching set for L w.r.t. $\mathcal{L} \setminus \mathcal{L}_{\prec L}$. Furthermore define*

$$\mathrm{PBTD}(L,\mathcal{L},\prec) = \inf\{|T| : T \text{ is a teaching set for } L \text{ w.r.t. } (\mathcal{L},\prec)\} \in \mathbb{N}_0 \cup \{\infty\} \ .$$

The number $\mathrm{PBTD}(\mathcal{L},\prec) = \sup_{L\in\mathcal{L}} \mathrm{PBTD}(L,\mathcal{L},\prec) \in \mathbb{N}_0 \cup \{\infty\}$ *is called the* teaching dimension of $(\mathcal{L},\prec)$.

Definition 2 implies that

$$\mathrm{PBTD}(L,\mathcal{L},\prec) = \mathrm{TD}(L, \mathcal{L} \setminus \mathcal{L}_{\prec L}) \ . \tag{1}$$

Let $L \mapsto T(L)$ be a mapping that assigns a teaching set for L w.r.t. $(\mathcal{L},\prec)$ to every $L \in \mathcal{L}$. It is obvious from Definition 2 that T must be injective, i.e., $T(L) \neq T(L')$ if L and L' are distinct concepts from $\mathcal{L}$. The classical model of teaching is obtained from the model described in Definition 2 when we plug in the empty preference relation $\prec_\emptyset$ for $\prec$. In particular, $\mathrm{PBTD}(\mathcal{L},\prec_\emptyset) = \mathrm{TD}(\mathcal{L})$.

We are interested in finding the partial order that is optimal for the purpose of teaching and we aim at determining the corresponding teaching dimension. This motivates the following notion:

Definition 3 *The* preference-based teaching dimension of $\mathcal{L}$ *is given by*

$$\mathrm{PBTD}(\mathcal{L}) = \inf\{\mathrm{PBTD}(\mathcal{L},\prec) : \prec \text{ is a strict partial order on } \mathcal{L}\} \ .$$

A relation R' on $\mathcal{L}$ is said to be an *extension of a relation R* if $R \subseteq R'$. The *order-extension principle* states that any partial order has a linear extension (Jech, 1973). The following result (whose second assertion follows from the first one in combination with the order-extension principle) is pretty obvious:

Lemma 4 *1. Suppose that $\prec'$ extends $\prec$. If T is a teaching set for L w.r.t. $(\mathcal{L},\prec)$, then T is a teaching set for L w.r.t. $(\mathcal{L},\prec')$. Moreover* $\mathrm{PBTD}(\mathcal{L},\prec') \leq \mathrm{PBTD}(\mathcal{L},\prec)$.

2. $\mathrm{PBTD}(\mathcal{L}) = \inf\{\mathrm{PBTD}(\mathcal{L},\prec) : \prec$ *is a strict linear order on* $\mathcal{L}\}$.

Recall that Goldman and Mathias (1996) suggested to avoid coding tricks by requesting that any superset S of a teaching set for a concept L remains a teaching set, if S is consistent with L. This property is obviously satisfied in preference-based teaching. A preference-based teaching set needs to distinguish a concept L from all concepts in $\mathcal{L}$ that are preferred over L. Adding more labeled examples from L to such a teaching set will still result in a set distinguishing L from all concepts in $\mathcal{L}$ that are preferred over L.

Preference-based teaching with positive examples only. Suppose that $\mathcal{L}$ contains two concepts L, L' such that $L \subset L'$. In the classical teaching model, any teaching set for L w.r.t. $\mathcal{L}$ has to employ a negative example in order to distinguish L from L'. Symmetrically, any teaching set for L' w.r.t. $\mathcal{L}$ has to employ a positive example. Thus classical teaching cannot be performed with one type of examples only unless $\mathcal{L}$ is an antichain w.r.t. inclusion. As for preference-based teaching, the restriction to one type of examples is much less severe, as our results below will show.

A teaching set T for $L \in \mathcal{L}$ w.r.t. $(\mathcal{L}, \prec)$ is said to be *positive* if it does not make use of negatively labeled examples, i.e., if $T^- = \emptyset$. In the sequel, we will occasionally identify a positive teaching set T with T^+. A positive teaching set for L w.r.t. $(\mathcal{L}, \prec)$ can clearly not distinguish L from a proper superset of L in $\mathcal{L}$. Thus, the following holds:

Lemma 5 *Suppose that $L \mapsto T^+(L)$ maps each $L \in \mathcal{L}$ to a positive teaching set for L w.r.t. $(\mathcal{L}, \prec)$. Then $\prec$ must be an extension of $\supset$ (so that proper subsets of a set L are strictly preferred over L) and, for every $L \in \mathcal{L}$, the set $T^+(L)$ must distinguish L from every proper subset of L in $\mathcal{L}$.*

Define

$$\mathrm{PBTD}^+(L, \mathcal{L}, \prec) = \inf\{|T| : T \text{ is a positive teaching set for } L \text{ w.r.t. } (\mathcal{L}, \prec)\} \ . \tag{2}$$

The number $\mathrm{PBTD}^+(\mathcal{L}, \prec) = \sup_{L \in \mathcal{L}} \mathrm{PBTD}^+(L, \mathcal{L}, \prec)$ (possibly ∞) is called the *positive teaching dimension of* $(\mathcal{L}, \prec)$. The *positive preference-based teaching dimension of* $\mathcal{L}$ is then given by

$$\mathrm{PBTD}^+(\mathcal{L}) = \inf\{\mathrm{PBTD}^+(\mathcal{L}, \prec) : \prec \text{ is a strict partial order on } \mathcal{L}\} \ . \tag{3}$$

Monotonicity. A complexity measure K that assigns a number $K(\mathcal{L}) \in \mathbb{N}_0$ to a concept class $\mathcal{L}$ is said to be *monotonic* if $\mathcal{L}' \subseteq \mathcal{L}$ implies that $K(\mathcal{L}') \leq K(\mathcal{L})$. It is well known (and trivial to see) that TD is monotonic. It is fairly obvious that PBTD is monotonic, too:

Lemma 6 PBTD *and* PBTD^+ *are monotonic.*

As an application of monotonicity, we show the following result:

Lemma 7 *For every finite subclass $\mathcal{L}'$ of $\mathcal{L}$, we have* $\mathrm{PBTD}(\mathcal{L}) \geq \mathrm{PBTD}(\mathcal{L}') \geq \mathrm{TD}_{min}(\mathcal{L}')$.

Proof The first inequality holds because PBTD is monotonic. The second inequality follows from the fact that a finite partially ordered set must contain a minimal element. Thus, for any fixed choice of $\prec$, $\mathcal{L}'$ must contain a concept L' such that $\mathcal{L}'_{\prec L'} = \emptyset$. Hence,

$$\mathrm{PBTD}(\mathcal{L}', \prec) \geq \mathrm{PBTD}(L', \mathcal{L}', \prec) \overset{(1)}{=} \mathrm{TD}(L', \mathcal{L}' \setminus \mathcal{L}'_{\prec L'}) = \mathrm{TD}(L', \mathcal{L}') \geq \mathrm{TD}_{min}(\mathcal{L}') \ .$$

Since this holds for any choice of $\prec$, we get $\mathrm{PBTD}(\mathcal{L}') \geq \mathrm{TD}_{min}(\mathcal{L}')$, as desired. ■

3. Preference-based versus Recursive Teaching

The preference-based teaching dimension is a relative of the recursive teaching dimension. In fact, both notions coincide on finite classes, as we will see shortly. We first recall the definitions of the recursive teaching dimension and of some related notions (Zilles et al., 2011; Mazadi et al., 2014).

A *teaching sequence for* $\mathcal{L}$ is a sequence of the form $\mathcal{S} = (\mathcal{L}_i, d_i)_{i \geq 1}$ where $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3, \ldots$ form a partition of $\mathcal{L}$ into non-empty sub-classes and, for every $i \geq 1$, we have that

$$d_i = \sup_{L \in \mathcal{L}_i} \mathrm{TD}\left(L, \mathcal{L} \setminus \cup_{j=1}^{i-1} \mathcal{L}_j\right) \ . \tag{4}$$

If, for every $i \geq 1$, d_i is the supremum over all $L \in \mathcal{L}_i$ of the smallest size of a *positive teaching set* for L w.r.t. $\cup_{j \geq i} \mathcal{L}_j$ (and $d_i = \infty$ if some $L \in \mathcal{L}_i$ does not have a positive teaching set w.r.t. $\cup_{j \geq i} \mathcal{L}_j$), then $\mathcal{S}$ is said to be a *positive teaching sequence for* $\mathcal{L}$. The *order* of a teaching sequence or a positive teaching sequence $\mathcal{S}$ (possibly ∞) is defined as $\mathrm{ord}(\mathcal{S}) = \sup_{i \geq 1} d_i$. The *recursive teaching dimension of* $\mathcal{L}$ (possibly ∞) is defined as the order of the teaching sequence of lowest order for $\mathcal{L}$. More formally, $\mathrm{RTD}(\mathcal{L}) = \inf_{\mathcal{S}} \mathrm{ord}(\mathcal{S})$ where $\mathcal{S}$ ranges over all teaching sequences for $\mathcal{L}$. Similarly, $\mathrm{RTD}^+(\mathcal{L}) = \inf_{\mathcal{S}} \mathrm{ord}(\mathcal{S})$, where $\mathcal{S}$ ranges over all positive teaching sequences for $\mathcal{L}$. Note that the following holds for every $\mathcal{L}' \subseteq \mathcal{L}$ and for every teaching sequence $\mathcal{S} = (\mathcal{L}_i, d_i)_{i \geq 1}$ for $\mathcal{L}'$ such that $\mathrm{ord}(\mathcal{S}) = \mathrm{RTD}(\mathcal{L}')$:

$$\mathrm{RTD}(\mathcal{L}) \geq \mathrm{RTD}(\mathcal{L}') = \mathrm{ord}(\mathcal{S}) \geq d_1 = \sup_{L \in \mathcal{L}_1} \mathrm{TD}(L, \mathcal{L}') \geq \mathrm{TD}_{min}(\mathcal{L}') \ . \tag{5}$$

Note an important difference between PBTD and RTD: while $\mathrm{RTD}(\mathcal{L}) \geq \mathrm{TD}_{min}(\mathcal{L}')$ for *all* $\mathcal{L}' \subseteq \mathcal{L}$, in general the same holds for PBTD only when restricted to finite $\mathcal{L}'$, cf. Lemma 7. This difference will become evident in the proof of Lemma 10.

The *depth* of $L \in \mathcal{L}$ w.r.t. a strict partial order imposed on $\mathcal{L}$ is defined as the length of the longest chain in $(\mathcal{L}, \prec)$ that ends with the $\prec$-maximal element L (resp. as ∞ if there is no bound on the length of these chains). The recursive teaching dimension is related to the preference-based teaching dimension as follows:

Lemma 8 $\mathrm{RTD}(\mathcal{L}) = \inf_{\prec} \mathrm{PBTD}(\mathcal{L}, \prec)$ *and* $\mathrm{RTD}^+(\mathcal{L}) = \inf_{\prec} \mathrm{PBTD}^+(\mathcal{L}, \prec)$ *where $\prec$ ranges over all strict partial orders on $\mathcal{L}$ that satisfy the following "finite-depth condition": every $L \in \mathcal{L}$ has a finite depth w.r.t. $\prec$.*

The following is an immediate consequence of Lemma 8 and the trivial observation that the finite-depth condition is always satisfied if $\mathcal{L}$ is finite:

Corollary 9 $\mathrm{PBTD}(\mathcal{L}) \leq \mathrm{RTD}(\mathcal{L})$*, with equality if $\mathcal{L}$ is finite.*

While $\mathrm{PBTD}(\mathcal{L})$ and $\mathrm{RTD}(\mathcal{L})$ refer to the same finite number when $\mathcal{L}$ is finite, there are classes for which the RTD is infinity and yet the PBTD is finite, as Lemma 10 will show.

Lemma 10 *There exists an infinite class $\mathcal{L}_\infty$ of VC-dimension 1 such that* $\mathrm{PBTD}^+(\mathcal{L}_\infty) = 1$ *and* $\mathrm{RTD}(\mathcal{L}_\infty) = \infty$.

Proof Let $\mathcal{L}_\infty$ be the family of closed half-intervals over $[0,1)$, i.e., $\mathcal{L}_\infty = \{[0,a] : 0 \leq a < 1\}$. We first prove that $\mathrm{PBTD}^+(\mathcal{L}_\infty) = 1$. Consider the preference relation given by $[0,b] \prec [0,a]$ iff $a < b$. Then, for each $0 \leq a < 1$, we have

$$\mathrm{PBTD}([0,a], \mathcal{L}_\infty, \prec) \stackrel{(1)}{=} \mathrm{TD}([0,a], \{[0,b] : \ 0 \leq b \leq a\}) = 1$$

because the single example $(a,+)$ suffices for distinguishing $[0,a]$ from any interval $[0,b]$ with $b < a$.

It was observed by Moran et al. (2015) already that $\mathrm{RTD}(\mathcal{L}_\infty) = \infty$ because every teaching set for some $[0,a]$ must contain an infinite sequence of distinct reals that converges from above to a. Thus, using Equation (5) with $\mathcal{L}' = \mathcal{L}$, we have $\mathrm{RTD}(\mathcal{L}_\infty) \geq \mathrm{TD}_{min}(\mathcal{L}_\infty) = \infty$. ■

Remark 11 *For every $k \geq 1$, there exists an infinite class $\mathcal{L}_k$ such that $\mathrm{PBTD}^+(\mathcal{L}_k) = 1$ and $\mathrm{RTD}(\mathcal{L}_k) = k$; see (Gao et al., 2017, Lemma 6).*

4. Preference-based Teaching with Positive Examples Only

The main purpose of this section is to relate positive preference-based teaching to "spanning sets" and "closure operators", which are well-studied concepts in the computational learning theory literature. Let $\mathcal{L}$ be a concept class over the universe $\mathcal{X}$. We say that $S \subseteq \mathcal{X}$ is a *spanning set* of $L \in \mathcal{L}$ w.r.t. $\mathcal{L}$ if $S \subseteq L$ and any set in $\mathcal{L}$ that contains S must contain L as well.[2] In other words, L is the unique smallest concept in $\mathcal{L}$ that contains S. We say that $S \subseteq \mathcal{X}$ is a *weak spanning set* of $L \in \mathcal{L}$ w.r.t. $\mathcal{L}$ if $S \subseteq L$ and S is not contained in any proper subset of L in $\mathcal{L}$.[3] We denote by $I(\mathcal{L})$ (resp. $I'(\mathcal{L})$) the smallest number k such that every concept $L \in \mathcal{L}$ has a spanning set (resp. a weak spanning set) w.r.t. $\mathcal{L}$ of size at most k. Note that S is a spanning set of L w.r.t. $\mathcal{L}$ iff S distinguishes L from all concepts in $\mathcal{L}$ except for supersets of L, i.e., iff S is a positive teaching set for L w.r.t. $(\mathcal{L}, \supset)$. Similarly, S is a weak spanning set of L w.r.t. $\mathcal{L}$ iff S distinguishes L from all its proper subsets in $\mathcal{L}$ (which is necessarily the case when S is a positive teaching set). These observations can be summarized as follows:

$$I'(\mathcal{L}) \leq \mathrm{PBTD}^+(\mathcal{L}) \leq \mathrm{PBTD}^+(\mathcal{L}, \supset) \leq I(\mathcal{L}) \ . \tag{6}$$

The last two inequalities are straightforward. The inequality $I'(\mathcal{L}) \leq \mathrm{PBTD}^+(\mathcal{L})$ follows from Lemma 5, which implies that no concept L can have a preference-based teaching set T smaller than its smallest weak spanning set. Such a set T would be consistent with some proper subset of L, which is impossible by Lemma 5.

Suppose $\mathcal{L}$ is intersection-closed. Then $\cap_{L \in \mathcal{L}: S \subseteq L} L$ is the unique smallest concept in $\mathcal{L}$ containing S. If $S \subseteq L_0$ is a weak spanning set of $L_0 \in \mathcal{L}$, then $\cap_{L \in \mathcal{L}: S \subseteq L} L = L_0$ because, on the one hand, $\cap_{L \in \mathcal{L}: S \subseteq L} L \subseteq L_0$ and, on the other hand, no proper subset of L_0 in $\mathcal{L}$ contains S. Thus the distinction between spanning sets and weak spanning sets is blurred for intersection-closed classes:

2. This generalizes the classical definition of a spanning set (Helmbold et al., 1990), which is given w.r.t. intersection-closed classes only.
3. Weak spanning sets have been used in the field of recursion-theoretic inductive inference under the name "tell-tale sets" (Angluin, 1980).

Lemma 12 *Suppose that $\mathcal{L}$ is intersection-closed. Then $I'(\mathcal{L}) = \mathrm{PBTD}^+(\mathcal{L}) = I(\mathcal{L})$.*

Example 1 *Let $\mathcal{R}_d$ denote the class of d-dimensional axis-parallel hyper-rectangles (= d-dimensional boxes). This class is intersection-closed and clearly $I(\mathcal{R}_d) = 2$. Thus $\mathrm{PBTD}^+(\mathcal{R}_d) = 2$.*

A mapping $\mathrm{cl} : 2^{\mathcal{X}} \to 2^{\mathcal{X}}$ is said to be a *closure operator* on the universe $\mathcal{X}$ if the following conditions hold for all sets $A, B \subseteq \mathcal{X}$:

$$A \subseteq B \Rightarrow \mathrm{cl}(A) \subseteq \mathrm{cl}(B) \ \text{ and } \ A \subseteq \mathrm{cl}(A) = \mathrm{cl}(\mathrm{cl}(A)) \ .$$

The following notions refer to an arbitrary but fixed closure operator. The set $\mathrm{cl}(A)$ is called the *closure* of A. A set C is said to be *closed* if $\mathrm{cl}(C) = C$. It follows that precisely the sets $\mathrm{cl}(A)$ with $A \subseteq \mathcal{X}$ are closed. With this notation, we observe the following lemma.

Lemma 13 *Let $\mathcal{C}$ be the set of all closed subsets of $\mathcal{X}$ under some closure operator* cl*, and let $L \in \mathcal{C}$. If $L = \mathrm{cl}(S)$, then S is a spanning set of L w.r.t. $\mathcal{C}$.*

Proof Suppose $L' \in \mathcal{C}$ and $S \subseteq L'$. Then $L = \mathrm{cl}(S) \subseteq \mathrm{cl}(L') = L'$. ■

For every closed set $L \in \mathcal{L}$, let $s_{cl}(L)$ denote the size (possibly ∞) of the smallest set $S \subseteq \mathcal{X}$ such that $\mathrm{cl}(S) = L$. With this notation, we get the following (trivial but useful) result:

Theorem 14 *Given a closure operator, let $\mathcal{C}[m]$ be the class of all closed subsets $C \subseteq \mathcal{X}$ with $s_{cl}(C) \leq m$. Then $\mathrm{PBTD}^+(\mathcal{C}[m]) \leq \mathrm{PBTD}^+(\mathcal{C}[m], \supset) \leq m$. Moreover, this holds with equality provided that $\mathcal{C}[m] \setminus \mathcal{C}[m-1] \neq \emptyset$.*

Proof The inequality $\mathrm{PBTD}^+(\mathcal{C}[m], \supset) \leq m$ follows directly from Equation (6) and Lemma 13. Pick a concept $C_0 \in \mathcal{C}[m]$ such that $s_{cl}(C_0) = m$. Then any subset S of C_0 of size less than m spans only a proper subset of C_0, i.e., $\mathrm{cl}(S) \subset C_0$. Thus S does not distinguish C_0 from $\mathrm{cl}(S)$. However, by Lemma 5, any preference-based learner must strictly prefer $\mathrm{cl}(S)$ over C_0. It follows that there is no positive teaching set of size less than m for C_0 w.r.t. $\mathcal{C}[m]$. ■

Many natural classes can be cast as classes of the form $\mathcal{C}[m]$ by choosing the universe and the closure operator appropriately; the following examples illustrate the usefulness of Theorem 14 in that regard.

Example 2 *Let*

$$\mathrm{LINSET}_k = \{\langle G \rangle : (G \subset \mathbb{N}) \wedge (1 \leq |G| \leq k)\}$$

where $\langle G \rangle = \left\{ \sum_{g \in G} a(g) g : a(g) \in \mathbb{N}_0 \right\}$. In other words, LINSET_k is the set of all non-empty linear subsets of $\mathbb{N}_0$ that are generated by at most k generators. Note that the mapping $G \mapsto \langle G \rangle$ is a closure operator over the universe $\mathbb{N}_0$. Since obviously $\mathrm{LINSET}_k \setminus \mathrm{LINSET}_{k-1} \neq \emptyset$, we obtain $\mathrm{PBTD}^+(\mathrm{LINSET}_k) = k$.

Example 3 *Let $\mathcal{X} = \mathbb{R}^2$ and let $\mathcal{C}_k$ be the class of convex polygons with at most k vertices. Defining $\mathrm{cl}(S)$ to be the convex closure of S, we obtain $\mathcal{C}[k] = \mathcal{C}_k$ and thus $\mathrm{PBTD}^+(\mathcal{C}_k) = k$.*

Example 4 *Let $\mathcal{X} = \mathbb{R}^n$ and let $\mathcal{C}_k$ be the class of polyhedral cones that can be generated by k (or less) vectors in $\mathbb{R}^n$. If we take $\mathrm{cl}(S)$ to be the conic closure of $S \subseteq \mathbb{R}^n$, then $\mathcal{C}[k] = \mathcal{C}_k$ and thus $\mathrm{PBTD}^+(\mathcal{C}_k) = k$.*

5. A Convenient Technique for Proving Upper Bounds

In this section, we give an alternative definition of the preference-based teaching dimension using the notion of an "admissible mapping". Given a concept class $\mathcal{L}$ over a universe $\mathcal{X}$, let T be a mapping $L \mapsto T(L) \subseteq \mathcal{X} \times \{-,+\}$ that assigns a set $T(L)$ of labeled examples to every set $L \in \mathcal{L}$ such that the labels in $T(L)$ are consistent with L. The *order* of T, denoted as $\mathrm{ord}(T)$, is defined as $\sup_{L\in\mathcal{L}} |T(L)| \in \mathbb{N} \cup \{\infty\}$. Define the mappings T^+ and T^- by setting $T^+(L) = \{x : (x,+) \in T(L)\}$ and $T^-(L) = \{x : (x,-) \in T(L)\}$ for every $L \in \mathcal{L}$. We say that T is *positive* if $T^-(L) = \emptyset$ for every $L \in \mathcal{L}$. In the sequel, we will occasionally identify a positive mapping $L \mapsto T(L)$ with the mapping $L \mapsto T^+(L)$. The symbol "+" as an upper index of T will always indicate that the underlying mapping T is positive.
The following relation will help to clarify under which conditions the sets $(T(L))_{L\in\mathcal{L}}$ are teaching sets w.r.t. a suitably chosen preference relation:

$$R_T = \{(L,L') \in \mathcal{L} \times \mathcal{L} : \ (L \neq L') \wedge (L \text{ is consistent with } T(L'))\} \ .$$

The transitive closure of R_T is denoted as $\mathrm{trcl}(R_T)$ in the sequel. The following notion will play an important role in this paper:

Definition 15 *A mapping $L \mapsto T(L)$ with L ranging over all concepts in $\mathcal{L}$ is said to be* admissible for $\mathcal{L}$ *if the following holds:*

1. *For every $L \in \mathcal{L}$, L is consistent with $T(L)$.*
2. *The relation $\mathrm{trcl}(R_T)$ is asymmetric (which clearly implies that R_T is asymmetric too).*

If T is admissible, then $\mathrm{trcl}(R_T)$ is transitive and asymmetric, i.e., $\mathrm{trcl}(R_T)$ is a strict partial order on $\mathcal{L}$. We will therefore use the notation $\prec_T$ instead of $\mathrm{trcl}(R_T)$ whenever T is known to be admissible.

Lemma 16 *Suppose that T^+ is a positive admissible mapping for $\mathcal{L}$. Then the relation $\prec_{T^+}$ on $\mathcal{L}$ extends the relation $\supset$ on $\mathcal{L}$. More precisely, the following holds for all $L, L' \in \mathcal{L}$:*

$$L' \subset L \Rightarrow (L,L') \in R_{T^+} \Rightarrow L \prec_{T^+} L' \ .$$

Proof If T^+ is admissible, then L' is consistent with $T^+(L')$. Thus $T^+(L') \subseteq L' \subset L$ so that L is consistent with $T^+(L')$ too. Therefore $(L,L') \in R_{T^+}$, i.e., $L \prec_{T^+} L'$. ■

The following result clarifies how admissible mappings are related to preference-based teaching:

Lemma 17 *For each concept class $\mathcal{L}$, the following holds:*

$$\mathrm{PBTD}(\mathcal{L}) = \inf_T \mathrm{ord}(T) \ \textit{ and } \ \mathrm{PBTD}^+(\mathcal{L}) = \inf_{T^+} \mathrm{ord}(T^+)$$

where T ranges over all mappings that are admissible for $\mathcal{L}$ and T^+ ranges over all positive mappings that are admissible for $\mathcal{L}$.

Proof We restrict ourselves to the proof for $\mathrm{PBTD}(\mathcal{L}) = \inf_T \mathrm{ord}(T)$ because the equation $\mathrm{PBTD}^+(\mathcal{L}) = \inf_{T^+} \mathrm{ord}(T^+)$ can be obtained in a similar fashion. We first prove that $\mathrm{PBTD}(\mathcal{L}) \leq \inf_T \mathrm{ord}(T)$. Let T be an admissible mapping for $\mathcal{L}$. It suffices to show that, for every $L \in \mathcal{L}$, $T(L)$ is a teaching set for L w.r.t. $(\mathcal{L}, \prec_T)$. Suppose $L' \in \mathcal{L} \setminus \{L\}$ is consistent with $T(L)$. Then $(L',L) \in R_T$ and thus $L' \prec_T L$. It follows that $\prec_T$ prefers L over all concepts $L' \in \mathcal{L} \setminus \{L\}$ that are consistent with $T(L)$. Thus T is a teaching set for L w.r.t. $(\mathcal{L}, \prec_T)$, as desired.

We now prove that $\inf_T \mathrm{ord}(T) \leq \mathrm{PBTD}(\mathcal{L})$. Let $\prec$ be a strict partial order on $\mathcal{L}$ and let T be a mapping such that, for every $L \in \mathcal{L}$, $T(L)$ is a teaching set for L w.r.t. $(\mathcal{L}, \prec)$. It suffices to show that T is admissible for $\mathcal{L}$. Consider a pair $(L',L) \in R_T$. The definition of R_T implies that $L' \neq L$ and that L' is consistent with $T(L)$. Since $T(L)$ is a teaching set w.r.t. $(\mathcal{L}, \prec)$, it follows that $L' \prec L$. Thus, $\prec$ is an extension of R_T. Since $\prec$ is transitive, it is even an extension of $\mathrm{trcl}(R_T)$. Because $\prec$ is asymmetric, $\mathrm{trcl}(R_T)$ must be asymmetric, too. It follows that T is admissible. ■

6. Preference-based Teaching of Linear Sets

Some work in computational learning theory (Abe, 1989; Gao et al., 2015; Takada, 1992) is concerned with learning *semi-linear sets*, i.e., unions of linear subsets of $\mathbb{N}^k$ for some fixed $k \geq 1$, where each linear set consists of exactly those elements that can be written as the sum of some constant vector c and a linear combination of the elements of some fixed set of generators, see Example 2. While semi-linear sets are of common interest in mathematics in general, they play a particularly important role in the theory of formal languages, due to *Parikh's theorem*, by which the so-called Parikh vectors of strings in a context-free language always form a semi-linear set (Parikh, 1966).

A recent study (Gao et al., 2015) analyzed computational teaching of classes of linear subsets of $\mathbb{N}$ (where $k = 1$) and some variants thereof, as a substantially simpler yet still interesting special case of semi-linear sets. In this section, we extend that study to preference-based teaching.

Within the scope of this section, all concept classes are formulated over the universe $\mathcal{X} = \mathbb{N}_0$. Let $G = \{g_1, \ldots, g_k\}$ be a finite subset of $\mathbb{N}$. We denote by $\langle G \rangle$ resp. by $\langle G \rangle_+$ the following sets:

$$\langle G \rangle = \left\{ \sum_{i=1}^{k} a_i g_i : \ a_1, \ldots, a_k \in \mathbb{N}_0 \right\} \ \text{ and } \ \langle G \rangle_+ = \left\{ \sum_{i=1}^{k} a_i g_i : \ a_1, \ldots, a_k \in \mathbb{N} \right\} \ .$$

We will determine (at least approximately) the preference-based teaching dimension of the following concept classes over $\mathbb{N}_0$:

$$\begin{aligned}
\mathrm{LINSET}_k &= \{\langle G \rangle : \ (G \subset \mathbb{N}) \wedge (1 \leq |G| \leq k)\} \ . \\
\mathrm{CF\text{-}LINSET}_k &= \{\langle G \rangle : \ (G \subset \mathbb{N}) \wedge (1 \leq |G| \leq k) \wedge (\gcd(G) = 1)\} \ . \\
\mathrm{NE\text{-}LINSET}_k &= \{\langle G \rangle_+ : \ (G \subset \mathbb{N}) \wedge (1 \leq |G| \leq k)\} \ . \\
\mathrm{NE\text{-}CF\text{-}LINSET}_k &= \{\langle G \rangle_+ : \ (G \subset \mathbb{N}) \wedge (1 \leq |G| \leq k) \wedge (gcd(G) = 1)\} \ .
\end{aligned}$$

A subset of $\mathbb{N}_0$ whose complement in $\mathbb{N}_0$ is finite is said to be *co-finite*. The letters "CF" in CF-LINSET mean "co-finite". The concepts in LINSET_k have the algebraic structure of a monoid w.r.t. addition. The concepts in $\mathrm{CF\text{-}LINSET}_k$ are also known as "numerical semigroups" (Rosales

and García-Sánchez, 2009). A zero coefficient $a_j = 0$ erases g_j in the linear combination $\sum_{i=1}^{k} a_i g_i$. Coefficients from $\mathbb{N}$ are non-erasing in this sense. The letters "NE" in "NE-LINSET" mean "non-erasing".

The *shift-extension* $\mathcal{L}'$ of a concept class $\mathcal{L}$ over the universe $\mathbb{N}_0$ is defined as follows:

$$\mathcal{L}' = \{c + L : (c \in \mathbb{N}_0) \wedge (L \in \mathcal{L})\} \ . \tag{7}$$

The following bounds on RTD and RTD^+ (for sufficiently large values of k)[4] are known from (Gao et al., 2015):

	RTD^+	RTD
LINSET_k	$= \infty$	?
$\mathrm{CF\text{-}LINSET}_k$	$= k$	$\in \{k-1, k\}$
$\mathrm{NE\text{-}LINSET}'_k$	$= k+1$	$\in \{k-1, k, k+1\}$

Here $\mathrm{NE\text{-}LINSET}'_k$ denotes the shift-extension of $\mathrm{NE\text{-}LINSET}_k$.

The following result shows the corresponding bounds with PBTD in place of RTD:

Theorem 18 *The bounds in the following table are valid:*

	PBTD^+	PBTD
LINSET_k	$= k$	$\in \{k-1, k\}$
$\mathrm{CF\text{-}LINSET}_k$	$= k$	$\in \{k-1, k\}$
$\mathrm{NE\text{-}LINSET}_k$	$\in \left[\left\lfloor \frac{k-1}{2} \right\rfloor : k\right]$	$\in \left[\left\lfloor \frac{k-1}{2} \right\rfloor : k\right]$
$\mathrm{NE\text{-}CF\text{-}LINSET}_k$	$\in \left[\left\lfloor \frac{k-1}{2} \right\rfloor : k\right]$	$\in \left[\left\lfloor \frac{k-1}{2} \right\rfloor : k\right]$

Moreover

$$\mathrm{PBTD}^+(\mathcal{L}') = k+1 \ \wedge \ \mathrm{PBTD}(\mathcal{L}') \in \{k-1, k, k+1\} \tag{8}$$

holds for all $\mathcal{L} \in \{\mathrm{LINSET}_k, \mathrm{CF\text{-}LINSET}_k, \mathrm{NE\text{-}LINSET}_k, \mathrm{NE\text{-}CF\text{-}LINSET}_k\}$.

Note that the equation $\mathrm{PBTD}^+(\mathrm{LINSET}_k) = k$ was already proven in Example 2, using the fact that $G \mapsto \langle G \rangle$ is a closure operator. Since $G \mapsto \langle G \rangle_+$ is not a closure operator, we give a separate argument to prove an upper bound of k on $\mathrm{PBTD}^+(\mathrm{NE\text{-}LINSET}_k)$ (see Lemma 37 in Appendix A). All other upper bounds in Theorem 18 are then easy to derive. The lower bounds in Theorem 18 are much harder to obtain. A complete proof of Theorem 18 will be given in Appendix A.

Remark 19 *The lower bound on* $\mathrm{PBTD}^+(\mathrm{NE\text{-}LINSET}_k)$ *and* $\mathrm{PBTD}^+(\mathrm{NE\text{-}CF\text{-}LINSET}_k)$ *may be improved to* $k-1$*; see (Gao et al., 2017, Theorem 2, Appendix A.3).*

4. For instance, $\mathrm{RTD}^+(\mathrm{LINSET}_k) = \infty$ holds for all $k \geq 2$ and $\mathrm{RTD}(\mathrm{LINSET}_k) =$? (where "?" means "unknown") holds for all $k \geq 4$.

7. Preference-based Teaching of Halfspaces

In this section, we study preference-based teaching of halfspaces. We will denote the all-zeros vector as $\vec{0}$. The vector with 1 in coordinate i and with 0 in the remaining coordinates is denoted as $\vec{e}_i$. The dimension of the Euclidean space in which these vectors reside will always be clear from the context. The sign of a real number x (with value 1 if $x > 0$, value -1 if $x < 0$, and value 0 if $x = 0$) is denoted by $\mathrm{sign}(x)$.

Suppose that $w \in \mathbb{R}^d \setminus \{\vec{0}\}$ and $b \in \mathbb{R}$. The *(positive) halfspace induced by* w *and* b is then given by

$$H_{w,b} = \{x \in \mathbb{R}^d : \ w^\top x + b \geq 0\} \ .$$

Instead of $H_{w,0}$, we simply write H_w. Let $\mathcal{H}_d$ denote the class of d-dimensional Euclidean halfspaces:

$$\mathcal{H}_d = \{H_{w,b} : \ w \in \mathbb{R}^d \setminus \{\vec{0}\} \wedge b \in \mathbb{R}\} \ .$$

Similarly, $\mathcal{H}_d^0$ denotes the class of d-dimensional homogeneous Euclidean halfspaces:

$$\mathcal{H}_d^0 = \{H_w : \ w \in \mathbb{R}^d \setminus \{\vec{0}\}\} \ .$$

Let S_{d-1} denote the $(d-1)$-dimensional unit sphere in $\mathbb{R}^d$. Moreover $S_{d-1}^+ = \{x \in S_{d-1} : x_d > 0\}$ denotes the "northern hemisphere". If not stated explicitly otherwise, we will represent homogeneous halfspaces with normalized vectors residing on the unit sphere. We remind the reader of the following well-known fact:

Remark 20 *The orthogonal group in dimension* d *(i.e., the multiplicative group of orthogonal* $(d \times d)$*-matrices) acts transitively on* S_{d-1} *and it conserves the inner product.*

We now prove a helpful lemma, stating that each vector w^* in the northern hemisphere may serve as a representative for some homogeneous halfspace H_u in the sense that all other elements of H_u in the northern hemisphere have a strictly smaller d-th component than w^*. This will later help to teach homogeneous halfspaces with a preference that orders vectors by the size of their last coordinate.

Lemma 21 *Let* $d \geq 2$*, let* $0 < h \leq 1$ *and let* $R_{d,h} = \{w \in S_{d-1} : w_d = h\}$*. With this notation the following holds. For every* $w^* \in R_{d,h}$*, there exists* $u \in \mathbb{R}^d \setminus \{\vec{0}\}$ *such that*

$$(w^* \in H_u) \wedge (\forall w \in (S_{d-1}^+ \cap H_u) \setminus \{w^*\} : w_d < h) \ . \tag{9}$$

Proof For $h = 1$, the statement is trivial, since $R_{d,1} = \{\vec{e}_d\}$. So let $h < 1$.

Because of Remark 20, we may assume without loss of generality that the vector $w^* \in R_{d,h}$ equals $(0, \ldots, 0, \sqrt{1-h^2}, h)$. It suffices therefore to show that, with this choice of w^*, the vector $u = (0, \ldots, 0, w_d^*, -w_{d-1}^*)$ satisfies (9). Note that $w \in H_u$ iff $\langle u, w \rangle = w_d^* w_{d-1} - w_{d-1}^* w_d \geq 0$. Since $\langle u, w^* \rangle = 0$, we have $w^* \in H_u$. Moreover, it follows that

$$S_{d-1}^+ \cap H_u = \left\{ w \in S_{d-1}^+ : \frac{w_{d-1}}{w_d} \geq \frac{w_{d-1}^*}{w_d^*} > 0 \right\} \ .$$

It is obvious that no vector $w \in S_{d-1}^+ \cap H_u$ can have a d-th component w_d exceeding $w_d^* = h$ and that setting $w_d = h = w_d^*$ forces the settings $w_{d-1} = w_{d-1}^* = \sqrt{1-h^2}$ and $w_1 = \ldots = w_{d-2} = 0$.

Consequently, (9) is satisfied, which concludes the proof. ■

With this lemma in hand, we can now prove an upper bound of 2 for the preference-based teaching dimension of the class of homogeneous halfspaces, independent of the underlying dimension d.

Theorem 22 $\mathrm{PBTD}(\mathcal{H}_1^0) = \mathrm{TD}(\mathcal{H}_1^0) = 1$ *and, for every* $d \geq 2$*, we have* $\mathrm{PBTD}(\mathcal{H}_d^0) \leq 2$.

Proof Clearly, $\mathrm{PBTD}(\mathcal{H}_1^0) = \mathrm{TD}(\mathcal{H}_1^0) = 1$ since $\mathcal{H}_1^0$ consists of the two sets $\{x \in \mathbb{R} : x \geq 0\}$ and $\{x \in \mathbb{R} : x \leq 0\}$.
Suppose now that $d \geq 2$. Let w^* be the target weight vector (i.e., the weight vector that has to be taught). Under the following conditions, we may assume without loss of generality that $w_d^* \neq 0$:

- For any $0 < s_1 < s_2$, the student prefers any weight vector that ends with s_2 zero coordinates over any weight vector that ends with only s_1 zero coordinates.
- If the target vector ends with (exactly) s zero coordinates, then the teacher presents only examples ending with (at least) s zero coordinates.

In the sequel, we specify a student and a teacher such that these conditions hold, so that we will consider only target weight vectors w^* with $w_d^* \neq 0$.
The student has the following preference relation:

- Among the weight vectors w with $w_d \neq 0$, the student prefers vectors with larger values of $|w_d|$ over those with smaller values of $|w_d|$.

The teacher will use two examples. The first one is chosen as

$$\begin{cases} (-\vec{e}_d, -) & \text{if } w_d^* > 0 \\ (\vec{e}_d, -) & \text{if } w_d^* < 0 \end{cases} .$$

This example reveals whether the unknown weight vector $w^* \in S_{d-1}$ has a strictly positive or a strictly negative d-th component. For reasons of symmetry, we may assume that $w_d^* > 0$. We are now precisely in the situation that is described in Lemma 21. Given w^* and $h = w_d^*$, the teacher picks as a second example $(u, +)$ where $u \in \mathbb{R}^d \setminus \{\vec{0}\}$ has the properties described in the lemma. It follows immediately that the student's preferences will make her choose the weight vector w^*. ■

The upper bound of 2 given in Theorem 22 is tight, as is stated in the following lemma.

Lemma 23 *For every* $d \geq 2$*, we have* $\mathrm{PBTD}(\mathcal{H}_d^0) \geq 2$.

Proof We verify this lemma via Lemma 7, by providing a finite subclass $\mathcal{F}$ of $\mathcal{H}_2^0$ such that $\mathrm{TD}_{min}(\mathcal{F}) = 2$. Let $\mathcal{F} = \{H_w : \vec{0} \neq w \in \{-1, 0, 1\}^2\}$. It is easy to verify that each of the 8 halfspaces in $\mathcal{F}$ has a teaching dimension of 2 with respect to $\mathcal{F}$. This example can be extended to higher dimensions in the obvious way. ■

We thus conclude that the class of homogeneous halfspaces has a preference-based teaching dimension of 2, independent of the dimensionality $d \geq 2$.

Corollary 24 *For every* $d \geq 2$*, we have* $\mathrm{PBTD}(\mathcal{H}_d^0) = 2$.

By contrast, we will show next that the recursive teaching dimension of the class of homogeneous halfspaces grows with the dimensionality.

Theorem 25 *For any* $d \geq 2$, $\mathrm{TD}(\mathcal{H}_d^0) = \mathrm{RTD}(\mathcal{H}_d^0) = d + 1$.

Proof Assume by normalization that the target weight vector has norm 1, i.e., it is taken from S_{d-1}. Remark 20 implies that all weight vectors in S_{d-1} are equally hard to teach. It suffices therefore to show that $\mathrm{TD}(H_{\vec{e}_1}, \mathcal{H}_d^0) = d + 1$.

We first show that $\mathrm{TD}(H_{\vec{e}_1}, \mathcal{H}_d^0) \leq d + 1$. Define $u = -\sum_{i=2}^d \vec{e}_i$. We claim that $T = \{(\vec{e}_i, +) : 2 \leq i \leq d\} \cup \{(u, +), (\vec{e}_1, +)\}$ is a teaching set for $H_{\vec{e}_1}$ w.r.t. $\mathcal{H}_d^0$. Consider any $w \in S_{d-1}$ such that H_w is consistent with T. Note that $w_i = \langle \vec{e}_i, w \rangle \geq 0$ for all $i \in \{2, \ldots, d\}$ and $\langle u, w \rangle = -\sum_{i=2}^d w_i \geq 0$ together imply that $w_i = 0$ for all $i \in \{2, \ldots, d\}$ and therefore $w = \pm\vec{e}_1$. Furthermore, $w_1 = \langle w, \vec{e}_1 \rangle \geq 0$, and so $w = \vec{e}_1$, as required.

Now we show that $\mathrm{TD}(H_{\vec{e}_1}, \mathcal{H}_d^0) \geq d + 1$ holds for all $d \geq 2$. It is easy to see that two examples do not suffice for distinguishing $\vec{e}_1 \in \mathbb{R}^2$ from all weight vectors in S_1. In other words, $\mathrm{TD}(H_{\vec{e}_1}, \mathcal{H}_2^0) \geq 3$. Suppose now that $d \geq 3$. It is furthermore easy to see that a teaching set T which distinguishes $\vec{e}_1$ from all weight vectors in S_{d-1} must contain at least one positive example u that is orthogonal to $\vec{e}_1$. The inequality $\mathrm{TD}(H_{\vec{e}_1}, \mathcal{H}_d^0) \geq d + 1$ is now obtained inductively because the example $(u, +) \in T$ leaves open a problem that is not easier than teaching $\vec{e}_1$ w.r.t. the $(d-2)$-dimensional sphere $\{x \in S_{d-1} : x \perp u\}$. ■

We have thus established that the class of homogeneous halfspaces has a recursive teaching dimension growing linearly with d, while its preference-based teaching dimension is constant. In the case of general (i.e., not necessarily homogeneous) d-dimensional halfspaces, the difference between RTD and PBTD is even more extreme. On the one hand, by generalizing the proof of Lemma 10, it is easy to see that $\mathrm{RTD}(\mathcal{H}_d) = \infty$ for all $d \geq 1$. On the other hand, we will show in the remainder of this section that $\mathrm{PBTD}(\mathcal{H}_d) \leq 6$, independent of the value of d.

We will assume in the sequel (by way of normalization) that an inhomogeneous halfspace has a bias $b \in \{\pm 1\}$. We start with the following result:

Lemma 26 *Let* $w^* \in \mathbb{R}^d$ *be a vector with a non-trivial* d*-th component* $w_d^* \neq 0$ *and let* $b^* \in \{\pm 1\}$ *be a bias. Then there exist three examples labeled according to* H_{w^*,b^*} *such that the following holds. Every weight-bias pair* (w, b) *consistent with these examples satisfies* $b = b^*$, $\mathrm{sign}(w_d) = \mathrm{sign}(w_d^*)$ *and*

$$\begin{cases} |w_d| \geq |w_d^*| & \text{if } b^* = -1 \\ |w_d| \leq |w_d^*| & \text{if } b^* = +1 \end{cases} . \tag{10}$$

Proof Within the proof, we use the label "1" instead of "+" and the label "−1" instead of "−". The pair (w, b) denotes the student's hypothesis for the target weight-bias pair (w^*, b^*). The examples shown to the student will involve the unknown quantities w^* and b^*. Each example will lead to a new constraint on w and b. We will see that the collection of these constraints reveals the required information. We proceed in three stages:

1. The first example is chosen as $(\vec{0}, b^*)$. The pair (w, b) can be consistent with this example only if $b = -1$ in the case that $b^* = -1$ and $b \in \{0, 1\}$ in the case that $b^* = 1$.

2. The next example is chosen as $\vec{a}_2 = -\frac{2b^*}{w_d^*} \cdot \vec{e}_d$ and labeled "$-b^*$". Note that $\langle w^*, \vec{a}_2 \rangle + b^* = -b^*$. We obtain the following new constraint:

$$\langle w, \vec{a}_2 \rangle + b = \begin{cases} -2\frac{w_d}{w_d^*} + \overbrace{b}^{\in\{0,1\}} < 0 & \text{if } b^* = 1 \\ +2\frac{w_d}{w_d^*} + \underbrace{b}_{=-1} \geq 0 & \text{if } b^* = -1 \end{cases} .$$

   The pair (w, b) with $b = b^*$ if $b^* = -1$ and $b \in \{0, 1\}$ if $b^* = 1$ can satisfy the above constraint only if the sign of w_d equals the sign of w_d^*.

3. The third example is chosen as the example $\vec{a}_3 = -\frac{b^*}{w_d^*} \cdot \vec{e}_d$ with label "1". Note that $\langle w^*, \vec{a}_3 \rangle^* + b^* = 0$. We obtain the following new constraint:

$$\langle w, \vec{a}_3 \rangle = -\frac{b^* w_d}{w_d^*} + b \geq 0 \ .$$

   Given that w is already constrained to weight vectors satisfying $\text{sign}(w_d) = \text{sign}(w_d^*)$, we can safely replace w_d / w_d^* by $|w_d|/|w_d^*|$. This yields $|w_d|/|w_d^*| \leq b$ if $b^* = 1$ and $|w_d|/|w_d^*| \geq -b$ if $b^* = -1$. Since b is already constrained as described in stage 1 above, we obtain $|w_d|/|w_d^*| \leq b \in \{0, 1\}$ if $b^* = 1$ and $|w_d|/|w_d^*| \geq -b = 1$ if $b^* = -1$. The weight-bias pair (w, b) satisfies these constraints only if $b = b^*$ and if (10) is valid.

The assertion of the lemma is immediate from this discussion. ■

Theorem 27 $\text{PBTD}(\mathcal{H}_d) \leq 6$.

Proof As in the proof of Lemma 26, we use the label "1" instead of "+" and the label "-1" instead of "$-$". As in the proof of Theorem 22, we may assume without loss of generality that the target weight vector $w^* \in \mathbb{R}^d$ satisfies $w_d^* \neq 0$. The proof will proceed in stages. On the way, we specify six rules which determine the preference relation of the student.

Stage 1 is concerned with teaching homogeneous halfspaces given by w^* (and $b^* = 0$). The student respects the following rules:

Rule 1: She prefers any pair $(w, 0)$ over any pair (w', b) with $b \neq 0$. In other words, any homogeneous halfspace is preferred over any non-homogeneous halfspace.

Rule 2: Among homogeneous halfspaces, her preferences are the same as the ones that were used within the proof of Theorem 22 for teaching homogeneous halfspaces.

Thus, if $b^* = 0$, then we can simply apply the teaching protocol for homogeneous halfspaces. In this case, w^* can be taught at the expense of only two examples.

Stage 1 reduces the problem to teaching inhomogeneous halfspaces given by (w^*, b^*) with $b^* \neq 0$. We assume, by way of normalization, that $b^* \in \{\pm 1\}$, but note that w^* can now not be assumed to be of unit (or any other fixed) length.

In **stage 2**, the teacher presents three examples in accordance with Lemma 26. It follows that the student will take into consideration only weight-bias pairs (w, b) such that the constraints $b = b^*$, $\text{sign}(w_d) = \text{sign}(w_d^*)$ and (10) are satisfied. The following rule will then induce the constraint $w_d = w_d^*$:

Rule 3: Among the pairs (w, b) such that $w_d \neq 0$ and $b \in \{\pm 1\}$, the student's preferences are as follows. If $b = -1$ (resp. $b = 1$), then she prefers vectors w with a smaller (resp. larger) value of $|w_d|$ over those with a larger (resp. smaller) value of $|w_d|$.

Thanks to Lemma 26 and thanks to Rule 3, we may from now on assume that $b = b^*$ and $w_d = w_d^*$. In the sequel, let w^* be decomposed according to $w^* = (\vec{w}_{d-1}^*, w_d^*) \in \mathbb{R}^{d-1} \times \mathbb{R}$. We think of $\vec{w}_{d-1}$ as the student's hypothesis for $\vec{w}_{d-1}^*$.

Stage 3 is concerned with the special case where $\vec{w}_{d-1}^* = \vec{0}$. The student will automatically set $\vec{w}_{d-1} = \vec{0}$ if we add the following to the student's rule system:

Rule 4: Given that the values for w_d and b have been fixed already (and are distinct from 0), the student prefers weight-bias pairs with $\vec{w}_{d-1} = \vec{0}$ over any weight-bias pair with $\vec{w}_{d-1} \neq \vec{0}$.

Stage 3 reduces the problem to teaching (w^*, b^*) with fixed non-zero values for w_d and b^* (known to the student) and with $\vec{w}_{d-1}^* \neq \vec{0}$. Thus, essentially, only $\vec{w}_{d-1}^*$ has still to be taught. In the next stage, we will argue that the problem of teaching $\vec{w}_{d-1}^*$ is equivalent to teaching a homogeneous halfspace.

In **stage 4**, the teacher will present only examples a such that $a_d = -\frac{b^*}{w_d^*}$ so that the contribution of the d-th component to the inner product of w^* and a cancels with the bias b^*. Given this commitment for a_d, the first $d-1$ components of the examples can be chosen so as to teach the homogeneous halfspace $H_{\vec{w}_{d-1}^*}$. According to Theorem 22, this can be achieved at the expense of two more examples. Of course the student's preferences must match with the preferences that were used in the proof of this theorem:

Rule 5: Suppose that the values of w_d and b have been fixed already (and are distinct from 0) and suppose that $\vec{w}_{d-1} \neq \vec{0}$. Then the preferences for the choice of $\vec{w}_{d-1}$ match with the preferences that were used in the protocol for teaching homogeneous halfspaces.

After stage 4, the student takes into consideration only weight-bias pairs (w, b) such that $w_d = w_d^*$, $b = b^*$ and $H_{\vec{w}_{d-1}} = H_{\vec{w}_{d-1}^*}$. However, since we had normalized the bias and not the weight vector, this does not necessarily mean that $\vec{w}_{d-1} = \vec{w}_{d-1}^*$. On the other hand, the two weight vectors already coincide modulo a positive scaling factor, say

$$\vec{w}_{d-1} = s \cdot \vec{w}_{d-1}^* \text{ for some } s > 0 \ . \tag{11}$$

In order to complete the proof, it suffices to teach the L_1-norm of $\vec{w}_{d-1}^*$ to the student (because (11) and $\|\vec{w}_{d-1}\|_1 = \|\vec{w}_{d-1}^*\|_1$ imply that $\vec{w}_{d-1} = \vec{w}_{d-1}^*$). The next (and final) stage serves precisely this purpose.

As for **stage 5**, we first fix some notation. For $i = 1, \ldots, k-1$, let $\beta_i = \text{sign}(w_i^*)$. Note that (11) implies that $\beta_i = \text{sign}(w_i)$. Let $L = \|\vec{w}_{d-1}^*\|_1$ denote the L_1-norm of $\vec{w}_{d-1}^*$. The final example is chosen as $\vec{a}_6 = (\beta_1, \ldots, \beta_{d-1}, -(L + b^*)/w_d^*)$ and labeled "1". Note that

$$\langle w^*, \vec{a_6} \rangle + b^* = |w_1^*| + \ldots + |w_{d-1}^*| - L = 0 \ .$$

Given that $\beta_i = \text{sign}(w_i)$, $w_d = w_d^*$ and $b = b^*$, the student can derive from $\vec{a_6}$ and its label the following constraint on $\vec{w}_{d-1}$:

$$\langle w, \vec{a_6} \rangle + b = |w_1| + \ldots + |w_{d-1}| - L \geq 0 \ .$$

In combination with the following rule, we can now force the constraint $\|\vec{w}_{d-1}\|_1 = L$:

Rule 6: Suppose that the values of w_d and b have been fixed already (and are distinct from 0) and suppose that $H_{\vec{w}_{d-1}}$ has already been fixed. Then, among the vectors representing $H_{\vec{w}_{d-1}}$, the ones with a smaller L_1-norm are preferred over the ones with a larger L_1-norm.

An inspection of the six stages reveals that at most six examples altogether were shown to the student (three in stage 2, two in stage 4, and one in stage 5). This completes the proof of the theorem. ■

Note that Theorems 22 and 27 remain valid when we allow w to be the all-zero vector, which extends $\mathcal{H}_d^0$ by $\{\mathbb{R}^d\}$ and $\mathcal{H}_d$ by $\{\mathbb{R}^d, \emptyset\}$. $\mathbb{R}^d$ will be taught with a single positive example, and $\emptyset$ with a single negative example. The student will give the highest preference to $\mathbb{R}^d$, the second highest to $\emptyset$, and among the remaining halfspaces, the student's preferences stay the same.

8. Classes with PBTD or PBTD^+ Equal to One

In this section, we will give complete characterizations of (i) the concept classes with a positive preference-based teaching dimension of 1, and (ii) the concept classes with a preference-based teaching dimension of 1. Throughout this section, we use the label "1" to indicate positive examples and the label "0" to indicate negative examples.

Let I be a (possibly infinite) index set. We will consider a mapping $A : I \times I \to \{0,1\}$ as a binary matrix $A \in \{0,1\}^{I \times I}$. A is said to be *lower-triangular* if there exists a linear ordering $\prec$ on I such that $A(i,i') = 0$ for every pair (i,i') such that $i \prec i'$.

We will occasionally identify a set $L \subseteq \mathcal{X}$ with its indicator function by setting $L(x) = \mathbb{1}_{[x \in L]}$. For each $M \subseteq \mathcal{X}$, we define

$$M \oplus L = (L \setminus M) \cup (M \setminus L)$$

and

$$M \oplus \mathcal{L} = \{M \oplus L : L \in \mathcal{L}\} \ .$$

For $T \subseteq \mathcal{X} \times \{0,1\}$, we define similarly

$$M \oplus T = \{(x, \bar{y}) : (x,y) \in T \text{ and } x \in M\} \cup \{(x,y) \in T : x \notin M\} \ .$$

Moreover, given $M \subseteq \mathcal{X}$ and a linear ordering $\prec$ on $\mathcal{L}$, we define a linear ordering $\prec_M$ on $M \oplus \mathcal{L}$ as follows:

$$M \oplus L' \prec_M M \oplus L \Leftrightarrow \underbrace{M \oplus (M \oplus L')}_{=L'} \prec \underbrace{M \oplus (M \oplus L)}_{=L} \ .$$

Lemma 28 *With this notation, the following holds. If the mapping $\mathcal{L} \ni L \mapsto T(L) \subseteq \mathcal{X} \times \{0,1\}$ assigns a teaching set to L w.r.t. $(\mathcal{L}, \prec)$, then the mapping $M \oplus \mathcal{L} \ni M \oplus L \mapsto M \oplus T(L) \subseteq \mathcal{X} \times \{0,1\}$ assigns a teaching set to $M \oplus L$ w.r.t. $(M \oplus \mathcal{L}, \prec_M)$.*

Since this result is rather obvious, we skip its proof.

We say that $\mathcal{L}$ and $\mathcal{L}'$ are *equivalent* if $\mathcal{L}' = M \oplus \mathcal{L}$ for some $M \subseteq \mathcal{X}$ (and this clearly is an equivalence relation). As an immediate consequence of Lemma 28, we obtain the following result:

Lemma 29 *If $\mathcal{L}$ is equivalent to $\mathcal{L}'$, then $\mathrm{PBTD}(\mathcal{L}) = \mathrm{PBTD}(\mathcal{L}')$.*

The following lemma provides a necessary condition for a concept class to have a preference-based teaching dimension of one.

Lemma 30 *Suppose that $\mathcal{L} \subseteq 2^{\mathcal{X}}$ is a concept class of PBTD 1. Pick a linear ordering $\prec$ on $\mathcal{L}$ and a mapping $\mathcal{L} \ni L \mapsto (x_L, y_L) \in \mathcal{X} \times \{0,1\}$ such that, for every $L \in \mathcal{L}$, $\{(x_L, y_L)\}$ is a teaching set for L w.r.t. $(\mathcal{L}, \prec)$. Then*

- *either every instance $x \in \mathcal{X}$ occurs at most once in $(x_L)_{L \in \mathcal{L}}$*
- *or there exists a concept $L^* \in \mathcal{L}$ that is preferred over all other concepts in $\mathcal{L}$ and x_{L^*} is the only instance from $\mathcal{X}$ that occurs twice in $(x_L)_{L \in \mathcal{L}}$.*

Proof Since the mapping T must be injective, no instance can occur twice in $(x_L)_{L \in \mathcal{L}}$ with the same label. Suppose that there exists an instance $x \in \mathcal{X}$ and concepts $L \prec L^*$ such that $x = x_L = x_{L^*}$ and, w.l.o.g., $y_L = 1$ and $y_{L^*} = 0$. Since $\{(x,1)\}$ is a teaching set for L w.r.t. $(\mathcal{L}, \prec)$, every concept $L' \succ L$ (including the ones that are preferred over L^*) must satisfy $L'(x) = 0$. For analogous reasons, every concept $L' \succ L^*$ (if any) must satisfy $L'(x) = 1$. A concept $L' \in \mathcal{L}$ that is preferred over L^* would have to satisfy $L'(x) = 0$ and $L'(x) = 1$, which is impossible. It follows that there can be no concept that is preferred over L^*. ■

The following result is a consequence of Lemmas 28 and 30.

Theorem 31 *If $\mathrm{PBTD}(\mathcal{L}) = 1$, then there exists a concept class $\mathcal{L}'$ that is equivalent to $\mathcal{L}$ and satisfies $\mathrm{PBTD}(\mathcal{L}') = \mathrm{PBTD}^+(\mathcal{L}') = 1$.*

Proof Pick a linear ordering $\prec$ on $\mathcal{L}$ and, for every $L \in \mathcal{L}$, a pair $(x_L, y_L) \in \mathcal{X} \times \{0,1\}$ such that $T(L) = \{(x_L, y_L)\}$ is a teaching set for L w.r.t. $(\mathcal{L}, \prec)$.

Case 1: Every instance $x \in \mathcal{X}$ occurs at most once in $(x_L)_{L \in \mathcal{L}}$.
Then choose $M = \{x_L : y_L = 0\}$ and apply Lemma 28.

Case 2: There exists a concept $L^* \in \mathcal{L}$ that is preferred over all other concepts in $\mathcal{L}$ and x_{L^*} is the only instance from $\mathcal{X}$ that occurs twice in $(x_L)_{L \in \mathcal{L}}$.
Then choose $M = \{x_L : y_L = 0 \wedge L \neq L^*\}$ and apply Lemma 28. With this choice, we obtain $M \oplus T(L) = \{(x_L, 1)\}$ for every $L \in \mathcal{L} \setminus \{L^*\}$. Since L^* is preferred over all other concepts in $\mathcal{L}$, we may teach L^* w.r.t. $(\mathcal{L}, \prec)$ by the empty set (instead of employing a possibly 0-labeled example).

The discussion shows that there is a class $\mathcal{L}'$ that is equivalent to $\mathcal{L}$ and can be taught in the preference-based model with positive teaching sets of size 1 (or size 0 in case of L^*). ■

We now have the tools required for characterizing the concept classes whose positive PBTD equals 1.

Theorem 32 $\mathrm{PBTD}^+(\mathcal{L}) = 1$ *if and only if there exists a mapping $\mathcal{L} \ni L \mapsto x_L \in \mathcal{X}$ such that the matrix $A \in \{0,1\}^{(\mathcal{L} \setminus \{\emptyset\}) \times (\mathcal{L} \setminus \{\emptyset\})}$ given by $A(L, L') = L'(x_L)$ is lower-triangular.*

Proof Suppose first that $\mathrm{PBTD}^+(\mathcal{L}) = 1$. Pick a linear ordering $\prec$ on $\mathcal{L}$ and, for every $L \in \mathcal{L}\setminus\{\emptyset\}$, pick $x_L \in \mathcal{X}$ such that $\{x_L\}$ is a positive teaching set for L w.r.t. $(\mathcal{L}, \prec)$.[5] If $L \prec L'$ (so that L' is preferred over L), we must have $L'(x_L) = 0$. It follows that the matrix A, as specified in the theorem, is lower-triangular.

Suppose conversely that there exists a mapping $\mathcal{L} \ni L \mapsto x_L \in \mathcal{X}$ such that the matrix $A \in \{0,1\}^{(\mathcal{L}\setminus\{\emptyset\})\times(\mathcal{L}\setminus\{\emptyset\})}$ given by $A(L, L') = L'(x_L)$ is lower-triangular, say w.r.t. the linear ordering $\prec$ on $\mathcal{L} \setminus \{\emptyset\}$. Then, for every $L \in \mathcal{L} \setminus \{\emptyset\}$, the singleton $\{x_L\}$ is a positive teaching set for L w.r.t. $(\mathcal{L}, \prec)$ because it distinguishes L from $\emptyset$ (of course) and also from every concept $L' \in \mathcal{L} \setminus \{\emptyset\}$ such that $L' \succ L$. If $\emptyset \in \mathcal{L}$, then extend the linear ordering $\prec$ by preferring $\emptyset$ over every other concept from $\mathcal{L}$ (so that $\emptyset$ is a positive teaching set for $\emptyset$ w.r.t. $(\mathcal{L}, \prec)$). ■

In view of Theorem 31, Theorem 32 characterizes every class $\mathcal{L}$ with $\mathrm{PBTD}(\mathcal{L}) = 1$ up to equivalence.

Let $\mathrm{Sg}(\mathcal{X}) = \{\{x\} : x \in \mathcal{X}\}$ denote the class of singletons over $\mathcal{X}$ and suppose that $\mathrm{Sg}(\mathcal{X})$ is a sub-class of $\mathcal{L}$ and $\mathrm{PBTD}(\mathcal{L}) = 1$. We will show that only fairly trivial extensions of $\mathrm{Sg}(\mathcal{X})$ with a preference-based dimension of 1 are possible.

Lemma 33 *Let $\mathcal{L} \subseteq 2^{\mathcal{X}}$ be a concept class of $\mathrm{PBTD}\,1$ that contains $\mathrm{Sg}(\mathcal{X})$. Let T be an admissible mapping for $\mathcal{L}$ that assigns a labeled example $(x_L, y_L) \in \mathcal{X} \times \{0,1\}$ to each $L \in \mathcal{L}$. For $b = 0, 1$, let $\mathcal{L}^b = \{L \in \mathcal{L} : y_L = b\}$. Similarly, let $\mathcal{X}^b = \{x \in \mathcal{X} : y_{\{x\}} \in \mathcal{L}^b\}$. With this notation, the following holds:*

1. *If $L \in \mathcal{L}^1$ and $L \subset L' \in \mathcal{L}$, then $L' \in \mathcal{L}^1$.*
2. *If $L' \in \mathcal{L}^0$ and $L' \supset L \in \mathcal{L}$, then $L \in \mathcal{L}^0$.*
3. *$|\mathcal{X}^0| \leq 2$. Moreover if $|\mathcal{X}^0| = 2$, then there exist $q \neq q' \in \mathcal{X}$ such that $\mathcal{X}^0 = \{q, q'\}$ and $x_{\{q\}} = q'$.*

Proof Recall that $R_T = \{(L, L') \in \mathcal{L} \times \mathcal{L} : \ (L \neq L') \wedge (L \text{ is consistent with } T(L'))\}$ and that R_T (and even the transitive closure of R_T) is asymmetric if T is admissible.

1. If $L \in \mathcal{L}^1$ and $L \subset L'$, then $y_L = 1$ so that L' is consistent with the example (x_L, y_L). It follows that $(L', L) \in R_T$. $L' \in \mathcal{L}^0$ would similarly imply that $(L, L') \in R_T$ so that R_T would not be asymmetric. This is in contradiction with the admissibility of T.

2. The second assertion in the lemma is a logically equivalent reformulation of the first assertion.

3. Suppose for the sake of contradiction that $\mathcal{X}^0$ contains three distinct points, say q_1, q_2, q_3. Since, for $i = 1, 2, 3$, T assigns a 0-labeled example to $\{q_i\}$, at least one of the remaining two points is consistent with $T(\{q_i\})$. Let G be the digraph with the nodes q_1, q_2, q_3 and with an edge from q_j to q_i iff $\{q_j\}$ is consistent with $T(\{q_i\})$. Then each of the three nodes has an indegree of at least 1. Digraphs of this form must contain a cycle so that $\mathrm{trcl}(R_T)$ is not asymmetric. This is in contradiction with the admissibility of R_T.

5. Such an x_L always exists, even if $\emptyset$ is a teaching set for L, because every superset of a teaching set for L that is still consistent with L is still a teaching set for L, cf. the discussion immediately after Lemma 4.

A similar argument holds if $\mathcal{X}^0$ contains only two distinct elements, say q and q'. If neither $x_{\{q\}} = q'$ nor $x_{\{q'\}} = q$, then $(\{q'\}, \{q\}) \in R_T$ and $(\{q\}, \{q'\}) \in R_T$ so that R_T is not asymmetric — again a contradiction to the admissibility of R_T.

■

We are now in the position to characterize those classes of PBTD one that contain all singletons.

Theorem 34 *Suppose that $\mathcal{L} \subseteq 2^{\mathcal{X}}$ is a concept class that contains $\mathrm{Sg}(\mathcal{X})$. Then $\mathrm{PBTD}(\mathcal{L}) = 1$ if and only if the following holds. Either $\mathcal{L}$ coincides with $\mathrm{Sg}(\mathcal{X})$ or $\mathcal{L}$ contains precisely one additional concept, which is either the empty set or a set of size 2.*

Proof We start with proving "⇐". It is well known that $\mathrm{PBTD}^+(\mathcal{L}) = 1$ for $\mathcal{L} = \mathrm{Sg}(\mathcal{X}) \cup \{\emptyset\}$: prefer $\emptyset$ over any singleton set, set $T(\emptyset) = \emptyset$ and, for every $x \in \mathcal{X}$, set $T(\{x\}) = \{(x, 1)\}$. In a similar fashion, we can show that $\mathrm{PBTD}(\mathcal{L}) = 1$ for $\mathcal{L} = \mathrm{Sg}(\mathcal{X}) \cup \{\{q, q'\}\}$ for any choice of $q \neq q' \in \mathcal{X}$. Prefer $\{q, q'\}$ over $\{q\}$ and $\{q'\}$, respectively. Furthermore, prefer $\{q\}$ and $\{q'\}$ over all other singletons. Finally, set $T(\{q, q'\}) = \emptyset$, $T(\{q\}) = \{(q', 0)\}$, $T(\{q'\}) = \{(q, 0)\}$ and, for every $x \in \mathcal{X} \setminus \{q, q'\}$, set $T(\{x\}) = \{(x, 1)\}$.

As for the proof of "⇒", we make use of the notions $T, x_L, y_L, \mathcal{L}^0, \mathcal{L}^1, \mathcal{X}^0, \mathcal{X}^1$ that had been introduced in Lemma 33 and we proceed by case analysis.

Case 1: $\mathcal{X}^0 = \emptyset$.
Since $\mathcal{X}^0 = \emptyset$, we have $\mathcal{X} = \mathcal{X}^1$. In combination with the first assertion in Lemma 33, it follows that $\mathcal{L} \setminus \{\emptyset\} = \mathcal{L}^1$. We claim that no concept in $\mathcal{L}$ contains two distinct elements. Assume for the sake of contradiction that there is a concept $L \in \mathcal{L}$ such that $|L| \geq 2$. It follows that, for every $q \in L$, $x_{\{q\}} = q$ and $y_{\{q\}} = 1$ so that $(L, \{q\}) \in R_T$. Moreover, there exists $q_0 \in L$ such that $x_L = q_0$ and $y_L = 1$. It follows that $(\{q_0\}, L) \in R_T$, which contradicts the fact that R_T is asymmetric.

Case 2: $\mathcal{X}^0 = \{q\}$ for some $q \in \mathcal{X}$.
Set $q' = x_{\{q\}}$ and note that $y_{\{q\}} = 0$. Moreover, since $\mathcal{X}^1 = \mathcal{X} \setminus \{q\}$, we have $x_{\{p\}} = p$ and $y_{\{p\}} = 1$ for every $p \in \mathcal{X} \setminus \{q\}$. We claim that $\mathcal{L}$ cannot contain a concept L of size at least 2 that contains an element of $\mathcal{X} \setminus \{q, q'\}$. Assume for the sake of contradiction, that there is a set L such that $|L| \geq 2$ and $p \in L$ for some $p \in \mathcal{X} \setminus \{q, q'\}$. The first assertion in Lemma 33 implies that $y_L = 1$ (because $y_{\{p\}} = 1$ and $\{p\} \subseteq L$). Since all pairs $(x, 1)$ with $x \neq q$ are already in use for teaching the corresponding singletons, we may conclude that $q \in L$ and $T(L) = \{(q, 1)\}$. This contradicts the fact that $\mathrm{trcl}(R_T)$ is asymmetric, because our discussion implies that $(L, \{p\}), (\{p\}, \{q\}), (\{q\}, L) \in R_T$. We may therefore safely assume that there is no concept of size at least 2 in $\mathcal{L}$ that has a non-empty intersection with $\mathcal{X} \setminus \{q, q'\}$. Thus, except for the singletons, the only remaining sets that possibly belong to $\mathcal{L}$ are $\emptyset$ and $\{q, q'\}$. We still have to show that not both of them can belong to $\mathcal{L}$. Assume for the sake of contradiction that $\emptyset, \{q, q'\} \in \mathcal{L}$. Since $\emptyset$ is consistent with $T(\{q\}) = \{(q', 0)\}$, we have $(\emptyset, \{q\}) \in R_T$. Clearly, $y_\emptyset = 0$. Since $\{q\}$ is consistent with every pair $(x, 0)$ except for $(q, 0)$, we must have $x_\emptyset = q$. (Otherwise, we have $(\{q\}, \emptyset) \in R_T$ and arrive at a contradiction.) Let us now inspect the possible teaching sets for $L = \{q, q'\}$. Since $\{q, q'\}$ is consistent with $T(\{q'\}) = \{(q', 1)\}$, setting $y_L = 0$ would lead to a contradiction. The example $(q', 1)$

is already in use for teaching $\{q'\}$. It is therefore necessary to set $T(L) = \{(q, 1)\}$. An inspection of the various teaching sets shows that $(\emptyset, \{q\}), (\{q\}, L), (L, \{q'\}), (\{q'\}, \emptyset) \in R_T$, which contradicts the fact that $\mathrm{trcl}(R_T)$ is asymmetric.

Case 3: $\mathcal{X}^0 = \{q, q'\}$ for some $q \neq q' \in \mathcal{X}$.
Note first that $y_{\{q\}} = y_{\{q'\}} = 0$ and $y_{\{p\}} = 1$ for every $p \in \mathcal{X} \setminus \{q, q'\}$. We claim that $\emptyset \notin \mathcal{L}$. Assume for the sake of contradiction that $\emptyset \in \mathcal{L}$. Then $(\emptyset, \{q\}), (\emptyset, \{q'\}) \in R_T$ since $\emptyset$ is consistent with the teaching sets for instances from $\mathcal{X}^0$. But then, no matter how x in $T(\emptyset) = \{(x, 0)\}$ is chosen, at least one of the sets $\{q\}$ and $\{q'\}$ will be consistent with $T(\emptyset)$ so that at least one of the pairs $(\{q\}, \emptyset)$ and $(\{q'\}, \emptyset)$ belongs to R_T. This contradicts the fact that R_T must be asymmetric. Thus $\emptyset \notin \mathcal{L}$, indeed. Now it suffices to show that $\mathcal{L}$ cannot contain a concept of size at least 2 that contains an element of $\mathcal{X} \setminus \{q, q'\}$. Assume for the sake of contradiction that there is a set $L \in \mathcal{L}$ such that $|L| \geq 2$ and $p \in L$ for some $p \in \mathcal{X} \setminus \{q, q'\}$. Observe that $(L, \{p\}) \in R_T$. Another application of the first assertion in Lemma 33 shows that $y_L = 1$ (because $y_{\{p\}} = 1$ and $p \in L$) and $x_L \in \{q, q'\}$ (because the other 1-labeled instances are already in use for teaching the corresponding singletons). It follows that one of the pairs $(\{q\}, L)$ and $(\{q'\}, L)$ belongs to R_T. The third assertion of Lemma 33 implies that $T(q) = \{(q', 0)\}$ or $T(q') = \{(q, 0)\}$. For reasons of symmetry, we may assume that $T(q) = \{(q', 0)\}$. This implies that $(\{p\}, \{q\}) \in R_T$. Let q'' be given by $T(q') = \{(q'', 0)\}$. Note that either $q'' = q$ or $q'' \in \mathcal{X} \setminus \{q, q'\}$. In the former case, we have that $(\{p\}, \{q'\}) \in R_T$ and in the latter case we have that $(\{q\}, \{q'\}) \in R_T$. Since $(\{p\}, \{q\}) \in R_T$ (which was observed above already), we conclude that in both cases, $(\{p\}, \{q\}), (\{p\}, \{q'\}) \in \mathrm{trcl}(R_T)$. Combining this with our observations above that $(L, \{p\}) \in R_T$ and that one of the pairs $(\{q\}, L)$ and $(\{q'\}, L)$ belongs to R_T, yields a contradiction to the fact that $\mathrm{trcl}(R_T)$ is asymmetric.

■

Corollary 35 *Let $\mathcal{L} \subseteq 2^{\mathcal{X}}$ be a concept class that contains $\mathrm{Sg}(\mathcal{X})$. If $\mathrm{PBTD}(\mathcal{L}) = 1$, then $\mathrm{RTD}(\mathcal{L}) = 1$.*

Proof According to Theorem 34, either L coincides with $\mathrm{Sg}(\mathcal{X})$ or $\mathcal{L}$ contains precisely one additional concept that is $\emptyset$ or a set of size 2. The partial ordering $\prec$ on $\mathcal{L}$ that is used in the first part of the proof of Theorem 34 (proof direction "$\Leftarrow$") is easily compiled into a recursive teaching plan of order 1 for $\mathcal{L}$.[6] ■

The characterizations proven above can be applied to certain geometric concept classes.

Consider a class $\mathcal{L}$, consisting of bounded and topologically closed objects in the d-dimensional Euclidean space, that satisfies the following condition: for every pair $(A, B) \in \mathbb{R}^d$, there is exactly one object in $\mathcal{L}$, denoted as $L_{A,B}$ in the sequel, such that $A, B \in L$ and such that $\|A - B\|$ coincides with the diameter of L. This assumption implies that $|\mathcal{L} \setminus \mathrm{Sg}(\mathbb{R}^d)| = \infty$. By setting $A = B$, it furthermore implies $\mathrm{Sg}(\mathbb{R}^d) \subseteq \mathcal{L}$. Let us prefer objects with a small diameter over objects with a larger diameter. Then, obviously, $\{A, B\}$ is a positive teaching set for $L_{A,B}$. Because

6. This also follows from Lemma 8 and the fact that there are no chains of a length exceeding 2 in $(\mathcal{L}, \prec)$.

of $|\mathcal{L} \setminus \mathrm{Sg}(\mathbb{R}^d)| = \infty$, $\mathcal{L}$ does clearly not satisfy the condition in Theorem 34, which is necessary for $\mathcal{L}$ to have a PBTD of 1. We may therefore conclude that $\mathrm{PBTD}(\mathcal{L}) = \mathrm{PBTD}^+(\mathcal{L}) = 2$.

The family of classes with the required properties is rich and includes, for instance, the class of d-dimensional balls as well as the class of d-dimensional axis-parallel rectangles.

9. Conclusions

Preference-based teaching uses the natural notion of preference relation to extend the classical teaching model. The resulting model is (i) more powerful than the classical one, (ii) resolves difficulties with the recursive teaching model in the case of infinite concept classes, and (iii) is at the same time free of coding tricks even according to the definition by Goldman and Mathias (1996). Our examples of algebraic and geometric concept classes demonstrate that preference-based teaching can be achieved very efficiently with naturally defined teaching sets and based on intuitive preference relations such as inclusion. We believe that further studies of the PBTD will provide insights into structural properties of concept classes that render them easy or hard to learn in a variety of formal learning models.

We have shown that spanning sets lead to a general-purpose construction for preference-based teaching sets of only positive examples. While this result is fairly obvious, it provides further justification of the model of preference-based teaching, since the teaching sets it yields are often intuitively exactly those a teacher would choose in the classroom (for instance, one would represent convex polygons by their vertices, as in Example 3). It should be noted, too, that it can sometimes be difficult to establish whether the upper bound on PBTD obtained this way is tight, or whether the use of negative examples or preference relations other than inclusion yield smaller teaching sets. Generally, the choice of preference relation provides a degree of freedom that increases the power of the teacher but also increases the difficulty of establishing lower bounds on the number of examples required for teaching.

Acknowledgements. Sandra Zilles was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), in the Discovery Grant and Canada Research Chairs programs. We thank the anonymous referees for their numerous thoughtful comments, which greatly helped to improve the presentation of the paper.

References

N. Abe. Polynomial learnability of semilinear sets. In *Proceedings of the 2nd Annual Conference on Learning Theory (COLT)*, pages 25–40, 1989.

D. Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45(2):117–135, 1980.

X. Chen, Y. Cheng, and B. Tang. On the recursive teaching dimension of vc classes. In *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, pages 2164–2171, 2016.

T. Doliwa, G. Fan, H. U. Simon, and S. Zilles. Recursive teaching dimension, VC-dimension, and sample compression. *Journal of Machine Learning Research*, 15:3107–3131, 2014.

Z. Gao, H. U. Simon, and S. Zilles. On the teaching complexity of linear sets. In *Proceedings of the 26th International Conference on Algorithmic Learning Theory (ALT)*, pages 102–116, 2015.

Z. Gao, C. Ries, H. U. Simon, and S. Zilles. Preference-based teaching. In *Proceedings of the 29th Conference on Learning Theory (COLT)*, pages 971–997, 2016.

Z. Gao, C. Ries, H. U. Simon, and S. Zilles. Preference-based teaching. *ArXiv e-prints*, February 2017. URL `https://arxiv.org/pdf/1702.02047.pdf`.

S. A. Goldman and M. J. Kearns. On the complexity of teaching. *Journal of Computer and System Sciences*, 50:20–31, 1995.

S. A. Goldman and H. D. Mathias. Teaching a smarter learner. *Journal of Computer and System Sciences*, 52:255–267, 1996.

D. Helmbold, R. Sloan, and M. K. Warmuth. Learning nested differences of intersection-closed concept classes. *Machine Learning*, pages 165–196, 1990.

T. J. Jech. *The Axiom of Choice*. North-Holland Pub. Co., Amsterdam, 1973.

Z. Mazadi, Z. Gao, and S. Zilles. Distinguishing pattern languages with membership examples. In *Proceedings of the 8th International Conference on Language and Automata Theory and Applications (LATA)*, pages 528–540, 2014.

S. Moran, A. Shpilka, A. Wigderson, and A. Yehudayoff. Compressing and teaching for low VC-dimension. In *Proceedings of the 56th Annual Symposium on the Foundations of Computer Science (FOCS)*, pages 40–51, 2015.

R. J. Parikh. On context-free languages. *Journal of the ACM*, 13(4):570–581, 1966.

J. G. Rosales and P. A. García-Sánchez. *Numerical Semigroups*. Springer, 2009.

A. Shinohara and S. Miyano. Teachability in computational learning. *New Generation Computing*, 8(4):337–347, 1991.

H. U. Simon and S. Zilles. Open problem: Recursive teaching dimension versus VC dimension. In *Proceedings of the 28th Annual Conference on Learning Theory (COLT)*, pages 1770–1772, 2015.

Y. Takada. Learning semilinear sets from examples and via queries. *Theoretical Computer Science*, 104(2):207–233, 1992.

S. Zilles, S. Lange, R. Holte, and M. Zinkevich. Models of cooperative teaching and learning. *Journal of Machine Learning Research*, 12:349–384, 2011.

Appendix A. Proof of Theorem 18

In Section A.1, we present a general result which helps to verify the upper bounds in Theorem 18. These upper bounds are then derived in Section A.2. Section A.3 is devoted to the derivation of the lower bounds.

A.1 The Shift Lemma

In this section, we assume that $\mathcal{L}$ is a concept class over a universe $\mathcal{X} \in \{\mathbb{N}_0, \mathbb{Q}_0^+, \mathbb{R}_0^+\}$. We furthermore assume that 0 is contained in every concept $L \in \mathcal{L}$. We can extend $\mathcal{L}$ to a larger class, namely the shift-extension $\mathcal{L}'$ of $\mathcal{L}$, by allowing each of its concepts to be shifted by some constant which is taken from $\mathcal{X}$:

$$\mathcal{L}' = \{c + L : \ (c \in \mathcal{X}) \wedge (L \in \mathcal{L})\} \ .$$

The next result states that this extension has little effect only on the complexity measures PBTD and PBTD^+:

Lemma 36 (Shift Lemma) *With the above notation and assumptions, the following holds:*

$$\mathrm{PBTD}(\mathcal{L}) \leq \mathrm{PBTD}(\mathcal{L}') \leq 1{+}\mathrm{PBTD}(\mathcal{L}) \text{ and } \mathrm{PBTD}^+(\mathcal{L}) \leq \mathrm{PBTD}^+(\mathcal{L}') \leq 1{+}\mathrm{PBTD}^+(\mathcal{L}) \ .$$

Proof It suffices to verify the inequalities $\mathrm{PBTD}(\mathcal{L}') \leq 1 + \mathrm{PBTD}(\mathcal{L})$ and $\mathrm{PBTD}^+(\mathcal{L}') \leq 1 + \mathrm{PBTD}^+(\mathcal{L})$ because the other inequalities hold by virtue of monotonicity. Let T be an admissible mapping for $\mathcal{L}$. It suffices to show that T can be transformed into an admissible mapping T' for $\mathcal{L}'$ such that $\mathrm{ord}(T') \leq 1 + \mathrm{ord}(T)$ and such that T' is positive provided that T is positive. To this end, we define T' as follows:

$$T'(c + L) = \{(c, +)\} \cup \{(c + x, b) : \ (x, b) \in T(L)\} \ .$$

Obviously $\mathrm{ord}(T') \leq 1{+}\mathrm{ord}(T)$. Note that $c \in c{+}L$ because of our assumption that 0 is contained in every concept in $\mathcal{L}$. Moreover, since the admissibility of T implies that L is consistent with $T(L)$, the above definition of $T'(c{+}L)$ makes sure that $c{+}L$ is consistent with $T'(c{+}L)$. It suffices therefore to show that the relation $\mathrm{trcl}(R_{T'})$ is asymmetric. Consider a pair $(c'{+}L', c{+}L) \in R_{T'}$. By the definition of $R_{T'}$, it follows that $c'{+}L'$ is consistent with $T'(c{+}L)$. Because of $(c, +) \in T'(c{+}L)$, we must have $c' \leq c$. Suppose that $c' = c$. In this case, L' must be consistent with $T(L)$. Thus $L' \prec_T L$. This reasoning implies that $(c' + L', c + L) \in R_{T'}$ can happen only if either $c' < c$ or $(c' = c) \wedge (L' \prec_T L)$. Since $\prec_T$ is asymmetric, we may now conclude that $\mathrm{trcl}(R_{T'})$ is asymmetric, as desired. Finally note that, according to our definition above, the mapping T' is positive provided that T is positive. This concludes the proof. ■

A.2 The Upper Bounds in Theorem 18

We remind the reader that the equality $\mathrm{PBTD}^+(\mathrm{LINSET}_k) = k$ was stated in Example 2. We will show in Lemma 37 that $\mathrm{PBTD}^+(\text{NE-LINSET}_k) \leq k$. In combination with the Shift Lemma, this implies that $\mathrm{PBTD}^+(\mathrm{LINSET}'_k) \leq k + 1$ and $\mathrm{PBTD}^+(\text{NE-LINSET}'_k) \leq k + 1$. All remaining upper bounds in Theorem 18 follow now by virtue of monotonicity.

Lemma 37 $\mathrm{PBTD}^+(\text{NE-LINSET}_k) \leq k$.

Proof We want to show that there is a preference relation for which k positive examples suffice to teach any concept in NE-LINSET_k. To this end, let $G = \{g_1, \ldots, g_\ell\}$ be a generator set with $\ell \leq k$ where $g_1 < \ldots < g_\ell$. We use $\mathrm{sum}(G) = g_1 + \ldots + g_\ell$ to denote the sum of all generators in G. We say that g_i is a *redundant generator* in G if $g_i \in \langle\{g_1, \ldots, g_{i-1}\}\rangle$. Let $G^* = \{g_1^*, \ldots, g_{\ell^*}^*\} \subseteq G$

with $g_1^* < \ldots < g_{\ell^*}^*$ be the set of non-redundant generators in G and let $\mathrm{tuple}(G) = (g_1^*, \ldots, g_{\ell^*}^*)$ be the corresponding ordered sequence. Then G^* is an independent subset of G generating the same linear set as G when allowing zero coefficients, i.e., we have $\langle G^* \rangle = \langle G \rangle$ (although $\langle G^* \rangle_+ \neq \langle G \rangle_+$ whenever G^* is a proper subset of G).

To define a suitable preference relation, let $G, \widehat{G}$ be generator sets of size k or less with $\mathrm{tuple}(G) = (g_1^*, \ldots, g_{\ell^*}^*)$ and $\mathrm{tuple}(\widehat{G}) = (\widehat{g}_1^*, \ldots, \widehat{g}_{\widehat{\ell}^*}^*)$. Let the student prefer G over $\widehat{G}$ if any of the following conditions is satisfied:

Condition 1: $\mathrm{sum}(G) > \mathrm{sum}(\widehat{G})$.

Condition 2: $\mathrm{sum}(G) = \mathrm{sum}(\widehat{G})$ and $\mathrm{tuple}(G)$ is lexicographically greater than $\mathrm{tuple}(\widehat{G})$ without having $\mathrm{tuple}(\widehat{G})$ as prefix.

Condition 3: $\mathrm{sum}(G) = \mathrm{sum}(\widehat{G})$ and $\mathrm{tuple}(G)$ is a proper prefix of $\mathrm{tuple}(\widehat{G})$.

To teach a concept $\langle G \rangle \in \text{NE-LINSET}_k$ with $\mathrm{sum}(G) = g$ and $\mathrm{tuple}(G) = (g_1^*, \ldots, g_{\ell^*}^*)$, one uses the teaching set

$$S = \{(g, +), (g + g_1^*, +), \ldots, (g + g_{h^*}^*, +)\}$$

where

$$h = \begin{cases} \ell^* - 1 & \text{if } G^* = G \\ \ell^* & \text{if } G^* \subset G \end{cases} . \tag{12}$$

Note that S contains at most $|G| \leq k$ examples. Let $\widehat{G}$ with $\left\langle \widehat{G} \right\rangle_+ \in \text{NE-LINSET}_k$ denote the generator set that is returned by the student. Clearly $\left\langle \widehat{G} \right\rangle$ satisfies $\mathrm{sum}(\widehat{G}) = g$ since

- concepts with larger generator sums are inconsistent with $(g, +)$, and
- concepts with smaller generator sums have a lower preference (compare with Condition 1 above).

It follows that $g + g_i^* \in \left\langle \widehat{G} \right\rangle_+$ is equivalent to $g_i^* \in \left\langle \widehat{G} \right\rangle = \left\langle \widehat{G}^* \right\rangle$. We conclude that the smallest generator in $\mathrm{tuple}(\widehat{G})$ equals g_1^* since

- a smallest generator in $\mathrm{tuple}(\widehat{G})$ that is greater than g_1^* would cause an inconsistency with $(g + g_1^*, +)$, and
- a smallest generator in $\mathrm{tuple}(\widehat{G})$ that is smaller than g_1^* would have a lower preference (compare with Condition 2 above).

Assume inductively that the $i-1$ smallest generators in $\mathrm{tuple}(\widehat{G})$ are $g_1^*, \ldots, g_{i-1}^*$. Since $g_i^* \notin \left\langle \{g_1^*, \ldots, g_{i-1}^*\} \right\rangle$, we may apply a reasoning that is similar to the above reasoning concerning g_1^* and conclude that the i'th smallest generator in $\mathrm{tuple}(\widehat{G})$ equals g_i^*. The punchline of this discussion is that the sequence $\mathrm{tuple}(\widehat{G})$ starts with $g_1^*, \ldots, g_h^*$ with h given by (12). Let $G' = G \setminus G^*$ be the set of redundant generators in G and note that

$$g - \sum_{i=1}^{h} g_i^* = \begin{cases} g_{\ell^*}^* & \text{if } G^* = G \\ \sum_{g' \in G'} g' & \text{if } G^* \subset G \end{cases} .$$

Let $\widehat{G}' = \widehat{G} \setminus \{g_1^*, \ldots, g_h^*\}$. We proceed by case analysis:

Case 1: $G^* = G$.
Since $\widehat{G}$ is consistent with $(g, +)$, we have $\sum_{g' \in \widehat{G}'} g' = g_{\ell^*}^*$. Since $g_{\ell^*}^* \notin \left\langle \{g_1^*, \ldots, g_{\ell^*-1}^*\} \right\rangle$, the set $\widehat{G}'$ must contain an element that cannot be generated by $g_1^*, \ldots, g_{\ell^*-1}^*$. Given the preferences of the student (compare with Condition 2), she will choose $\widehat{G}' = \{g_{\ell^*}^*\}$. It follows that $\widehat{G} = G$.

Case 2: $G^* \subset G$.
Here, we have $\sum_{g' \in \widehat{G}'} g' = \sum_{g' \in G'} g'$. Given the preferences of the student (compare with Condition 3), she will choose $\widehat{G}$ such that $\widehat{G}^* = G^*$ and $\widehat{G}'$ consists of elements from $\langle G^* \rangle$ that sum up to $\sum_{g' \in G'} g'$ (with $\widehat{G}' = \left\{ \sum_{g' \in G'} g' \right\}$ among the possible choices). Clearly, $\left\langle \widehat{G} \right\rangle_+ = \langle G \rangle_+$.

Thus, in both cases, the student comes up with the right hypothesis. ■

A.3 The Lower Bounds in Theorem 18

The lower bounds in Theorem 18 are an immediate consequence of the following result:

Lemma 38 *The following lower bounds are valid:*

$$\begin{aligned} \mathrm{PBTD}^+(\text{NE-CF-LINSET}'_k) &\geq k+1 . && (13) \\ \mathrm{PBTD}(\text{NE-CF-LINSET}'_k) &\geq k-1 . && (14) \\ \mathrm{PBTD}(\text{NE-CF-LINSET}_k) &\geq \frac{k-1}{2} . && (15) \\ \mathrm{PBTD}(\text{CF-LINSET}_k) &\geq k-1 . && (16) \end{aligned}$$

This lemma can be seen as an extension and a strengthening of a similar result in (Gao et al., 2015) where the following lower bounds were shown:

$$\begin{aligned} \mathrm{RTD}^+(\text{NE-LINSET}'_k) &\geq k+1 . \\ \mathrm{RTD}(\text{NE-LINSET}'_k) &\geq k-1 . \\ \mathrm{RTD}(\text{CF-LINSET}_k) &\geq k-1 . \end{aligned}$$

The proof of Lemma 38 builds on some ideas that are found in (Gao et al., 2015) already, but it requires some elaboration to obtain the stronger results.

We now briefly explain why the lower bounds in Theorem 18 directly follow from Lemma 38. Note that the lower bound $k-1$ in (8) is immediate from (14) and a monotonicity argument. This is because $\text{NE-LINSET}'_k \supseteq \text{NE-CF-LINSET}'_k$ as well as $\text{LINSET}'_k \supseteq \text{CF-LINSET}'_k \supseteq \text{NE-CF-LINSET}'_k$. Note furthermore that $\mathrm{PBTD}^+(\text{CF-LINSET}'_k) \geq k+1$ because of (13) and a monotonicity argument. Then the Shift Lemma implies that $\mathrm{PBTD}^+(\text{CF-LINSET}_k) \geq k$. All remaining lower bounds in Theorem 18 are obtained from these observations by virtue of monotonicity.

The proof of Theorem 18 can therefore be accomplished by proving Lemma 38. It turns out that the proof of this lemma is quite involved. We will present in Section A.3.1 some theoretical prerequisites. Sections A.3.2 and A.3.3 are devoted to the actual proof of the lemma.

A.3.1 SOME BASIC CONCEPTS IN THE THEORY OF NUMERICAL SEMIGROUPS

Recall from Section 6 that $\langle G\rangle = \left\{\sum_{g\in G} a(g)g : a(g)\in\mathbb{N}_0\right\}$. The elements of G are called *generators* of $\langle G\rangle$. A set $P\subset\mathbb{N}$ is said to be *independent* if none of the elements in P can be written as a linear combination (with coefficients from $\mathbb{N}_0$) of the remaining elements (so that $\langle P'\rangle$ is a proper subset of $\langle P\rangle$ for every proper subset P' of P). It is well known (Rosales and García-Sánchez, 2009) that independence makes generating systems unique, i.e., if P, P' are independent, then $\langle P\rangle = \langle P'\rangle$ implies that $P = P'$. Moreover, for every independent set P, the following implication is valid:

$$(S\subseteq\langle P\rangle \wedge P\not\subseteq S) \Rightarrow (\langle S\rangle\subset\langle P\rangle)\ . \tag{17}$$

Let $P=\{a_1,\ldots,a_k\}$ be independent with $a_1=\min P$. It is well known[7] and easy to see that the residues of $a_1, a_2,\ldots,a_k$ modulo a_1 must be pairwise distinct (because, otherwise, we would obtain a dependence). If a_1 is a prime and $|P|\geq 2$, then the independence of P implies that $\gcd(P)=1$. Thus the following holds:

Lemma 39 *If $P\subset\mathbb{N}$ is an independent set of cardinality at least 2 and $\min P$ is a prime, then $\gcd(P)=1$.*

In the remainder of the paper, the symbols P and P' are reserved for denoting independent sets of generators.

It is well known that $\langle G\rangle$ is co-finite iff $\gcd(G)=1$ (Rosales and García-Sánchez, 2009). Let P be a finite (independent) subset of $\mathbb{N}$ such that $\gcd(P)=1$. The largest number in $\mathbb{N}\setminus\langle P\rangle$ is called the *Frobenius number of* P and is denoted as $F(P)$. It is well known (Rosales and García-Sánchez, 2009) that

$$F(\{p,q\}) = pq-p-q \tag{18}$$

provided that $p,q\geq 2$ satisfy $\gcd(p,q)=1$.

A.3.2 PROOF OF (13)

The shift-extension of NE-CF-LINSET$_k$ is (by way of definition) the following class:

$$\text{NE-CF-LINSET}'_k = \{c+\langle P\rangle_+ :\ (c\in\mathbb{N}_0)\wedge(P\subset\mathbb{N})\wedge(|P|\leq k)\wedge(\gcd(P)=1)\}\ . \tag{19}$$

It is easy to see that this can be written alternatively in the form

$$\text{NE-CF-LINSET}'_k = \left\{N+\langle P\rangle :\ N\in\mathbb{N}_0\wedge P\subset\mathbb{N}\wedge|P|\leq k\wedge\gcd(P)=1\wedge\sum_{p\in P}p\leq N\right\} \tag{20}$$

where N in (20) corresponds to $c+\sum_{p\in P}p$ in (19).

For technical reasons, we define the following subfamilies of NE-CF-LINSET$'_k$. For each $N\geq 0$, let

$$\text{NE-CF-LINSET}'_k[N] = \{N+L : L\in\text{LINSET}_k[N]\}$$

7. E.g., see (Rosales and García-Sánchez, 2009)

where

$$\text{LINSET}_k[N] = \left\{\langle P\rangle\in\text{LINSET}_k : (\gcd(P)=1)\wedge\left(\sum_{p\in P}p\leq N\right)\right\}\ .$$

In other words, NE-CF-LINSET$'_k[N]$ is the subclass consisting of all concepts in NE-CF-LINSET$'_k$ (written in the form (20)) whose constant is N.
A central notion for proving (13) is the following one:

Definition 40 *Let $k,N\geq 2$ be integers. We say that a set $L\in$ NE-CF-LINSET$'$ is (k,N)-*special *if it is of the form $L=N+\langle P\rangle$ such that the following holds:*

1. *P is an independent set of cardinality k and $\min P$ is a prime (so that $\gcd(P)=1$ according to Lemma 39, which furthermore implies that $\langle P\rangle$ is co-finite).*
2. *Let $q(P)$ denote the smallest prime that is greater than $F(P)$ and greater than $\max P$. For $a=\min P$ and $r=0,\ldots,a-1$, let*

$$t_r(P)=\min\{s\in\langle P\rangle : s\equiv r \pmod a\} \text{ and } t_{max}(P)=\max_{0\leq r\leq a-1}t_r(P)\ .$$

Then

$$N\geq k(a+t_{max}(P)) \text{ and } N\geq q(P)+\sum_{p\in P\setminus\{a\}}p\ . \tag{21}$$

We need at least k positive examples in order to distinguish a (k,N)-special set from all its proper subsets in NE-CF-LINSET$'_k[N]$, as the following result shows:

Lemma 41 *For all $k\geq 2$, the following holds. If $L\in$ NE-CF-LINSET$'$ is (k,N)-special, then $L\in$ NE-CF-LINSET$'[N]$ and $I'(L,\text{NE-CF-LINSET}_k[N])\geq k$.*

Proof Suppose that $L=N+\langle P\rangle$ is of the form as described in Definition 40. Let $P=\{a,a_2\ldots,a_k\}$ with $a=\min P$. For the sake of simplicity, we will write t_r instead of $t_r(P)$ and t_{max} instead of $t_{max}(P)$. The independence of P implies that $t_{a_i \bmod a}=a_i$ for $i=2,\ldots,k$. It follows that $t_{max}\geq\max P$. Since, by assumption, $N\geq k\cdot t_{max}$, it becomes obvious that $L\in$ NE-CF-LINSET$'[N]$.

Assume by way of contradiction that the following holds:

(A) There is a weak spanning set S of size $k-1$ for L w.r.t. NE-CF-LINSET$'_k[N]$.

Since N is contained in any concept from NE-CF-LINSET$'_k[N]$, we may assume that $N\notin S$ so that S is of the form $S=\{N+x_1,\ldots,N+x_{k-1}\}$ for integers $x_i\geq 1$. For $i=1,\ldots,k-1$, let $r_i=x_i \bmod a\in\{0,1,\ldots,a-1\}$. It follows that each x_i is of the form $x_i=q_ia+t_{r_i}$ for some integer $q_i\geq 0$. Let $X=\{x_1,\ldots,x_{k-1}\}$. We proceed by case analysis:

Case 1: $X\subseteq\{a_2,\ldots,a_k\}$ (so that, in view of $|X|=k-1$, we even have $X=\{a_2,\ldots,a_k\}$). Let $L'=N+\langle X\rangle$. Then $S\subseteq L'$. Note that $X\subseteq P$ but $P\not\subseteq X$. We may conclude from (17) that $\langle X\rangle\subset\langle P\rangle$ and, therefore, $L'\subset L$. Thus L' is a proper subset of L which contains S. Note that (21) implies that $N\geq\sum_{i=2}^{k}a_i=\sum_{i=1}^{k-1}x_i$. If $\gcd(X)=1$, then $L'\in$ NE-CF-LINSET$[N]$ and we have an immediate contradiction to the above assumption (A).

Otherwise, if $\gcd(X) \geq 2$, then we define $L'' = N + \langle X \cup \{q(P)\}\rangle$. Note that $S \subseteq L' \subseteq L''$. Since $q(P) > F(P)$, we have $X \cup \{q(P)\} \subseteq \langle P\rangle$ and, since $q(P) > \max P$, we have $P \not\subseteq X \cup \{q(P)\}$. We may conclude from (17) that $\langle X \cup \{q(P)\}\rangle \subset \langle P\rangle$ and, therefore, $L'' \subset L$. Thus, L'' is a proper subset of L which contains S. Because $X = \{a_2, \ldots, a_k\}$ and $q(P)$ is a prime that is greater than $\max P$, it follows that $\gcd(X \cup \{q(P)\}) = 1$. In combination with (21), it easily follows now that $L'' \in$ NE-CF-LINSET$[N]$. Putting everything together, we arrive at a contradiction to the assumption (A).

Case 2: $X \not\subseteq \{a_2, \ldots, a_k\}$.

If $r_i = 0$ for $i = 1, \ldots, k-1$, then each x_i is a multiple of a. In this case, $N + \langle a, q(P)\rangle$ is a proper subset of $L = N + \langle P\rangle$ that is consistent with S, which yields a contradiction. We may therefore assume that there exists $i' \in \{1, \ldots, k-1\}$ such that $r_{i'} \neq 0$. From the case assumption, $X \not\subseteq \{a_2, \ldots, a_k\}$, it follows that there must exist an index $i'' \in \{1, \ldots, k-1\}$ such that $q_{i''} \geq 1$ or $t_{r_{i''}} \notin \{a_2, \ldots, a_k\}$. For $i = 1, \ldots, k-1$, let $q'_i = \min\{q_i, 1\}$ and $x'_i = q'_i a + t_{r_i}$. Note that $q'_{i''} = 1$ iff $q_{i''} \geq 1$. Define $L'' = N + \langle X'\rangle$ for $X' = \{a, x'_1, \ldots, x'_{k-1}\}$ and observe the following. First, the set L'' clearly contains S. Second, the choice of $x'_1, \ldots, x'_{k-1}$ implies that $X' \subseteq \langle P\rangle$. Third, it easily follows from $q'_{i''} = 1$ or $t_{r_{i''}} \notin \{a_2, \ldots, a_k\}$ that $P \not\subseteq \{a, x'_1, \ldots, x'_{k-1}\}$. We may conclude from (17) that $\langle X'\rangle \subset \langle P\rangle$ and, therefore, $L'' \subset L$. Thus, L'' is a proper subset of L which contains S. Since $r_{i'} \neq 0$ and a is a prime, it follows that $\gcd(a, x'_{i'}) = 1$ and, therefore, $\gcd(X') = 1$. In combination with (21), it easily follows now that $L'' \in$ NE-CF-LINSET$[N]$. Putting everything together, we obtain again a contradiction to the assumption (A).

■

For the sake of brevity, let $\mathcal{L} =$ NE-CF-LINSET$'$. Assume by way of contradiction that there exists a positive mapping T of order k that is admissible for $\mathcal{L}_k$. We will pursue the following strategy:

1. We define a set $L \in \mathcal{L}_k$ of the form $L = N + p + \langle 1\rangle$.
2. We define a second set $L' = N + \langle G\rangle \in \mathcal{L}$ that is (k, N)-special and consistent with $T^+(L)$. Moreover, $L' \setminus L = \{N\}$.

If this can be achieved, then the proof will be accomplished as follows:

- According to Lemma 41, $T^+(L')$ must contain at least k examples (all of which are different from N) for distinguishing L' from all its proper subsets in $\mathcal{L}_k[N]$.
- Since L' is consistent with $T^+(L)$, the set $T^+(L')$ must contain an example which distinguishes L' from L. But the only example which fits this purpose is $(N, +)$.
- The discussion shows that $T^+(L')$ must contain k examples in order to distinguish L' from all its proper subsets in $\mathcal{L}_k$ plus one additional example, N, needed to distinguish L' from L.
- We obtain a contradiction to our initial assumption that T^+ is of order k.

We still have to describe how our proof strategy can actually be implemented. We start with the definition of L. Pick the smallest prime $p \geq k+1$. Then $\{p, p+1, \ldots, p+k\}$ is independent. Let $M = F(\{p, p+1\}) \overset{(18)}{=} p(p+1) - p - (p+1)$. An easy calculation shows that $k \geq 2$ and $p \geq k+1$ imply that $M \geq p+k$. Let $I = \{p, p+1, \ldots, M\}$. Choose N large enough so that all concepts of the form

$$N + \langle P\rangle \text{ where } |P| = k,\ p = \min P \text{ and } P \subseteq I$$

are (k, N)-special. With these choices of p and N, let $L = N + p + \langle 1\rangle$. Note that $N+p, N+p+1 \in T^+(L)$ because, otherwise, one of the concepts $N + p + 1 + \langle 1\rangle, N + p + \langle 2, 3\rangle \subset L$ would be consistent with $T^+(L)$ whereas $T^+(L)$ must distinguish L from all its proper subsets in $\mathcal{L}_k$. Setting $A = \{x : N + x \in T^+(L)\}$, it follows that $|A| = |T^+(L)| \leq k$ and $p, p+1 \in A$. The set A is not necessarily independent but it contains an independent subset B such that $p, p+1 \in B$ and $\langle A\rangle = \langle B\rangle$. Since $M = F(\{p, p+1\})$, it follows that any integer greater than M is contained in $\langle p, p+1\rangle$. Since B is an independent extension of $\{p, p+1\}$, it cannot contain any integer greater than M. It follows that $B \subseteq I$. Clearly, $|B| \leq k$ and $\gcd(B) = 1$. We would like to transform B into another generating system $G \subseteq I$ such that

$$\langle B\rangle \subseteq \langle G\rangle, \gcd(G) = 1 \text{ and } |G| = k\ .$$

If $|B| = k$, we can simply set $G = B$. If $|B| < k$, then we make use of the elements in the independent set $\{p, p+1, \ldots, p+k\} \subseteq I$ and add them, one after the other, to B (thereby removing other elements from B whenever their removal leaves $\langle B\rangle$ invariant) until the resulting set G contains k elements. We now define the set L' by setting $L' = N + \langle G\rangle$. Since $G \subseteq I = \{p, p+1, \ldots, M\}$, and $p, p+1 \in G$, it follows that $p = \min G$, $\gcd(G) = 1$ and $\min(L' \setminus \{N\})$ is $N + p$. Thus, $L' \setminus L = \{N\}$, as desired. Moreover, since N had been chosen large enough, the set L' is (k, N)-special. Thus L and L' have all properties that are required by our proof strategy and the proof of (13) is complete.

A.3.3 PROOF OF (14), (15), AND (16)

We make use of some well known (and trivial) lower bounds on TD_{min}:

Example 5 *For every $k \in \mathbb{N}$, let $[k] = \{1, 2, \ldots, k\}$, let $2^{[k]}$ denote the powerset of $[k]$ and, for all $\ell = 0, 1, \ldots, k$, let*

$$\binom{[k]}{\ell} = \{S \subseteq [k] :\ |S| = \ell\}$$

denote the class of those subsets of $[k]$ that have exactly ℓ elements. It is trivial to verify that

$$\mathrm{TD}_{min}\left(2^{[k]}\right) = k \text{ and } \mathrm{TD}_{min}\left(\binom{[k]}{\ell}\right) = \min\{\ell, k-\ell\}\ .$$

In view of $\mathrm{PBTD}^+(\mathrm{LINSET}_k) = k$, the next results show that negative examples are of limited help only as far as preference-based teaching of concepts from LINSET_k is concerned:

Lemma 42 *For every $k \geq 1$ and for all $\ell = 0, \ldots, k-1$, let*

$$\begin{aligned} \mathcal{L}_k &= \{\langle k, p_1, \ldots, p_{k-1}\rangle : \ p_i \in \{k+i, 2k+i\}\} \ , \\ \mathcal{L}_{k,\ell} &= \{\{\langle k, p_1, \ldots, p_{k-1}\rangle \in \mathcal{L}_k : \ |\{i : p_i = k+i\}| = \ell\} \ . \end{aligned}$$

With this notation, the following holds:

$$\mathrm{TD}_{min}(\mathcal{L}_k) \geq k-1 \ \textit{and} \ \mathrm{TD}_{min}(\mathcal{L}_{k,\ell}) \geq \min\{\ell, k-1-\ell\} \ .$$

Proof For $k = 1$, the assertion in the lemma is vacuous. Suppose therefore that $k \geq 2$. An inspection of the generators $k, p_1, \ldots, p_{k-1}$ with $p_i \in \{k+i, 2k+i\}$ shows that

$$\begin{aligned} \mathcal{L}_k &= \{L_{k,S} : \ S \subseteq \{k+1, k+2, \ldots, 2k-1\}\} \\ \mathcal{L}_{k,\ell} &= \{L_{k,S} : \ (S \subseteq \{k+1, k+2, \ldots, 2k-1\}) \wedge (|S| = \ell)\} \end{aligned}$$

where

$$L_{k,S} = \{0, k\} \cup \{2k, 2k+1, \ldots\} \cup S \ .$$

Note that the examples in $\{0, 1, \ldots, k\} \cup \{2k, 2k+1, \ldots, \}$ are redundant because they do not distinguish between distinct concepts from $\mathcal{L}_k$. The only useful examples are therefore contained in the interval $\{k+1, k+2, \ldots, 2k-1\}$. From this discussion, it follows that teaching the concepts of $\mathcal{L}_k$ (resp. of $\mathcal{L}_{k,\ell}$) is not essentially different from teaching the concepts of $2^{[k-1]}$ $\left(\text{resp. of } \binom{[k-1]}{\ell}\right)$. This completes the proof of the lemma because we know from Example 5 that $\mathrm{TD}_{min}(2^{[k-1]}) = k-1$ and $\mathrm{TD}_{min}\left(\binom{[k-1]}{\ell}\right) = \min\{\ell, k-1-\ell\}$. ■

We claim now that the inequalities (14), (15) and (16) are valid, i.e., we claim that the following holds:

1. $\mathrm{PBTD}(\text{CF-LINSET}_k) \geq k-1$.
2. $\mathrm{PBTD}(\text{NE-CF-LINSET}_k) \geq \lfloor (k-1)/2 \rfloor$.
3. $\mathrm{PBTD}(\text{NE-CF-LINSET}'_k) \geq k-1$.

Proof For $k = 1$, the inequalities are obviously valid. Suppose therefore that $k \geq 2$.

1. Since $\gcd(k, k+1) = \gcd(k, 2k+1) = 1$, it follows that $\mathcal{L}_k$ is a finite subclass of CF-LINSET_k. Thus $\mathrm{PBTD}(\text{CF-LINSET}_k) \geq \mathrm{PBTD}(\mathcal{L}_k) \geq \mathrm{TD}_{min}(\mathcal{L}_k) \geq k-1$.

2. Define $\mathcal{L}_k[N] = \{N + L : \ L \in \mathcal{L}_k\}$ and $\mathcal{L}_{k,\ell}[N] = \{N + L : \ L \in \mathcal{L}_{k,\ell}\}$. Clearly $\mathrm{TD}_{min}(\mathcal{L}_k[N]) = \mathrm{TD}_{min}(\mathcal{L}_k)$ and $\mathrm{TD}_{min}(\mathcal{L}_{k,\ell}[N]) = \mathrm{TD}_{min}(\mathcal{L}_{k,\ell})$ holds for every $N \geq 0$. It follows that the lower bounds in Lemma 42 are also valid for the classes $\mathcal{L}_k[N]$ and $\mathcal{L}_{k,\ell}[N]$ in place of $\mathcal{L}_k$ and $\mathcal{L}_{k,\ell}$, respectively. Let

$$N(k) = k^2 + (k-1-\lfloor (k-1)/2 \rfloor)k + \sum_{i=1}^{k-1} i = k^2 + (k-1-\lfloor (k-1)/2 \rfloor)k + \frac{1}{2}(k-1)k \ . \quad (22)$$

It suffices to show that $N(k) + \mathcal{L}_{k,\lfloor (k-1)/2 \rfloor}$ is a finite subclass of NE-CF-LINSET_k. To this end, first note that

$$\langle k, p_1, \ldots, p_{k-1}\rangle_+ = k + \sum_{i=1}^{k-1} p_i + \langle k, p_1, \ldots, p_{k-1}\rangle \ .$$

Call p_i "light" if $p_i = k+i$ and call it "heavy" if $p_i = 2k+i$. Note that a concept L from $N(k) + \mathcal{L}_{k,\ell}$ is of the general form

$$L = N(k) + \langle k, p_1, \ldots, p_{k-1}\rangle \quad (23)$$

with exactly ℓ light parameters among $p_1, \ldots, p_{k-1}$. A straightforward calculation shows that, for $\ell = \lfloor (k-1)/2 \rfloor$, the sum $k + \sum_{i=1}^{k-1} p_i$ equals the number $N(k)$ as defined in (22). Thus, the concept L from (23) with exactly $\lfloor (k-1)/2 \rfloor$ light parameters among $\{p_1, \ldots, p_{k-1}\}$ can be rewritten as follows:

$$L = N(k) + \langle k, p_1, \ldots, p_{k-1}\rangle = \langle k, p_1, \ldots, p_{k-1}\rangle_+ \ .$$

This shows that $L \in \text{NE-CF-LINSET}_k$. As L is a concept from $N(k) + \mathcal{L}_{k,\lfloor (k-1)/2 \rfloor}$ in general form, we may conclude that $N(k) + \mathcal{L}_{k,\lfloor (k-1)/2 \rfloor}$ is a finite subclass of NE-CF-LINSET_k, as desired.

3. The proof of the third inequality is similar to the above proof of the second one. It suffices to show that, for every $k \geq 2$, there exists $N \in \mathbb{N}$ such that $N + \mathcal{L}_k$ is a subclass of $\text{NE-CF-LINSET}'_k$. To this end, we set $N = 3k^2$. A concept L from $3k^2 + \mathcal{L}_k$ is of the general form

$$L = 3k^2 + \langle k, p_1, \ldots, p_{k-1}\rangle$$

with $p_i \in \{k+i, 2k+i\}$ (but without control over the number of light parameters). It is easy to see that the constant $3k^2$ is large enough so that L can be rewritten as

$$L = 3k^2 - \left(k + \sum_{i=1}^{k-1} p_i\right) + \langle k, p_1, \ldots, p_{k-1}\rangle_+$$

where $3k^2 - \left(k + \sum_{i=1}^{k-1} p_i\right) \geq 0$. This shows that $L \in \text{NE-CF-LINSET}'_k$. As L is a concept from $3k^2 + \mathcal{L}_k$ in general form, we may conclude that $3k^2 + \mathcal{L}_k$ is a finite subclass of $\text{NE-CF-LINSET}'_k$, as desired.

■

Journal of Machine Learning Research 18 (2017) 1-40 Submitted 8/13; Revised 2/17; Published 4/17

Nonparametric Risk Bounds for Time-Series Forecasting

Daniel J. McDonald DAJMCDON@INDIANA.EDU
Department of Statistics
Indiana University
Bloomington, IN 47404

Cosma Rohilla Shalizi CSHALIZI@CMU.EDU
Mark Schervish MARK@CMU.EDU
Department of Statistics
Carnegie Mellon University
Pittsburgh, PA 15208

Editor: Mehryar Mohri

Abstract

We derive generalization error bounds for traditional time-series forecasting models. Our results hold for many standard forecasting tools including autoregressive models, moving average models, and, more generally, linear state-space models. These non-asymptotic bounds need only weak assumptions on the data-generating process, yet allow forecasters to select among competing models and to guarantee, with high probability, that their chosen model will perform well. We motivate our techniques with and apply them to standard economic and financial forecasting tools—a GARCH model for predicting equity volatility and a dynamic stochastic general equilibrium model (DSGE), the standard tool in macroeconomic forecasting. We demonstrate in particular how our techniques can aid forecasters and policy makers in choosing models which behave well under uncertainty and mis-specification.

Keywords: generalization error, prediction risk, model selection, VC dimension, state-space models, linear time-invariant systems

1. Introduction

Generalization error bounds are probabilistically valid, non-asymptotic tools for characterizing the predictive ability of forecasting models. This methodology is fundamentally about choosing particular prediction functions out of some class of plausible alternatives so that, with high reliability, the resulting predictions will be nearly as accurate as possible ("probably approximately correct"). While many of these results are aimed at classification problems with independent and identically distributed (i.i.d.) data, this paper adapts and extends these methods to time-series models, so that economic and financial forecasting techniques can be evaluated rigorously. In particular, these methods control the expected accuracy of future predictions from mis-specified models based on finite samples. This allows for immediate model comparisons which neither appeal to asymptotics nor make strong assumptions about the data-generating process, in stark contrast to such popular model-selection tools as AIC.

©2017 Daniel J. McDonald, Cosma Rohilla Shalizi, and Mark Schervish.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/13-336.html.

To fix ideas, imagine i.i.d. data $((Y_1, X_1), \ldots, (Y_n, X_n))$ with $(Y_i, X_i) \in \mathcal{Y} \times \mathcal{X}$, some prediction function $f : \mathcal{X} \to \mathcal{Y}$, and a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ which measures the cost of bad predictions. The *generalization error* or *risk* of f is

$$R(f) := \mathbb{E}[\ell(Y, f(X))],$$

where the expectation is taken with respect to $\mathbb{P}$, the joint distribution of (Y, X). The generalization error measures the inaccuracy of our predictions when we use f on future data, making it a natural criterion for model selection, and a target for performance guarantees. To actually calculate the risk, we would need to know the data-generating distribution $\mathbb{P}$ and have a single fixed prediction function f, neither of which is common. Because explicitly calculating the risk is infeasible, forecasters typically try to estimate it, which calls for detailed assumptions on $\mathbb{P}$. The alternative we employ here is to find upper bounds on risk which hold uniformly over large classes of models $\mathcal{F}$ from which some particular f is chosen, possibly in a data-dependent way, and uniformly over distributions $\mathbb{P}$.

Our main results in Section 5 assert that for wide classes of time-series models, the expected cost of poor predictions is bounded by the model's in-sample performance inflated by a term which balances the amount of observed data with the complexity of the model. The bound holds with high probability under the unknown distribution $\mathbb{P}$ assuming only mild conditions—existence of some moments, stationarity, and the decay of temporal dependence as data points become widely separated in time. We give applications in Section 6.

Our goal in this paper is to provide general bounds for common time-series models with unbounded loss functions, no explicit regularization, and potential dependence on the entirety of the observed data. The bounds we derive here are relevant for the time-series models typically used in applied settings—finance, economics, engineering, etc.—as well as covering models more common in machine learning. In particular, we derive results for non-linear models which depend only on a fixed quantity of recent data and linear time invariant systems, state-space models, which use the entire past to predict new data. These results however do not cover, e.g. HMMs in the strictest sense, as they require absolutely continuous latent states rather than discrete valued ones.

The remainder of this paper is structured as follows. Section 2 provides motivation and background for our results, giving intuition in the i.i.d. setting by focusing on concentration of measure ideas and characterizations of model complexity. Section 3 gives the explicit assumptions we make and describes how to leverage powerful ideas from time series to generalize the i.i.d. methods. Section 4 introduces linear time-invariant systems and discusses how such forecasters are different from, e.g., autoregressive models. Section 5 states and proves risk bounds for the time-series forecasting setting, while we demonstrate how to use the results in Section 6 and give some properties of those results in Section 7. Finally, Section 8 concludes and illustrates the path toward generalizing our methods to more elaborate model classes.

1.1 Related Work

Yu (1994) and Nobel and Dembo (1993) showed that it was possible to transfer some i.i.d. results to β-mixing sequences, albeit without explicit rates of convergence (c.f. Vidyasagar, 2003, §3.4). Karandikar and Vidyasagar (2002); Meir (2000); Vidyasagar (2003) presented

bounds for model classes with finite covering numbers (a sufficient condition for which is finite VC-dimension of a related class) but additionally require that $\mathcal{Y}$ is compact and ℓ is bounded. Early work in signal processing (Modha and Masry, 1998) proposes predictors based on sequences of parametric models of increasing memory which minimize a complexity regularized least squares criterion and establish that these predictors deliver the same statistical performance as oracle predictors. Steinwart and Christmann (2009) prove an oracle inequality regularized ERM algorithms when observations are α-mixing which are close to the optimal i.i.d. rates. Mohri and Rostamizadeh (2009) give results using Rademacher complexity which are both tighter than those using VC-dimension or covering numbers as well as being computable from the data in many cases. Mohri and Rostamizadeh (2010) and Agarwal and Duchi (2013) consider another family of bounds for ϕ-mixing and β-mixing sequences when the predictors are algorithmically stable. Many classes of common machine learning algorithms are amenable to either Rademacher or algorithmic-stability bounds: Kernel-regularized methods, support-vector machines, relative-entropy based regularization, and kernel ridge regression among others. However, methods common to time-series such as AR models, ARIMA models, ARCH and GARCH models (Engle, 1982, 2001), state-space models, and other Box-Jenkins type predictors are not because they are not explicitly regularized, the loss functions are not bounded, and the predictions can depend on more than simply a fixed dimensional past. McDonald et al. (2011b) shows that stationarity alone can be used to impose a kernel-type regularization on an AR model, and hence, following the results of Mohri and Rostamizadeh (2009), is amenable to Rademacher complexity for a bounded loss function.

Other dependence conditions apart from stationary and strong mixing are also considered in the literature. Alquier et al. (2012) develop oracle inequalities and model selection procedures for linear models, neural networks, and non-parametric autoregressions when observations come from causal Bernoulli shifts or bounded, weakly-dependent processes. Under the same weak-dependence conditions, Alquier et al. (2014) extends this result to convex Lipschitz loss functions and examines forecasting of the French GDP. Finally, recent work by Kuznetsov and Mohri (2014) examines both average-path generalization and path-dependent generalization for certain types of non-stationary mixing processes and derives Rademacher complexity bounds.

2. Statistical Learning Theory for I.I.D. Data

Our goal is to control the risk of predictive models, that is, their expected inaccuracy on new data from the same stochastic source as the data used to fit the model. To orient readers, we present some standard results for i.i.d. data, which are adapted to the dependent setting in Section 5.

Let $f : \mathcal{X} \to \mathcal{Y}$ be some function used for making predictions of Y from X. We define a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ which measures the cost of making poor predictions. Throughout this paper, we will assume that $\ell(y, y')$ is a function solely of the difference $y - y'$ where $\ell(\cdot)$ is nonnegative and $\ell(0) = 0$; we take the liberty of denoting that function $\ell(y - y')$. Then the risk of any predictor $f \in \mathcal{F}$ (where f is fixed independently of the data)

is given by

$$R(f) = \mathbb{E}\left[\ell\left(Y - f(X)\right)\right],$$

where $(X, Y) \sim \mathbb{P}$. The risk or generalization error is the expected cost of using f to predict Y from X on a new observation.

Since the true distribution $\mathbb{P}$ is unknown, so is $R(f)$, but we can try to estimate it based on our observed data. The *training error* or *empirical risk* of f is

$$\widehat{R}_n(f) := \frac{1}{n}\sum_{i=1}^{n} \ell\left(Y_i - f(X_i)\right).$$

In other words, the in-sample training error, $\widehat{R}_n(f)$, is the average loss over the actual training points. For any given f, we can bound $R(f)$ in terms of $\widehat{R}_n(f)$ using deviation inequalities, as illustrated below.

When we use the data to chose an $\widehat{f}$ from $\mathcal{F}$, we would like to bound $R(\widehat{f})$. To do so, we must consider not just $\widehat{R}_n(\widehat{f})$, but also the size, in some sense, of $\mathcal{F}$. There are a number of measures for the size or capacity of a model many of which lead to learning theoretic risk bounds. Algorithmic stability (Bousquet and Elisseeff, 2002; Kearns and Ron, 1999) quantifies the sensitivity of the chosen function to small perturbations to the data. Similarly, maximal discrepancy (Vapnik, 2000) asks how different the predictions could be if two functions are chosen using two separate data sets. A more direct, functional-analytic approach partitions $\mathcal{F}$ into equivalence classes under some metric, leading to covering numbers (Pollard, 1984, 1990). Rademacher complexity (Bartlett and Mendelson, 2002; Koltchinskii and Panchenko, 2002) directly describes a model's ability to fit random noise. We focus on a measure which is both intuitive and powerful: Vapnik-Chervonenkis (VC) dimension (Vapnik, 2000; Vapnik and Chervonenkis, 1971).

VC dimension starts as an idea about collections of sets.

Definition 1 *Let $\mathbb{U}$ be some (infinite) set and S a finite-cardinality subset of $\mathbb{U}$. Let $\mathcal{C}$ be a family of subsets of $\mathbb{U}$. We say that $\mathcal{C}$* shatters *S if for every $S' \subseteq S$, $\exists C \in \mathcal{C}$ such that $S' = S \cap C$.*

Essentially, $\mathcal{C}$ can shatter a set S if it can pick out every subset of points in S. This says that the collection $\mathcal{C}$ is very complicated or flexible. The cardinality of the largest set S that can be shattered by $\mathcal{C}$ is the latter's VC dimension.

Definition 2 (VC dimension) *The* Vapnik-Chervonenkis (VC) dimension *of a collection $\mathcal{C}$ of subsets of $\mathbb{U}$ is*

$$\text{VCD}(\mathcal{C}) := \sup\{|S| : S \subseteq \mathbb{U} \text{ and } S \text{ is shattered by } \mathcal{C}\}.$$

To see why this is a "dimension", we need one more notion.

Definition 3 (Growth function) *The* growth function *$G(n, \mathcal{C})$ of a collection $\mathcal{C}$ of subsets of $\mathbb{U}$ is the maximum number of subsets which can be formed by intersecting a set $S \subset \mathbb{U}$ of cardinality n with $\mathcal{C}$,*

$$G(n, \mathcal{C}) := \sup_{S \subset U \,:\, |S| = n} |S \wedge \mathcal{C}|,$$

where $\mathcal{A} \wedge \mathcal{B}$ is the class of all sets $A \cap B$, $A \in \mathcal{A}$, $B \in \mathcal{B}$.

The growth function counts how many *effectively* distinct sets the collection contains, when we can only observe what is going on at n points, not all of $\mathbb{U}$. If $n \leq \text{VCD}(\mathcal{C})$, then from the definitions $G(n, \mathcal{C}) = 2^n$, If the VC dimension is finite, however, and $n > \text{VCD}(\mathcal{C})$, then $G(n, \mathcal{C}) < 2^n$, and in fact it can be shown (Vapnik and Chervonenkis, 1971) that

$$G(n, \mathcal{C}) \leq \left(\frac{en}{\text{VCD}(\mathcal{C})}\right)^{\text{VCD}(\mathcal{C})} \leq (n+1)^{\text{VCD}(\mathcal{C})}.$$

This polynomial growth of capacity with n is why VCD is a "dimension".

Using VC dimension to measure the capacity of function classes is straightforward. Define the indicator function $\mathbf{1}_A(x)$ to take the value 1 if $x \in A$ and 0 otherwise. Suppose that $f \in \mathcal{F}$, $f : \mathbb{U} \to \mathbb{R}$. Each f corresponds to the set

$$C_f = \{(u, u_0) \in \mathbb{U} \times \mathbb{R} : \mathbf{1}_{[0,\infty)}(f(u) - u_0) = 1\},$$

so $\mathcal{F}$ corresponds to the class $\mathcal{C}_{\mathcal{F}} := \{C_f : f \in \mathcal{F}\}$. This extension is sometimes called the pseudo dimension (see e.g. Anthony and Bartlett, 1999; Pollard, 1990).

Theorem 4 (Vapnik and Chervonenkis, 1971) *Suppose that $0 \leq \ell(y, y') \leq M < \infty$. Then,*

$$\mathbb{P}\left(\sup_{f \in \mathcal{F}} |R(f) - \widehat{R}_n(f)| > \epsilon\right) \leq 4G(2n, \mathcal{C}_{\mathcal{F}}) \exp\left\{-\frac{n\epsilon^2}{\Upsilon}\right\} ,$$

where Υ depends only on M and not n or $\mathcal{F}$.

When the loss function is unbounded, similar results hold, but we must consider the composition of the loss function with f. This leads to the set

$$C_{\ell \circ f} = \{(u, u_0, u_0') \in \mathbb{U} \times \mathbb{R} \times \mathbb{R} : \mathbf{1}_{[0,\infty)}(\ell(u_0 - f(u)) - u_0') = 1\},$$

and the corresponding class $\mathcal{C}_{\ell \circ \mathcal{F}} := \{C_{\ell \circ f} : f \in \mathcal{F}\}$.

The concentration result in Theorem 4 works well for independent data. However, for time series, we must be able to handle dependent data. In particular the length n of a sample path $Y_1, \ldots, Y_n$ exaggerates how much information it contains relative to independent observations. Knowing the past allows forecasters to predict future data (at least to some degree), so actually observing those future data points gives less information about the underlying process than in the i.i.d. case. Thus, while in Theorem 4 the probability of large discrepancies between empirical means and their expectations decreases exponentially in n, in the dependent case, the effective sample size may be much less than n, resulting in looser bounds.

3. Time Series

In moving from the i.i.d. setting to time-series forecasting, we need a number of modifications to our initial setup. Rather than observing input/output pairs (Y_i, X_i), we observe a single sequence of random variables $Y_1, \ldots, Y_n$ where each Y_i takes values in $\mathbb{R}^p$, though we can generalize to arbitrary metric spaces at some cost in notational clarity. We are interested in using functions which take past observations as inputs and predict future values of the

process. Specifically, given data from time 1 to time n, we wish to predict time $n+1$. To be clear about notation, we will use the following conventions: $Y_{i:j} := (Y_i, Y_{i+1}, \ldots, Y_j)$, $Y_\infty := Y_{-\infty:\infty}$ is an infinite dimensional sequence; we also have the associated joint distributions $\mathbb{P}_{i:j}$ and $\mathbb{P}_\infty$ and σ-fields $\sigma_{i:j} = \sigma(Y_{i:j})$ and $\sigma_\infty = \sigma(Y_\infty)$.

While we no longer presume i.i.d. data, we still need to restrict the sort of dependent process we work with. We first remind the reader of the notion of (strict or strong) stationarity.

Definition 5 (Stationarity) *A random sequence Y_∞ is stationary when all its finite-dimensional distributions are time-invariant: for all t and all non-negative integers i and j, the random vectors $Y_{t:t+i}$ and $Y_{t+j:t+i+j}$ have the same distribution.*

Stationarity does not imply that the random variables Y_i are independent across time i, only that the marginal distribution of Y_i is constant in time. (And similarly for $Y_{i:i+j}$.) We limit ourselves not just to stationary processes, but also to ones in which widely-separated observations are asymptotically independent. Without this restriction, convergence of the training error to the expected risk could occur arbitrarily slowly, and finite-sample bounds may not exist. In fact, Adams and Nobel (2010) demonstrate that for ergodic processes, finite VC dimension is enough to give consistency, but cannot itself provide rates. The next definition describes the sort of serial dependence which we entertain.

Definition 6 (β-Mixing) *Consider a stationary random sequence Y_∞ defined on a probability space $(\Omega, \Sigma, \mathbb{P}_\infty)$. Let $\mathbb{P}_0$ be the restriction of $\mathbb{P}_\infty$ to $\sigma_{-\infty:0}$, $\mathbb{P}_a$ be the restriction of $\mathbb{P}_\infty$ to $\sigma_{a:\infty}$, and $\mathbb{P}_{0\otimes a}$ be the restriction of $\mathbb{P}_\infty$ to $\sigma(Y_{\infty:0}, Y_{a:\infty})$. The* coefficient of absolute regularity*, or* β-mixing coefficient*, β_a, is given by*

$$\beta_a := \|\mathbb{P}_0 \times \mathbb{P}_a - \mathbb{P}_{0\otimes a}\|_{TV},$$

where $\|\cdot\|_{TV}$ is the total variation norm. A stochastic process is absolutely regular*, or* β-mixing*, if $\beta_a \to 0$ as $a \to \infty$.*

This is only one of many equivalent characterizations of β-mixing (see Bradley, 2005, for others). This definition makes clear that a process is β-mixing if the joint probability of events which are widely separated in time approaches the product of the individual probabilities, that is that Y_∞ is asymptotically independent. Many common time-series models are known to be β-mixing, and the rates of decay are known up to constant factors which are functions of the true parameters of the process. Among the processes for which such results are known are ARMA models (Mokkadem, 1988), GARCH models (Carrasco and Chen, 2002), and certain Markov processes (see Doukhan, 1995, for an overview). Additionally, functions of β-mixing processes are β-mixing, so if $\mathbb{P}_\infty$ could be specified by a linear time-invariant system (see below), state-space model, vector auto regression, or some function of a hidden Markov model, the observed data would satisfy this condition.

Knowing β_a would let us determine the effective sample size of a time series $Y_{1:n}$. In effect, having n dependent-but-mixing data points is like having $\mu < n$ independent ones. Once we determine the correct μ, we can (as we will show) use concentration results for i.i.d. data like Theorem 4 with small corrections.

Algorithm 1: Kalman filter

Recursively generate minimum mean squared error predictions $\widehat{Y}_i$ using the state space model in (1).

Input: Initial guesses for the mean and variance of α_1: $\widehat{\alpha}_1$ and $\widehat{P}_1$

Set $\widehat{Y}_1 = T\widehat{\alpha}_1$.

for $1 \leq i \leq n$ **do**

Filter

$$v_i = Y_i - \widehat{Y}_i, \qquad F_i = (Z\widehat{P}_i Z^\top + H)^{-1},$$
$$K_i = T\widehat{P}_i Z^\top F_i, \qquad L_i = T - K_i Z,$$
$$\widehat{\alpha}_{i+1} = T\widehat{\alpha}_i + K_i v_i, \qquad \widehat{P}_{i+1} = T\widehat{P}_i L_i^\top + GQG^\top.$$

Predict $\widehat{Y}_{i+1} = Z\widehat{\alpha}_{i+1}$.

end

return $\widehat{Y}_{1:n+1}$

4. Linear Time-Invariant Dynamical Systems

Our goal in this paper will be to derive risk bounds for linear time-invariant dynamical systems (LTIs). Such models presume an underlying latent process, and attempt, given observations, to learn that process, predict future values of the latent process, and forecast future observations. Learning algorithms for these goals are linear functions of all previously observed values. Such models nest many common time-series forecasting techniques—ARIMA models, GARCH models, linear-Gaussian state-space models—but, due to their (in general) dependence on the entire past, are not covered by the work discussed in Section 1.1. We present the general form of such models here, provide a canonical forecasting algorithm, and discuss some properties of such models.

Linear dynamical systems model observations Y_i as

$$\begin{aligned} Y_i &= Z\alpha_i + \epsilon_i, \\ \alpha_{i+1} &= T\alpha_i + G\eta_{i+1}. \end{aligned} \tag{1}$$

This is essentially a hidden markov model under certain conditions: Y_i denotes observations, α_i are hidden "state" variables, ϵ_i and η_i are both absolutely continuous random noise with with $\mathbb{E}[\epsilon_i] = \mathbb{E}[\eta_i] = 0$, $\mathbb{E}[\epsilon_i \epsilon_j^\top] = \delta_{ij} H$, and $\mathbb{E}[\eta_i \eta_j^\top] = \delta_{ij} Q$ for all i, j. We further assume that ϵ and η are mutually independent even though this is not strictly necessary, because it makes notation simpler. We require stationarity for our results, and so we also require the LTI to be stationary. This amounts to forcing the complex eigenvalues of T to lie inside the unit circle. We note that the condition that the noise distributions are absolutely continuous means that HMMs in the strictest sense are not members of this family. We also allow the parameter matrices Z, T, H, G, and Q to depend on a (possibly unknown) parameter vector θ, and assume that H and Q are positive definite for all θ.

The *filtering* problem uses observations $Y_{1:i}$ up to time i to learn information about the distribution of α_i. Then, conditional on an estimate $\widehat{\alpha}_i$, we can forecast $\widehat{\alpha}_{i+1}$ and hence

derive a prediction $\widehat{Y}_{i+1}$. For models of this form, one uses the Kalman filter (Anderson and Moore, 2012; Durbin and Koopman, 2001; Kalman, 1960) both to estimate the latent variables, $\alpha_{1:n+1}$ and to generate predictions $\widehat{Y}_{1:n+1}$ (Algorithm 1). This procedure gives the minimum mean-squared error predictions of α_{i+1} (and hence of Y_{i+1}) given $Y_{1:i}$ in the sense that

$$\widehat{\alpha}_{i+1} = \operatorname*{argmin}_a \mathbb{E}\left[tr\left((\alpha_{i+1} - a)(\alpha_{i+1} - a)^\top\right) \mid Y_{1:i}\right].$$

Furthermore, if ϵ_i and η_i are Gaussian, then Algorithm 1 also gives the likelihood for the unknown parameter vector θ. To estimate the unknown parameters, we either: (1) maximize the likelihood returned by the filter; or (2) use the EM algorithm, alternating between running the Kalman filter (the E-step) and maximizing the conditional likelihood by least squares (the M-step). Bayesian estimation works like EM, replacing the M-step with Bayesian updating.

Predictions based on Algorithm 1 are linear functions of previous observations, but these predictions depend on of all previous observations rather than simply a fixed number as would be the case with, say, autoregressive models. More specifically, $\widehat{Y}_{1:n+1} = \mathbf{B} Y_{1:n}$ where

$$(\mathbf{B})_{ij} = b_{ij} = \begin{cases} Z \prod_{k=j+1}^{i-1} L_k K_j & i - j > 1 \\ Z K_j & i - j = 1 \\ 0 & i - j \leq 0 \end{cases} \tag{2}$$

for all $1 \leq i \leq n+1$ (Durbin and Koopman, 2001). Here b_{ij} is the weight for predicting Y_i based on Y_j. Because of this dependence on the entire past, we will require some information about the behavior of the matrices b_{ij} in terms of i and j. Define $\lambda_k(A)$ to be the k^{th} largest absolute eigenvalue of a square matrix A, and let $\lambda_{\max}(A) = \max_k |\lambda_k(A)|$ and $\lambda_{\min}(A) = \min_k |\lambda_k(A)|$.

Conditions for good behavior or LTIs are governed entirely by stationarity. In particular, if $r = \lambda_{\max}(T) < 1$, then the sequence of predictions generated by Algorithm 1 will be stationary, and their dependence on the remote past will decay quickly. The rate of this decay depends on r and $\rho = \lambda_{\max}(T - \overline{K}Z)$, where $K_i \to \overline{K}$. The next three results show that the convergence is exponential in ρ. Proofs of these results are given in Section B.

Lemma 7 *If H is positive definite and $r = \lambda_{\max}(T) < 1$, then $\lambda_{\max}(b_{ij}) = O(r^{i-j-1})$ for any $i > 1$, $j < i$.*

While $\widehat{P}_i = Var(\alpha_{i+1} | Y_{1:i})$ changes with i, for stationary LTIs, $\widehat{P}_i$ converges to a limiting value as $i \to \infty$. This means that the algorithm converges to a steady state as i grows. The next result gives the values to which the algorithm converges, and shows that this convergence occurs exponentially fast.

Lemma 8 *If $\lambda_{\max}(T) < 1$ then:*

1. *The solution, $\overline{P}$ to the matrix equation $P = TPT^\top - TPZ^\top(ZPZ^\top + H)^{-1}ZPT^\top + GQG^\top$ exists and is positive definite.*
2. *$\widehat{P}_i \to \overline{P}$ and $K_i \to \overline{K} = T\overline{P}Z^\top(Z\overline{P}Z^\top + H)^{-1}$.*

Algorithm 2: Steady-state approximate filter

Recursively generate approximate minimum mean squared error predictions $\widehat{Y}_i$ using the state space model in (1).
Input: Initial guess for the mean of α_1: $\widehat{\alpha}_1$
Set $\widehat{Y}_1 = Z\widehat{\alpha}_1$.
Solve $P = TPT^\top - TPZ^\top(ZPZ^\top + H)^{-1}ZPT^\top + GQG^\top$ for P, denote the solution as $\overline{P}$
Set $\overline{K} = T\overline{P}Z^\top(Z\overline{P}Z^\top + H)^{-1}$
for $1 \leq i \leq n$ **do**
  Filter

$$v_i = Y_i - \widehat{Y}_i, \qquad \widehat{\alpha}_{i+1} = T\widehat{\alpha}_i + \overline{K}v_i,$$

  Predict $\widehat{Y}_{i+1} = Z\widehat{\alpha}_{i+1}$.
end
return $\widehat{Y}_{1:n+1}$

3. $0 < \lambda_{\max}(T - \overline{K}Z) = \rho < 1$.

4. *For any matrix norm* $\|\cdot\|$, $\|\widehat{P}_i - \overline{P}\| = O\left(\rho^i\right)$, $\|F_i - \overline{F}\| = O(\rho^i)$, *and* $\|K_i - \overline{K}\| = O\left(\rho^i\right)$.

Since the Kalman filter algorithm converges quickly to a steady state, one could instead use Algorithm 2 which approximates Algorithm 1 but is more computationally efficient.

For Algorithm 2, we can similarly write predictions as linear functions of previous observations. In this case, $\widehat{Y}_{1:n+1} = \mathbf{S}Y_{1:n}$ where the prediction weights are given by

$$(\mathbf{S})_{ij} = s_{ij} = s_{i-j-1} = \begin{cases} Z(T - \overline{K}Z)^{i-j-1}\overline{K} & i - j > 0 \\ 0 & i - j \leq 0 \end{cases} \tag{3}$$

To arrive at this formulation, simply substitute $L_k = (T - \overline{K}Z)$ and $K_j = \overline{K}$ in (2). Notice in particular that the weights depend only on the difference $i - j$ for this algorithm. The next result shows that the prediction weights for the two algorithms converge rapidly.

Lemma 9 *If* $\lambda_{\max}(T) < 1$ *and* H *positive definite. Then,* $\|s_{ij}\| = O(\rho^{i-j-1})$ *and for* $j < i$, $\|b_{ij} - s_{ij}\| = O(\rho^j)$.

We will refer to the class of predictors given by Algorithm 1 as $\mathcal{F}_1$ and those given by Algorithm 2 as $\mathcal{F}_2$.

5. Risk Bounds

With the relevant background in place, we can put the pieces together to derive our results. We use β-mixing to find out how much information is in the data and VC dimension to measure the capacity of the state-space model's prediction functions. The result is a bound on the generalization error of the chosen function $\widehat{f}$. After slightly modifying the definition of "risk" to fit the time-series forecasting scenario and stating necessary technical assumptions, we derive risk bounds for traditional time-series forecasting models.

5.1 Setup and Assumptions

We observe a finite subsequence of random vectors $Y_{1:n}$ from a process Y_∞ defined on a probability space $(\Omega, \Sigma, \mathbb{P}_\infty)$, with $Y_i \in \mathbb{R}^p$. We make the following assumption on the process.

Assumption A $\mathbb{P}_\infty$ *is a stationary,* β*-mixing process with mixing coefficients* β_a, $\forall a > 0$.[1]

Under stationarity, the marginal distribution of Y_t is the same for all t. We deal mainly with the joint distribution of $Y_{1:n+1}$, where we observe the first n observations and try predicting Y_{n+1}. For the remainder of this paper, we will call this joint distribution $\mathbb{P}$. Our results extend to predicting more than one step ahead, but the notation becomes cumbersome.

We must define generalization error and training error slightly differently for time series than in the i.i.d. setting. Using the same notion of loss functions as before, we consider prediction functions $f : \mathbb{R}^{n\times p} \to \mathbb{R}^p$. The function f may use some or all of the past to generate predictions. A function using only the most recent d observations as inputs will be said to have *fixed memory* of length d. Other functions, in particular, the linear time-invariant systems we discuss below, have *growing memory* which means that f may use all the previous data to predict the next data point. These concepts require us to state with some care what we mean by prediction functions, and by time-series training error and risk.

Definition 10 (Time-series risk)

$$R_n(f) := \mathbb{E}\Big[\ell\left(Y_{n+1} - f(Y_{1:n})\right)\Big].$$

The expectation is taken with respect to the joint distribution $\mathbb{P}$ and therefore depends on n.

Definition 11 (Time-series training error)

$$\widehat{R}_n(f) := \frac{1}{n-d}\sum_{i=d+1}^{n} \ell\left(Y_i - f(Y_{1:i-1})\right).$$

In order to make use of this single definition of training error, we let $d \geq 0$. In fixed memory cases—say an AR(2)—d has an obvious meaning, and $f(Y_{1:i}) = f(Y_{i-d+1:i})$ by definition of fixed memory, while with growing memory, $d = 0$, and we define $Y_{1:0} := \varnothing$.

To control the generalization error for time-series forecasting, we make one final assumption, about the possible magnitude of the losses. Specifically, we weaken the bounded loss assumption we used in Section 2 to allow for unbounded loss as long as we retain some control on moments of the loss.

1. In order to apply the results, one must either know β_a for some a or be able to estimate it with sufficient precision and accuracy. McDonald et al. (2011a, 2015) show how to estimate the mixing coefficients nonparametrically, based on a single sample from the process. However, those results (and those contained in this paper) only apply if the data generating process *is* β-mixing, an assumption that cannot be verified.

Assumption B *Assume that for all $f \in \mathcal{F}$ and all $d \in \mathbb{N}$*

$$Q_d(f) := \sqrt{\mathbb{E}_{\mathbb{P}}\left[\ell\left(Y_{d+1} - f(Y_{1:d})\right)^2\right]} \leq M < \infty.$$

Assumption B will be satisfied for Y_i sub-Gaussian, as well as other distributions with bounded second moment. These include, for instance, heavy-tailed Lévy noises where the tails of the pdf decay faster than an inverse cubic.

5.2 Fixed Memory

We can now state our results giving finite sample risk bounds for the problem of time-series forecasting. We begin with the fixed memory setting; the next section will allow the memory length to grow.

Theorem 12 *Suppose that Assumption A and Assumption B hold, that the model class $\mathcal{F}$ has a fixed memory length $d < n$, and that we have a sample $Y_{1:n}$. Let μ and $a > d$ be integers such that $2\mu a + d = n$.*[2] *Then, for all $0 < \epsilon < \frac{e^{3/2}}{\sqrt{2}}$,*

$$\mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_n(f)}{Q_d(f)} > \epsilon\right) \leq 8G(n, \mathcal{C}_{\ell\circ\mathcal{F}}) \exp\left\{-\frac{\mu \exp\left(W\left(-\frac{2\epsilon^2}{e^4}\right) + 4\right)}{4}\right\} + 2\mu\beta_{a-d},$$

where $W(\cdot)$ is the Lambert W function.

The implications of this theorem are considerable. Given a finite sample of length n, we can say that with high probability, future prediction errors will not be much larger than our observed training errors. It makes no difference whether the model is correctly specified. This stands in stark contrast to model selection tools like AIC or BIC which appeal to asymptotics. Moreover, given a model class $\mathcal{F}$, we can say exactly how much data we need to have good control of the prediction risk. As the effective data size increases, the training error is a better and better estimate of the generalization error, uniformly over all of $\mathcal{F}$, provided $\mu, a \to \infty$ such that $\beta_{a-d} = o(1/\mu)$.

The Lambert W function in the exponential term deserves some explanation. The Lambert W function is defined as the inverse of $f(w) = w \exp w$ (c.f. Corless et al., 1996). A strictly, but only slightly, worse bound can be achieved by noting that

$$\exp\left(W\left(-\frac{2\epsilon^2}{e^4}\right) + 4\right) \leq \frac{\epsilon^{8/3}}{4^{2/3}}$$

for all $\epsilon \in [0, 1]$ (see Cortes et al., 2010, for the derivation).

The difference between expected and empirical risk is only interesting when $R_n(f)$ exceeds $\widehat{R}_n(f)$. Due to the supremum preceding $\frac{R_n(f)-\widehat{R}_n(f)}{Q_d(f)}$, events where the training error exceeds the expected risk are irrelevant (as this term will be negative). Therefore, we are

2. By making appropriate modifications to the definition of the training error and some of the proof elements, one could allow $2\mu a + d < n$, but we obviate this issue for the sake of clarity.

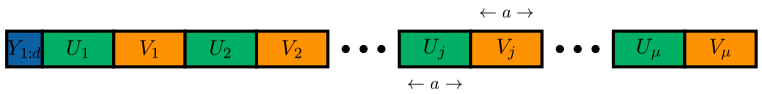


Figure 1: The blocking procedure divides $Y_{d+1:n}$ into 2μ alternating blocks U_j (green) and V_j (orange) each of length a. It ignores the first d observations $Y_{1:d}$ (blue).

only concerned with $0 \leq \widehat{R}_n(f) \leq R_n(f)$. Of course, as discussed in Section 2, for most estimation procedures, f is chosen to make $\widehat{R}_n(f)$ as small as possible.

Before we prove Theorem 12, we will state a corollary which puts the same result in a form that is sometimes easier to use.

Corollary 13 *Under the conditions of Theorem 12, for any $f \in \mathcal{F}$, the following bound holds with probability at least $1 - \eta$, for all $\eta > 2\mu\beta_{a-d}$,*

$$R_n(f) \leq \widehat{R}_n(f) + Me^2\sqrt{\frac{\mathcal{E}(4 - \log \mathcal{E})}{2}},$$

with

$$\mathcal{E} = \frac{4 \log G(n, \mathcal{C}_{\ell\circ\mathcal{F}}) + 4 \log 8/\eta'}{\mu},$$

and $\eta' = \eta - 2\mu\beta_{a-d}$.

We now prove both Theorem 12 and Corollary 13 to provide the reader with some intuition for the types of arguments necessary. We defer proof of the remainder of our results to Section B.

Proof [of Theorem 12 and Corollary 13] The first step is to move from the actual sample size n to the effective sample size μ which depends on the β-mixing behavior. Let a and μ be non-negative integers such that $2a\mu + d = n$. Now divide $Y_{d+1:n}$ into 2μ blocks, each of length a, ignoring the first d observations. Identify the blocks as follows:

$$U_j = \{Y_i : 2(j-1)a + d + 1 \leq i \leq (2j-1)a + d\},$$
$$V_j = \{Y_i : (2j-1)a + d + 1 \leq i \leq 2ja + d\}.$$

Let $\mathbf{U}$ be the sequence of odd blocks U_j, and let $\mathbf{V}$ be the sequence of even blocks V_j. A graphical depiction of the blocking procedure is shown in Figure 1. Finally, let $\mathbf{U}'$ be a sequence of blocks which are mutually independent but such that each block has the same distribution as a block from the original sequence. That is construct U'_j such that

$$\mathcal{L}(U'_j) = \mathcal{L}(U_j) = \mathcal{L}(U_1),$$

where $\mathcal{L}(\cdot)$ means the probability law of the argument.

Let $\widehat{R}_{\mathbf{U}}(f)$, $\widehat{R}_{\mathbf{U}'}(f)$, and $\widehat{R}_{\mathbf{V}}(f)$ be the empirical risk of f based on the block sequences $\mathbf{U}$, $\mathbf{U}'$, and $\mathbf{V}$ respectively. We have

$$\begin{aligned}
\widehat{R}_n(f) &= \frac{1}{n-d}\sum_{i=d+1}^{n} \ell\left(Y_i - f(Y_{1:i-1})\right) \\
&= \frac{1}{n-d}\left[\sum_{i:Y_i\in\mathbf{U}} \ell\left(Y_i - f(Y_{1:i-1})\right) + \sum_{j:Y_j\in\mathbf{V}} \ell\left(Y_j - f(Y_{1:j-1})\right)\right] \\
&= \frac{1}{2}\left[\frac{2}{n-d}\sum_{i:Y_i\in\mathbf{U}} \ell\left(Y_i - f(Y_{1:i-1})\right) + \frac{2}{n-d}\sum_{j:Y_j\in\mathbf{V}} \ell\left(Y_j - f(Y_{1:j-1})\right)\right] \\
&= \frac{1}{2}\left[\widehat{R}_{\mathbf{U}}(f) + \widehat{R}_{\mathbf{V}}(f)\right].
\end{aligned}$$

Then,

$$\begin{aligned}
&\mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_n(f)}{Q_d(f)} > \epsilon\right) \\
&= \mathbb{P}\left(\sup_{f\in\mathcal{F}} \left[\frac{R_n(f) - \widehat{R}_{\mathbf{U}}(f)}{2Q_d(f)} + \frac{R_n(f) - \widehat{R}_{\mathbf{V}}(f)}{2Q_d(f)}\right] > \epsilon\right) \\
&\le \mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_{\mathbf{U}}(f)}{Q_d(f)} + \sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_{\mathbf{V}}(f)}{Q_d(f)} > 2\epsilon\right) \qquad (4) \\
&\le \mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_{\mathbf{U}}(f)}{Q_d(f)} > \epsilon\right) + \mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_{\mathbf{V}}(f)}{Q_d(f)} > \epsilon\right) \qquad (5) \\
&= 2\mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_{\mathbf{U}}(f)}{Q_d(f)} > \epsilon\right) \\
&= 2\mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_d(f) - \widehat{R}_{\mathbf{U}}(f)}{Q_d(f)} > \epsilon\right). \qquad (6)
\end{aligned}$$

Here, (4) follows by the convexity of the supremum and (5) by a union bound. Now, for (6), as f has fixed memory d, we have that $f(Y_{1:j}) = f(Y_{j-d+1:j})$. Thus

$$\begin{aligned}
R_n(f) &= \mathbb{E}\left[\ell\left(Y_{n+1} - f(Y_{1:n})\right)\right] = \mathbb{E}\left[\ell\left(Y_{n+1} - f(Y_{n-d+1:n})\right)\right] \\
&= \mathbb{E}\left[\ell\left(Y_{d+1} - f(Y_{1:d})\right)\right] = R_d(f)
\end{aligned}$$

by stationarity and, similarly, $Q_d(f) = \sqrt{\mathbb{E}\left[\ell\left(Y_{d+1} - f(Y_{1:d})\right)^2\right]}$. Thus $R_d(f)$ and $Q_d(f)$ depend on only $d+1$ data values. Likewise, a "point" in the training error $\widehat{R}_{\mathbf{U}}(f)$ depends on $d+1$ data values. Therefore, a prediction at any Y_i in some block U_j is separated by at least $a-d$ observations from any $Y_{i'}$ in different block $U_{j'}$. Furthermore, we are estimating an expectation $R_d(f)$ which depends on $d+1$ values with an empirical expectation $\widehat{R}_{\mathbf{U}}(f)$ which is a dependent sum of components each of which depends on $d+1$ observed data values.

Therefore, we can apply Lemma 4.1 in Yu (1994) (reproduced as Lemma 21 in Section A) to the event $\left\{\sup_{f\in\mathcal{F}} \frac{R_d(f) - \widehat{R}_{\mathbf{U}}(f)}{Q_d(f)} > \epsilon\right\}$. This allows us to move from statements about the dependent blocks in $\widehat{R}_{\mathbf{U}}(f)$ to statements about the independent blocks in $\widehat{R}_{\mathbf{U}'}(f)$ with a slight correction which accounts for the worst-case dependence between adjacent blocks: β_{a-d}. Therefore,

$$2\mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_d(f) - \widehat{R}_{\mathbf{U}}(f)}{Q_d(f)} > \epsilon\right) \le 2\mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_d(f) - \widehat{R}_{\mathbf{U}'}(f)}{Q_d(f)} > \epsilon\right) + 2\mu\beta_{a-d}, \qquad (7)$$

where the probability on the right of (7) is for the σ-field generated by the independent block sequence $\mathbf{U}'$. Let us now introduce the growth function for the class of "blocked" predictor losses $\mathcal{C}^{\Sigma}_{\ell\circ\mathcal{F}} = \{C^{\Sigma}_{\ell\circ\mathcal{F}} : f \in \mathcal{F}\}$, where[3]

$$C^{\Sigma}_{\ell\circ\mathcal{F}} = \left\{(\mathbf{u}, \mathbf{d}, b) : \mathbf{1}_{[0,\infty)}\left(\sum_{j=1}^{a} \ell(d_j - f(u_j)) - b\right) = 1, \quad \mathbf{u} \in \mathbb{U}^a, \quad \mathbf{d} \in \mathbb{R}^{k\times a}, \ b \in \mathbb{R}\right\}.$$

This allows us to state that

$$2\mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_d(f) - \widehat{R}_{\mathbf{U}'}(f)}{Q_d(f)} > \epsilon\right) \le 8G(2\mu, \mathcal{C}^{\Sigma}_{\ell\circ\mathcal{F}}) \exp\left\{-\frac{\mu \exp\left(W\left(-\frac{2\epsilon^2}{e^4}\right) + 4\right)}{4}\right\}, \qquad (8)$$

where we have applied Corollary 23 to bound the independent blocks $\mathbf{U}'$. Since there are μ independent blocks, this upper bound is in terms of μ effectively independent data "points" penalized by the correction β_{a-d} which adjusts for the worst case dependence between "points" rather than n dependent data points. However, the growth function is in terms of the class of blocked predictors and depends on a. To remove this dependence, we present a novel result, Lemma 24. This lemma shows that $G(2\mu, C^{\Sigma}_{\ell\circ\mathcal{F}}) \le G(2\mu a, \mathcal{C}_{\ell\circ\mathcal{F}}) \le G(n, \mathcal{C}_{\ell\circ\mathcal{F}})$, giving the desired result.

To prove the corollary, set the right hand side of (8) to η, take $\eta' = \eta - 2\mu\beta_{a-d}$, and solve for ϵ. We get that with probability at least $1-\eta$, for all $f \in \mathcal{F}$,

$$\frac{R_n(f) - \widehat{R}_n(f)}{Q_d(f)} \le \epsilon.$$

Solving the equation

$$\eta' = 8G(n, \mathcal{C}_{\ell\circ\mathcal{F}}) \exp\left\{-\frac{\mu \exp\left(W\left(-\frac{2\epsilon^2}{e^4}\right) + 4\right)}{4}\right\}$$

implies

$$\epsilon = e^2\sqrt{\frac{\mathcal{E}(4 - \log \mathcal{E})}{2}}$$

3. The Σ indicates a sum over the coordinates of the block.

with

$$\mathcal{E} = \frac{4\log G(n, \mathcal{C}_{\ell\circ\mathcal{F}}) + 4\log 8/\eta'}{\mu}.$$

■

The only obstacle to the use of Theorem 12 is knowledge of the $G(n, \mathcal{C}_{\ell\circ\mathcal{F}})$ or $\text{VCD}(\mathcal{C}_{\ell\circ\mathcal{F}})$. For some models, these can be calculated explicitly.

Theorem 14 *If $\ell(y - y')$ is monotone increasing in $|y - y'|$, then for the class of AR(d) models,*

$$G(n, \mathcal{C}_{\ell\circ\mathcal{F}}) \leq \left(\frac{2en}{d+1}\right)^{d+1}.$$

If $\ell(y-y')$ is monotone increasing in each coordinate of $|y_j - y'_j|$, then for the class of vector autoregressive models of order d with p time-series,

$$G(n, \mathcal{C}_{\ell\circ\mathcal{F}}) \leq \left(\frac{2en}{d+1}\right)^{p(d+1)}.$$

The proof of Theorem 14 is given in Section B. We note that this result applies equally to Bayesian VARs, however, this is likely conservative as the prior tends to restrict the effective complexity of the function class.[4]

5.3 Bounds for LTIs

As discussed in Section 4, our goal is to derive similar bounds for linear time-invariant dynamical systems which produce forecasts via Algorithm 1 or Algorithm 2. We begin with a result for the simple case with predictions generated by Algorithm 2.

Theorem 15 *Suppose that Assumption A and Assumption B hold, and that the model class $\mathcal{F}_2$ is generated by Algorithm 2 with $\lambda_{\max}(T) < 1$. Further assume that the loss function ℓ is a norm and let $\ell^*(A) = \sup_{z\neq 0} \ell(Az)/\ell(z)$ be the matrix norm induced by ℓ. Given a time-series of length n, fix some $1 \leq d < n$, and let μ and a be integers such that $2\mu a + d = n$.*

4. Here we should mention that these risk bounds are frequentist in nature. We mean that if one treats Bayesian methods as a regularization technique and predicts with the posterior mean or mode, then our results hold. However, from a subjective Bayesian perspective, our results add nothing since all inference can be derived from the posterior. For further discussion of the frequentist risk properties of Bayesian methods under mis-specification, see for example Kleijn and van der Vaart (2006); Müller (2013) or Shalizi (2009).

Then

$$\begin{aligned}
&\mathbb{P}\left(\sup_{f_2\in\mathcal{F}_2} \frac{R_n(f_2) - \widehat{R}_n(f_2) - \delta_d(f_2)}{Q_d(f_2)} > \epsilon\right) \\
&\leq 8G(n, \mathcal{C}_{\ell\circ\mathcal{F}'_2}) \exp\left\{-\frac{\mu\exp\left(W\left(-\frac{2\epsilon^2}{e^4}\right)+4\right)}{4}\right\} + 2\mu\beta_{a-d}, \\
&\leq 8\left(\frac{2en}{d+1}\right)^{p(d+1)} \exp\left\{-\frac{\mu\exp\left(W\left(-\frac{2\epsilon^2}{e^4}\right)+4\right)}{4}\right\} + 2\mu\beta_{a-d},
\end{aligned}$$

where

$$\delta_d(f_2) = \frac{1}{n-d}\sum_{i=d+1}^{n}\left[\ell\left(Y_i - \sum_{j=i-d}^{i-1} s_{ij}Y_j\right) - \ell\left(Y_i - \sum_{j=1}^{i-1} s_{ij}Y_j\right)\right] + \mathbb{E}[\ell(Y_1)]\sum_{j=1}^{n-d}\ell^*(s_{nj}).$$

The $\delta_d(f_2)$ term arises from taking a fixed-memory approximation, of length d, to predictors with growing memory[5]. As becomes clear in the proof (see Section B), we make this approximation to apply the previous theorem, but it involves a trade-off. As $d \nearrow n$, $\delta_d(f_2) \searrow 0$, but this drives $\mu \searrow 0$, resulting in fewer effective training points, whereas smaller d has the opposite effect. The summation inside square braces on the left is the difference between empirical risks for f_2 and that of the truncated predictor f'_2 which uses only the most recent d data values. That is

$$\frac{1}{n-d-1}\sum_{i=d}^{n-1}\left[\ell\left(Y_{i+1} - \sum_{j=i-d+1}^{i} s_{ij}Y_j\right) - \ell\left(Y_{i+1} - \sum_{j=1}^{i} s_{ij}Y_j\right)\right] = \widehat{R}_n(f'_2) - \widehat{R}_n(f_2).$$

This term is easily calculated from the data. Also, $\delta_d(f_2)$ depends on $\mathbb{E}\big[\ell(Y_1)\big]$ which is not necessarily desirable. However, Assumption B has the consequence that $\mathbb{E}\big[\ell(Y_1)\big] \leq M < \infty$ as long as $\mathcal{F}_2$ allows $s_{ij} = 0$. Finally, we reiterate that $s_{ij} = s_{i-j-1}$ is a function only of the difference between i and j. Finally, note that the upper bound depends on the growth function of the truncated class $\mathcal{F}'_2$, which can be bounded using Theorem 14.

Corollary 16 *Under the conditions of either Theorem 15, for any $f_2 \in \mathcal{F}_2$, with probability at least $1 - \eta$,*

$$R_n(f_2) \leq \widehat{R}_n(f_2) + \delta_d(f_2) + Me^2\sqrt{\frac{\mathcal{E}(4 - \log\mathcal{E})}{2}},$$

where

$$\mathcal{E} = \frac{4\log G(n, \mathcal{C}_{\ell\circ\mathcal{F}'_2}) + 4\log 8/\eta'}{\mu},$$

and $\eta' = \eta - 2\mu\beta_{a-d}$. Furthermore,

$$\delta_d(f_2) = O_{\mathbb{P}}(\rho^d).$$

5. There are several ways one could make such an approximation. To simplify the proof, we have set all coefficients at lags beyond d to zero rather than, e.g., asking for the d-memory linear predictor coming closest in L_2 to the infinite-memory predictor.

For the case of Algorithm 1, Assumption B no longer makes sense as it fails to reflect the fact each $f_1 \in \mathcal{F}_1$ is a time varying function of previous observations as illustrated in (2). We therefore strengthen the assumption slightly. Notice in particular that f_1' depends on n as illustrated in (2).

Assumption C *For all f_1' generated by truncating Algorithm 1 to depend on the most recent d observations, and all $n > d$,*

1. $$Q_d(f_1') := \sqrt{\mathbb{E}_{\mathbb{P}}\left[\ell\left(Y_{n+1} - f_1'(Y_{n-d+1:n})\right)^2\right]} \leq M < \infty.$$

2. *For every $\epsilon > 0$ there exists $\delta > 0$ such that, for all measurable sets A, with $\mathbb{P}(A) \leq \delta$, and every f_1',*
$$\int_A d\mathbb{P}\left[\ell\left(Y_{n+1} - f_1'(Y_{n-d+1:n})\right)^2\right] \leq \epsilon.$$

The first part of this assumption is analogous to Assumption B but for predictors $f_1 \in \mathcal{F}_1$. However, we need only impose the assumption on a fixed-memory version of these growing memory predictors. Note also that for the case of Algorithm 2, as in Theorem 15, for any n, $f_2 \in \mathcal{F}_2$, we have $Q_d(f_2') = \sqrt{\mathbb{E}_{\mathbb{P}}\left[\ell\left(Y_n - f_2'(Y_{n-d:n-1})\right)^2\right]}$. For Algorithm 1, this is no longer the case (as f_1' depends on n). However, the dependence on n will not be necessary. The second part of this assumption ensures that the loss of the truncated versions of these predictors is uniformly integrable. Under this assumption, since predictions $f_1'(Y_{n-d+1:n}) - f_2'(Y_{n-d+1:n}) = O_{\mathbb{P}}(1)$, we have that $\lim_{n\to} Q_d(f_1') = Q_d(f_2)$. This statement is made rigorous in the proof.

Theorem 17 *Suppose that Assumption A and Assumption C hold, and that the model class $\mathcal{F}_1$ is generated by Algorithm 1 with $\lambda_{\max}(T) < 1$. Further assume that the loss function ℓ is a norm and let $\ell^*(A) = \sup_{z\neq 0} \frac{\ell(Az)}{\ell(z)}$ be the matrix norm induced by ℓ. Given a time-series of length n, fix some $1 \leq d < n$, and let μ and a be integers such that $2\mu a + d = n$. Then*

$$\begin{aligned}
&\mathbb{P}\left(\sup_{f_n \in \mathcal{F}} \frac{R_n(f_1) - \widehat{R}_n(f_1) - \delta_d(f_1)}{\lim_{n\to\infty} Q_d(f_1')} > \epsilon\right) \\
&\leq 8G(n, \mathcal{C}_{\ell\circ\mathcal{F}_2'}) \exp\left\{-\frac{\mu \exp\left(W\left(-\frac{2\epsilon^2}{e^4}\right) + 4\right)}{4}\right\} + 2\mu\beta_{a-d}, \\
&\leq 8\left(\frac{2en}{d+1}\right)^{p(d+1)} \exp\left\{-\frac{\mu \exp\left(W\left(-\frac{2\epsilon^2}{e^4}\right) + 4\right)}{4}\right\} + 2\mu\beta_{a-d},
\end{aligned}$$

where

$$\begin{aligned}
\delta_d(f_1) = \frac{1}{n-d}\sum_{i=d+1}^{n}\left[\ell\left(Y_i - \sum_{j=i-d}^{i-1} b_{ij}Y_j\right) - \ell\left(Y_i - \sum_{j=1}^{i-1} s_{i-j-1}Y_j\right)\right] \\
+ \mathbb{E}[\ell(Y_1)]\sum_{j=1}^{n-d}\ell^*(b_{nj}).
\end{aligned}$$

For this result, the summation inside square braces on the left is the difference between empirical risk for f_1 and that of the truncated steady-state predictor f_2' which uses only the most recent d data values. That is

$$\frac{1}{n-d}\sum_{i=d+1}^{n}\left[\ell\left(Y_i - \sum_{j=i-d}^{i-1} b_{ij}Y_j\right) - \ell\left(Y_{i+1} - \sum_{j=1}^{i} s_{ij}Y_j\right)\right] = \widehat{R}_n(f_1) - \widehat{R}_n(f_2')$$

where f_2' is given by Algorithm 2. Again, this term is easily calculated from the data. Furthermore, the upper bound depends on the growth function of the level sets of $\mathcal{F}_2'$ which is easily bounded using Theorem 14.

Corollary 18 *Under the conditions of Theorem 17, for any $f_1 \in \mathcal{F}_1$, with probability at least $1 - \eta$,*

$$R_n(f_1) \leq \widehat{R}_n(f_1) + \delta_d(f_1) + Me^2\sqrt{\frac{\mathcal{E}(4 - \log\mathcal{E})}{2}},$$

where

$$\mathcal{E} = \frac{4\log G(n, \mathcal{C}_{\ell\circ\mathcal{F}_2'}) + 4\log 8/\eta'}{\mu},$$

and $\eta' = \eta - 2\mu\beta_{a-d}$. Furthermore,

$$\delta_d(f_1) = O_{\mathbb{P}}(r^d) + O(\rho^{n-d+1}) + O_{\mathbb{P}}((n-d)^{-1}).$$

This corollary shows that $\delta_d(f_1)$ decays rapidly as long as $d \to \infty$ and $d/n \to 0$. But for LTIs, it is simple to compute $\delta_d(f_1)$ or $\delta_d(f_2)$ using results from Algorithm 1 or Algorithm 2, so this and Corollary 16 let us compute risk bounds for common time-series forecasting models without appealing to the asymptotic form of $\delta_d(f)$.

As this bound is rather complicated, in that d, a, μ and η are expected to go to either 0 or ∞ as functions of n, we provide the following simple example demonstrating convergence.

Example *Suppose Y_∞ is a 1-dimensional geometrically β-mixing time series with $\beta_a = O(k^a)$ for some constant $0 < k < 1$. We will choose as our predictor the state-space model as in* (1)

$$\begin{aligned}
Y_i &= Z\alpha_i + H\epsilon_i, \\
\alpha_{i+1} &= T\alpha_i + G\eta_{i+1},
\end{aligned}$$

with a univariate state α. We will maximize the likelihood to estimate $\theta = (Z, H, T, G)$ and apply Algorithm 1 to generate forecasts from n samples. Thus

$$\mathcal{F}_1 = \{f_1 : Z \in R,\ H, G > 0,\ -1 < T < 1\}.$$

To achieve reasonable bounds, we fix $0 < \gamma < 1$ and choose $\mu = (n-d)^{1-\gamma}$, $a = 0.5(n-d)^{\gamma}$, and $\eta = 3\mu\beta_{a-d}$. With these choices, and using C as a mutable constant independent of n and d,

$$\mathcal{E} = C\frac{4\log(d+1)\log(2en/(d+1)) - (1-\gamma)\log(n-d) + [(n-d)^{\gamma} - d]\log(1/k)}{(n-d)^{1-\gamma}}.$$

Here, we have used Theorem 14 to bound the logarithm of the growth function of the growing memory predictor with $\log(d+1)\log(2en/(d+1))$ for truncation length d. Therefore, for n and d such that $(n-d)^\gamma > C\log(n)\log(d)$, $\mathcal{E} < C(n-d)^{-(1-2\gamma)}$. To complete the bound, we need to choose d such that $\eta \to 0$, $r^d \to 0$, and $n-d \to \infty$. To achieve the first, we have

$$\eta = \mu\beta_{a-d} = g(n-d)^{1-\gamma}k^{0.5(n-d)^\gamma - d}.$$

Therefore, we require $d = o(n^\gamma)$. Thus, as long as $d = o(n^\gamma)$ and $d \to \infty$, for n large enough (depending on many constants), Corollary 18 gives

$$R_n(f_1) \leq \widehat{R}_n(f_1) + C\sqrt{\frac{1}{n^{1-2\gamma}}}$$

for some C with high probability.

6. Bounds in Practice

We now show how the theorems of the previous section can be used both to quantify prediction risk and to select models. We first estimate a simple stochastic volatility model using IBM return data and calculate the bound for the predicted volatility using Corollary 18. Then we show how the same methods can be used for typical macroeconomic forecasting models.

6.1 Stochastic Volatility Model

We estimate a standard stochastic volatility model using daily log returns for IBM from January 1962 until October 2011 ($n = 12541$ observations). Figure 2 shows the squared log-return series.

The model we investigate is

$$\begin{aligned} Y_i &= \sigma z_i \exp(\alpha_i/2), & z_i &\sim \mathrm{N}(0,1), \\ \alpha_{i+1} &= \phi\alpha_i + \eta_i, & \eta_i &\sim \mathrm{N}(0,\sigma_\eta^2), \end{aligned}$$

where the disturbances z_i and w_i are mutually and serially independent. Following Harvey et al. (1994), we linearize this by taking the natural log of the observation equation:

$$\begin{aligned} \log Y_i^2 &= \varsigma + \frac{1}{2}\alpha_i + \epsilon_i, & \epsilon_i &= \log z_i^2 - \xi, \\ \varsigma &= \log\sigma^2 + \xi, & \xi &= \mathbb{E}[\log z_1^2]. \end{aligned}$$

The noise term ϵ_i is no longer Gaussian (it has a shifted log-gamma distribution), but the Kalman filter will still give the minimum-mean-squared-error linear estimate of the variance sequence $\alpha_{1:n+1}$. The observation variance is now $\pi^2/2$.

To match the data to the model, let y_i be the log returns and remove 688 observations where the return was 0 (the price did not change from one day to the next). Using the Kalman filter, the negative log likelihood is given by

$$\mathcal{L}(Y_{1:n}|\varsigma,\phi,\sigma_\eta^2) \propto \sum_{i=1}^{n} \log F_i + v_i^2 F_i^{-1}.$$

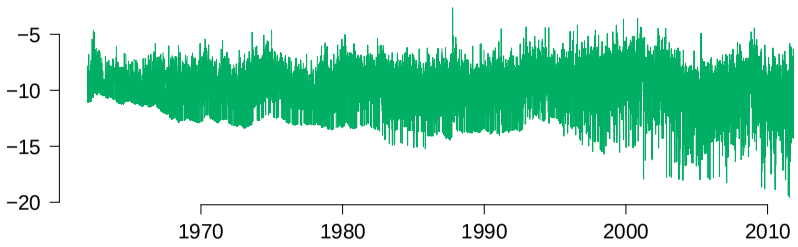


Figure 2: Daily realized volatility (squared log returns) for IBM from 1962–2011.

Model	Training error	AIC-Baseline	Risk bound ($1-\eta > 0.95$)	Effective VC
SV	1.44	-2816	15.92	3
AR(2)	1.49	-348	15.00	3
Mean	1.51	0	12.63	1

Table 1: This table shows the training error and risk bounds for 3 models. AIC is given as the difference from predicting with the global mean (the smaller the value, the more support for that model). The "Effective VC" dimension column reports the exponent in the bound on the growth function. This is slightly different from the VC dimension of the model.

Minimizing this gives estimates $\varsigma = -9.62$, $\phi = 0.996$, and $\sigma_\eta^2 = 0.003$. Using absolute error loss gives training error $\widehat{R}_n(f) = 1.44$.

To actually calculate the bound, we need a few more values. First, using the methods in McDonald et al. (2011a, 2015), we can estimate $\beta_8 = 0.017$. For $a > 8$, the optimal point estimate of β_a is 0. While this is presumably an underestimate, we will take $\beta_a = 0$ for $a > 8$. For the upper bound in Assumption B, we use $M = \sqrt{\pi^2/2}$ which corresponds to the SV model being true.

Combining these values with the VC dimension for the stochastic volatility model, we can bound the prediction risk. Finally, we take $\mu = 538$, $a = 11$, $d = 2$, and $\mathbb{E}[|Y_1|] = M$. The result is the bound $R_n(f) \leq 15.92$ with probability at least 0.95. In other words, the bound is much larger than the training error, but this is to be expected: the data are highly dependent, so the large n translates into a relatively small effective sample size μ.

For comparison, we also computed the bound for forecasts produced with an AR(2) model (with intercept) and with the global mean alone. In the case of the mean, we take $\mu = 658$ and $a = 9$ since in this case, $d = 0$. The results are shown in Table 1. The stochastic volatility model reduces the training error by 5% relative to predicting with the mean, an improvement which is marginal at best. But the resulting risk bound clearly demonstrates that given the small effective sample size, even this gain may be spurious: it is likely that the stochastic volatility model is simply over-fitting.

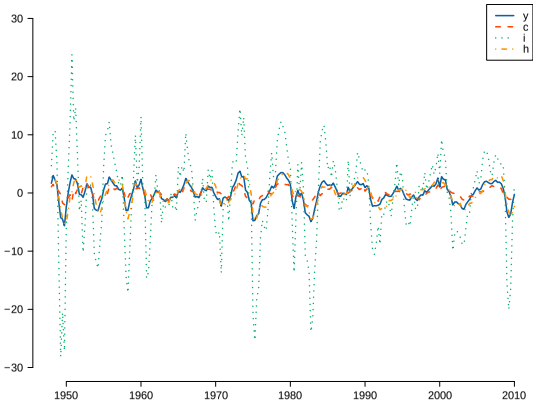


Figure 3: Time series used to estimate the RBC model. These are quarterly data from 1948:I until 2010:I. The blue line is GDP (output), the red line is consumption, the green line is investment, and the orange line is hours worked. These data are plotted as percentage deviations from trend as discussed in the Online Appendix.

6.2 Real Business Cycle Model

In this section, we will discuss the methodology for applying risk bounds to the forecasts generated by the real business cycle (RBC) model. This is a standard tool in macroeconomic forecasting. For a discussion of the RBC model and the standard methods used to bring such models to the data, see, for example DeJong and Dave (2007); DeJong et al. (2000); Fernández-Villaverde (2010); Kydland and Prescott (1982); Romer (2011); Sims (2002); Smets and Wouters (2007).

To estimate the parameters of this model, we use four data series. These are GDP y_t, consumption c_t, investment i_t, and hours worked n_t which are from the Federal Reserve Economic Database. The series we use are shown in Figure 3.

The basic idea of the estimation is to transform the model from an inter-temporal optimization form into a state space model. This leads to a linear, Gaussian state-space model with four observed variables (listed above), and two unobserved state variables. The mapping from parameters of the optimization problem to parameters of the state-space model is nonlinear, but, for each parameter setting, the Kalman filter returns the likelihood, so that likelihood methods are possible. As the data are fairly uninformative about many of the parameters, we estimate by maximizing a penalized likelihood, rather than a simple likelihood. Then the Kalman filter produces in-sample forecasts which are linear in past values of the data, so that we could potentially apply the growing memory bound.

For macroeconomic time series, there is not enough data to give nontrivial bounds, regardless of the mixing coefficients or the size of the finite memory approximation. Figure 3

shows $n = 245$ observations. The minimal possible finite approximation model is a VAR with one lag and four time series. In this case, since we are dealing with vector valued forecasts, we take $\ell(y - y') = \|y - y'\|_2$ so that the induced matrix norm of A is $\ell^*(A) = \sigma_{\max}(A)$, the largest singular value of A. We assume that Assumption C is satisfied with $M = 0.1$ and demand confidence 0.95 ($\eta = 0.05$). By Theorem 14, this class has growth function bounded by $(en)^8$.

Again, using the methods of McDonald et al. (2011a), we can estimate the β-mixing coefficients of the macroeconomic data set. The result is a point estimate $\beta_4 = 0$. Assuming that this is approximately accurate (0 is of course an underestimate), this suggests that the effective size of the macroeconomic data set is no more than about $\mu = 25$, much smaller then $n = 245$. To calculate the bound, we assume that $\mathbb{E}[\|Y\|_2] < 0.1$. The training error of the fitted RBC model is $\widehat{R}_n(f) = 0.046$. The bound is $R_n(f) < 2.29$.

The bound here is three orders of magnitude larger than the training error. If the bound is tight, then this suggests that the training error severely underestimates the true prediction risk. Of course, this should not be too surprising since the RBC model has 11 parameters and we are trying to get confidence intervals using only 25 effective data points.

In some sense, the empirical results in this section may seem slightly unreasonable. Since the results are only upper bounds, it is important to get an idea as to how tight they may be. We address this issue in the next section.

7. Properties of Our Results

In the previous section, we showed that the upper bound for the risk of standard macroeconomic forecasting models may be large. This of course raises the question "How tight are these bounds?" We address this question next and then discuss how to use the bounds for model selection.

7.1 How Tight Are the Bounds?

Here we give some idea of how tight the bounds presented in Section 5 are. Denote the function that minimizes the training error (or penalized training error) over $\mathcal{F}$ by $\widehat{f}_{erm}$, and let

$$f^* = \operatorname*{argmin}_{f \in \mathcal{F}} R_n(f)$$

be the oracle predictor with respect to $\mathcal{F}$. We call

$$L_n(\Pi) := \sup_{\mathbb{P} \in \Pi} \mathbb{E}_{\mathbb{P}}[R_n(\widehat{f}_{erm}) - R_n(f^*)] = \sup_{\mathbb{P} \in \Pi} \mathbb{E}_{\mathbb{P}}[R_n(\widehat{f}_{erm})] - R_n(f^*)$$

the "oracle loss"; it describes how well empirical risk minimization works relative to the best possible predictor f^* over the worst distribution $\mathbb{P}$. Vapnik (1998) shows that for classification and i.i.d. data, for sufficiently large n, there exist constants Υ_* and Υ^* such that

$$\Upsilon_* \sqrt{\frac{\mathrm{VCD}(\mathcal{C}_{\mathcal{F}})}{n}} \leq L_n(\Pi) \leq \Upsilon^* \sqrt{\frac{\mathrm{VCD}(\mathcal{C}_{\mathcal{F}}) \log n}{n}},$$

where Π is the class of all distributions on $\mathbb{U} \times \{0,1\}$ and $\mathcal{C}_\mathcal{F}$ has finite VC-dimension $\text{VCD}(\mathcal{C}_\mathcal{F})$. In other words, for i.i.d. data, the best we can hope to do is a rate of

$$O\left(\sqrt{\frac{\text{VCD}(\mathcal{C}_\mathcal{F})}{n}}\right)$$

, and prediction methods which perform worse than $O\left(\sqrt{\text{VCD}(\mathcal{C}_\mathcal{F}) \log n/n}\right)$ are inefficient. We will derive similar bounds for the β-mixing setting. First, we need a slightly different version of Theorem 12.

Theorem 19 *Suppose that $\ell(y-y') < M$, that Assumption A holds, and that $\mathcal{F}$ has a fixed memory length $d < n$. Let μ and a be integers such that $2\mu a + d \leq n$. Then, for all $\epsilon > 0$,*

$$\mathbb{P}\left(\sup_{f\in\mathcal{F}}(R_n(f) - \widehat{R}_n(f)) > \epsilon\right) \leq 8G(n, \mathcal{C}_{\ell\circ\mathcal{F}}) \exp\left\{-\frac{\mu\epsilon^2}{\Upsilon}\right\} + 2\mu\beta_{a-d}.$$

where Υ depends only on M.

The proof of Theorem 19 is exactly like that for Theorem 12 using the result for bounded loss in Theorem 4 rather than that for relative loss in Corollary 23 to control the probability that empirical risk over blocks deviates from its expectation (see (8)).

Assumption D *The time series Y_∞ is* geometrically *β-mixing, that is $\beta_a = \upsilon_1 \exp(-\upsilon_2 a^\kappa)$ for some constants $\upsilon_1, \upsilon_2, \kappa$.*

Theorem 20 *Suppose $\ell(y-y') < M$ and that Assumption D holds. Further assume that $\mathcal{C}_{\ell\circ\mathcal{F}}$ has finite VC-dimension $\text{VCD}(\mathcal{C}_{\ell\circ\mathcal{F}})$. Then, for sufficiently large n, there exist constants Υ_* and Υ^*, independent of n and $\text{VCD}(\mathcal{C}_{\ell\circ\mathcal{F}})$, such that*

$$\Upsilon_*\sqrt{\frac{\text{VCD}(\mathcal{C}_{\ell\circ\mathcal{F}})}{n}} \leq L_n(\Pi) \leq \Upsilon^*\sqrt{\frac{\text{VCD}(\mathcal{C}_{\ell\circ\mathcal{F}}) \log n}{n^{\kappa/(1+\kappa)}}}.$$

If we instead assume *algebraic mixing*, that is $\beta_a = \upsilon a^{-\rho}$, then we can retrieve the same rate where $0 < \kappa < (\rho-1)/2$ (see Meir, 2000). Theorem 20 says that in dependent data settings, using the blocking approach developed here, we may pay a penalty: the upper bound on $L_n(\Pi)$ goes to zero more slowly than in the i.i.d. case. But, the lower bound cannot be made any tighter since i.i.d. processes are still allowed under Assumption D (and of course under the more general Assumption A). In other words, we may have $\kappa \to \infty$ so we can not rule out the faster learning rate of $O\left(\sqrt{\frac{\text{VCD}(\mathcal{C}_{\ell\circ\mathcal{F}}) \log n}{n}}\right)$.

7.2 Structural Risk Minimization

Our presentation so far has focused on choosing one function $\widehat{f}$ from a model $\mathcal{F}$ and demonstrating that the prediction risk $R_n(\widehat{f})$ is well characterized by the training error inflated by a complexity term. The procedure for actually choosing $\widehat{f}$ has been ignored. Common ways of choosing $\widehat{f}$ are frequently referred to as *empirical risk minimization* or ERM: approximate the expected risk $R_n(f)$ with the empirical risk $\widehat{R}_n(f)$, and choose $\widehat{f}$ to minimize

the empirical risk. Many likelihood based methods have exactly this flavor. But more frequently, forecasters have many different models in mind, each with a different empirical risk minimizer. Regularized model classes (ridge regression, lasso, Bayesian methods) implicitly have this structure — altering the amount of regularization leads to different models $\mathcal{F}$. Or one may have many different forecasting models from which the forecaster would like to choose the best. This scenario leads to a generalization of ERM called *structural risk minimization* or SRM.

Given a collection of models $\mathcal{F}_1, \mathcal{F}_2, \ldots$ each with associated empirical risk minimizers $\widehat{f}_1, \widehat{f}_2, \ldots$, we wish to use the function which has the smallest risk. Of course different models have different complexities, and those with larger complexities will tend to have smaller empirical risk. To choose the best function, we therefore penalize the empirical risk and select that function which minimizes the penalized version. Model selection tools like AIC or BIC have exactly this form, but they rely on specific knowledge of the data likelihood and use asymptotics to derive approximate penalties. In contrast, we have finite-sample bounds for the expected risk. This leads to a natural model selection rule: choose the predictor which has the smallest bound on the expected risk.

The generalization error bounds in Section 5 allow one to perform model selection via the SRM principle without knowledge of the likelihood or appeals to asymptotic results. The penalty accounts for the complexity of the model through the VC dimension. Most useful however is that by using generalization error bounds for model selection, we are minimizing the prediction risk. So in the volatility forecasting exercise above, we would choose the mean.

If we want to make the prediction risk as small as possible, we can minimize the generalization error bound simultaneously over models $\mathcal{F}$ and functions within those models. This amounts to treating VC dimension as a control variable. Therefore, by minimizing both the empirical risk and the VC dimension, we can choose that model and function which has the smallest prediction risk, a claim which other model selection procedures cannot make (Massart, 2007; Vapnik, 2000).

8. Conclusion

This paper demonstrates how to control the generalization error of common time-series forecasting models, especially those used in economics and engineering—ARMA models, vector autoregressions (Bayesian or otherwise), linearized dynamic stochastic general equilibrium models, and linear state-space models. We derive upper bounds on the risk, which hold with high probability while requiring only weak assumptions on the data-generating process. These bounds are finite sample in nature, unlike standard model selection penalties such as AIC or BIC. Furthermore, they do not suffer the biases inherent in other risk estimation techniques such as the pseudo-cross validation approach often used in the economic forecasting literature.

While we have stated these results in terms of standard economic forecasting models, they have very wide applicability. Theorem 12 applies to any forecasting procedure with fixed memory length, linear or non-linear. Theorem 17 applies only to methods whose forecasts are linear in the observations, but a similar result for nonlinear methods would

just need to ensure that the dependence of the forecast on the past decays in some suitable way.

Rather than deriving bounds theoretically, one could attempt to estimate bounds on the risk. While cross-validation is tricky (Racine, 2000), nonparametric bootstrap procedures may do better. A fully nonparametric version is possible, using the circular bootstrap (reviewed in Lahiri, 1999). Bootstrapping lengthy out-of-sample sequences for testing fitted model predictions yields intuitively sensible estimates of $R_n(f)$, but there is currently no theory about the coverage level. Also, while models like VARs can be fit quickly to simulated data, general state-space models, let alone DSGEs, require large amounts of computational power, which is an obstacle to any resampling method.

While our results are a crucial first step for the learning-theoretic analysis of time-series forecasts, many avenues remain for future exploration. To gain a more complete picture of the performance of forecasting algorithms, we would want minimax lower bounds (c.f. Tsybakov, 2008). These would tell us the smallest risk we could hope to achieve using any forecaster in some larger model class, letting us ask whether any of the models in common use actually approach this minimum. Another possibility is to target not the *ex ante* risk of the forecast, but the *ex post* regret: how much better might our forecasts have been, in retrospect and on the actually-realized data, had we used a different prediction function from the model $\mathcal{F}$ (Cesa-Bianchi and Lugosi, 2006; Rakhlin et al., 2010)? Remarkably, we can find forecasters which have low *ex post* regret, even if the data came from an adversary trying to make us perform badly. If we target regret rather than risk, we can actually ignore mixing, and even stationarity (Shalizi et al., 2011).

An increased recognition of the abilities and benefits of statistical learning theory can be of tremendous aid to financial and economic forecasters. The results presented here represent an initial yet productive foray in this direction. They allow for principled model comparisons as well as high probability performance guarantees. Future work in this direction will only serve to sharpen our ability to measure predictive power.

Acknowledgments

This work is partially supported by grants from the Institute for New Economic Thinking. DJM is partially supported by NSF grant DMS1407439. CRS is partially supported by NIH Grant R01 NS047493 and by NSF grants DMS1207759 and DMS 1418124. The authors wish to thank David N. DeJong, Larry Wasserman, Alessandro Rinaldo and Darren Homrighausen for valuable suggestions on early drafts. Finally, we wish to think two anonymous referees for their thorough and invaluable comments. Their suggestions have dramatically improved this paper.

Appendix A. Auxiliary Results

Lemma 21 (Yu, 1994 Lemma 4.1) *Let Z be an event with respect to the block sequence $\mathbf{U}$. Then,*

$$|\mathbb{P}(Z) - \widetilde{\mathbb{P}}(Z)| \leq \beta_a(\mu - 1),$$

where the first probability is with respect to the dependent block sequence, $\mathbf{U}$, and $\widetilde{\mathbb{P}}$ is with respect to the independent sequence, $\mathbf{U}'$.

This lemma essentially gives a way to apply i.i.d. results to β-mixing data. Because the dependence decays as we increase the separation between blocks, widely spaced blocks are nearly independent of each other. In particular, the difference between expectations over these nearly independent blocks and expectations over blocks which are actually independent can be controlled by the β-mixing coefficient. Very similar results are also given in Nobel and Dembo (1993) and Vidyasagar (2003).

Lemma 22 (Cortes et al., 2010 Theorem 7) *Under Assumption B,*

$$\mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_n(f)}{Q_d(f)} > \epsilon\sqrt{2 + \log\frac{1}{\epsilon}}\right) \leq 4G(2n, \mathcal{C}_{\ell\circ\mathcal{F}}) \exp\left\{-\frac{n\epsilon^2}{4}\right\}.$$

Corollary 23 *Under Assumption B, for $0 < \epsilon \leq e^{3/2}/\sqrt{2}$,*

$$\mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_n(f)}{Q_d(f)} > \epsilon\right) \leq 4G(2n, \mathcal{C}_{\ell\circ\mathcal{F}}) \exp\left\{-\frac{n\exp\left(W\left(-\frac{2\epsilon^2}{e^4}\right) + 4\right)}{4}\right\}.$$

Proof Letting $\epsilon' = \epsilon\sqrt{2 + \log\frac{1}{\epsilon}}$ and solving for ϵ gives

$$\epsilon = \exp\left\{\frac{1}{2}\left(W\left(-\frac{2\epsilon'^2}{e^4}\right) + 4\right)\right\}$$

if $0 < \epsilon' \leq e^{3/2}/\sqrt{2}$, so

$$\mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_n(f)}{Q_d(f)} > \epsilon\sqrt{2 + \log\frac{1}{\epsilon}}\right) \leq 4G(2n, \mathcal{C}_{\ell\circ\mathcal{F}}) \exp\left\{-\frac{n\epsilon^2}{4}\right\}$$

$$\Rightarrow \mathbb{P}\left(\sup_{f\in\mathcal{F}} \frac{R_n(f) - \widehat{R}_n(f)}{Q_d(f)} > \epsilon\right) \leq 4G(2n, \mathcal{C}_{\ell\circ\mathcal{F}}) \exp\left\{-\frac{n\exp\left(W\left(-\frac{2\epsilon^2}{e^4}\right) + 4\right)}{4}\right\}.$$

■

Lemma 24 *Suppose we have a function class $\mathcal{F}$ with growth function $G(n, \mathcal{F})$. Let $2\mu a = n$. Consider the class $\mathcal{F}^\Sigma = \{\sum_{j=1}^a f(z_j) : f \in \mathcal{F}\}$.*

$$G(\mu, \mathcal{F}^\Sigma) \leq G(n/2, \mathcal{F}).$$

Proof Let $z_1, \ldots, z_\mu \in \mathbb{U}^a$, $b_1, \ldots, b_\mu \in \mathbb{R}$ be a set such that $\exists f_1^\Sigma, \ldots, f_K^\Sigma \in \mathcal{F}^\Sigma$ with $K = G(\mu, \mathcal{F}^\Sigma)$ which shatters$\{z_i\}$. That is, for $v_{ik} \in \{0,1\}$, $i = 1, \ldots, \mu$, $k = 1, \ldots, K$, we have

$$I(f_k^\Sigma(z_i) > b_i) \text{ iff } v_{ik} = 1.$$

But this means, $I(a^{-1}\sum_{j=1}^a f_k(z_{ij}) > b_i/a)$ iff $v_{ik} = 1$. Since this is a convex combination of the $f_k(z_{ij})$, there is a subset of the z_{ij} such that

$$I(f_k(z_{ij}) > b_i/a) \text{ iff } v_{ik} = 1.$$

Thus $G(\mu, \mathcal{F}^\Sigma) \leq G(\mu a, \mathcal{F}) = G(n/2, \mathcal{F})$. ■

Lemma 25 *Let $\mathcal{F}$ be a class of predictor functions $\mathbb{U} \mapsto \mathbb{R}$, and let $h : \mathbb{R} \mapsto \mathbb{R}$, be a monotone-increasing function. Define $h \circ \mathcal{F} = \{h \circ f : f \in \mathcal{F}\}$. Then $\forall n \geq 1$, $G(n, h \circ \mathcal{F}) \leq G(n, \mathcal{F})$.*

Proof Let $z_1, \ldots, z_n \in \mathbb{U}$, $b_1, \ldots, b_n \in \mathbb{R}$ be a set such that $\exists f_1, \ldots, f_K \in \mathcal{F}$ and $K = G(\mu, h \circ \mathcal{F})$ which shatters $\{z_i\}$. That is, for $v_{ik} \in \{0,1\}$, $i = 1, \ldots, n$, $k = 1, \ldots, K$, we have

$$I(h(f_k(z_i)) > b_i) \text{ iff } v_{ik} = 1.$$

Set

$$c_i = \max_{k \in 1:K} \{f_k(z_i) \ : \ h(f_k(u_i)) \leq b_i\}$$

Then, because h is monotone,

$$b_i \geq h(f_k(u_i)) \ \Leftrightarrow \ c_i \geq f_k(u_i)$$

Hence this set of points is shattered by the same collection of functions, and $G(n, h \circ \mathcal{F}) \leq G(n, \mathcal{F})$.

■

Appendix B. Proofs of Selected Results

Proof [Lemma 7] We have that

$$\begin{aligned} b_{ij} &= Z \prod_{k=j+1}^{i-1} L_k K_j \\ &= Z(I - P_{j+1}Z^\top F_{j+1}Z)T(I - P_{j+2}Z^\top F_{j+2}Z) \cdots T(I - P_{i-1}Z^\top F_{i-1}Z)K_j. \end{aligned}$$

Now, by assumption $\lambda_{max}(T) < 1$. Furthermore, $\lambda_{max}\left((I - P_k Z' F_k Z)\right) \leq 1$ for all k. To see this, write

$$\begin{aligned} I - P_k Z^\top F_k Z &= I - P_k Z^\top (Z P_k Z^\top + H)^{-1} Z \\ &= I - Z^\dagger [Z P_k Z^\top][(Z P_k Z^\top + H)^{-1}]Z, \end{aligned}$$

where $Z^\dagger$ is a generalized inverse for Z. As, P_k and H are positive definite, $[ZP_kZ'][(ZP_kZ' + H)^{-1}]$ is positive definite with $\lambda_{\max}([ZP_kZ^\top][(ZP_kZ^\top + H)^{-1}]) < 1$ and therefore $I - Z^\dagger[ZP_kZ^\top][(ZP_kZ^\top + H)^{-1}]Z$ is positive semi-definite if $dim(H) > dim(P_k)$ and positive definite if $dim(P_k) \leq dim(H)$. Therefore, $0 < \lambda_{\max}\left((I - P_kZ'F_kZ)\right) \leq 1$. Finally, $\lambda_{\max}(b_{i,j}) \leq \|Z\|_2 \|T\|_2^{i-j-1} \prod_{k=j+1}^{i-1} \|I - P_kZ'F_kZ\|_2 \leq \lambda_{\max}(T)^{i-j-1} \|Z\|_2 = O(r^{i-j-1})$. ■

Proof [Lemma 8]

1. This result is given in Anderson and Moore (2012, 4.4).

2. The convergence of $\widehat{P}_i$ is in Anderson and Moore (2012, 4.4). As

   $$K_i = T\widehat{P}_i Z^\top (Z\widehat{P}_i Z^\top + H)^{-1}$$

   is continuous in $\widehat{P}_i$, it converges as well.

3. This result is given in Anderson and Moore (2012, 4.4).

4. $\|\widehat{P}_i - \overline{P}\| = O(\lambda_{\max}(T - \overline{K}Z)^i)$ is given Anderson and Moore (2012, 4.4). For F_i,

   $$\|F_i - \overline{F}\| = \|F_i(F_i^{-1} - \overline{F}^{-1})\overline{F}\| = O(\|F_i^{-1} - \overline{F}^{-1}\|)$$

   since $\lambda_{\min}(F_i^{-1}) > \lambda_{\min}(H) > 0 \Rightarrow \lambda_{\max}(F_i) < 1/\lambda_{\min}(H) = O(1)$. Proceeding,

   $$O(\|F_i^{-1} - F^{-1}\|) = O(\|Z\widehat{P}_i Z^\top - Z\overline{P}Z^\top\|) = O(\|\widehat{P}_i - \overline{P}\|) = O(\rho^i).$$

   For K_i, note that

   $$K_i - K = TP_iZ^\top F_i - T\overline{P}Z^\top\overline{F} = TP_iZ^\top F_i - T\overline{P}Z^\top F_i + T\overline{P}Z^\top F_i - T\overline{P}Z^\top\overline{F}.$$

   We have $\|TP_iZ^\top F_i - T\overline{P}Z^\top F_i\| = O(\rho^i)$ by the result for $\widehat{P}_i$ and $\|T\overline{P}Z^\top F_i - T\overline{P}Z^\top\overline{F}\| = O(\rho^i)$ by the result for F_i.

■

Proof [Lemma 9] By (3), $s_{ij} = Z(T - \overline{K}Z)^{i-j-1}\overline{K}$ with $\lambda_{\max}(T - \overline{K}Z) = \rho < 1$. Thus $\|s_{ij}\| = \|s_{i-j-1}\| = O(\rho^{i-j-1})$. For $j < i - 1$,

$$b_{ij} = Z \prod_{k=j+1}^{i-1} (T - K_kZ)K_j \qquad\qquad s_{ij} = Z(T - \overline{K}Z)^{i-j}\overline{K}.$$

Therefore,

$$\begin{aligned}
\|b_{ij} - s_{ij}\| &\leq \|Z\| \left\| \prod_{k=j+1}^{i-1} (T - K_k Z) K_j - (T - \overline{K}Z)^{i-j-1} \overline{K} \right\| \\
&\leq \|Z\| \left\| \prod_{k=j+1}^{i-1} (T - K_k Z) K_j - \prod_{k=j+1}^{i-1} (T - K_k Z) \overline{K} \right\| \\
&\quad + \|Z\| \left\| \prod_{k=j+1}^{i-1} (T - K_k Z) \overline{K} - (T - \overline{K}Z)^{i-j-1} \overline{K} \right\| \\
&\leq \|Z\| \left\| \prod_{k=j+1}^{i-1} (T - K_k Z) \right\| \left\| K_j - \overline{K} \right\| \\
&\quad + \|Z\| \left\| \overline{K} \right\| \left\| \prod_{k=j+1}^{i-1} (T - K_k Z) - (T - \overline{K}Z)^{i-j-1} \right\| \\
&= O\left(\lambda_{\max}(T - \overline{K}Z)^j\right) + O\left(\left\| \prod_{k=j+1}^{i-1} (T - K_k Z) - (T - \overline{K}Z)^{i-j-1} \right\| \right) \quad (9) \\
&= O\left(\lambda_{\max}(T - \overline{K}Z)^j\right) + O\left(\left\| (T - K_{j+1}Z)^{i-j-1} - (T - \overline{K}Z)^{i-j-1} \right\| \right) \quad (10) \\
&= O\left(\lambda_{\max}(T - \overline{K}Z)^j\right).
\end{aligned}$$

Here, (9) follows from Lemma 8 and (10) follows because $\left\|(T - K_k Z) - (T - \overline{K}Z)\right\|$ is increasing in k, so

$$\begin{aligned}
\left\| \prod_{k=j+1}^{i-1} (T - K_k Z) - (T - \overline{K}Z) \right\| &\leq \left\| \prod_{k=j+1}^{i-1} (T - K_{j+1} Z) - (T - \overline{K}Z) \right\| \\
&= \left\| (T - K_{j+1}Z)^{i-j-1} - (T - \overline{K}Z) \right\|.
\end{aligned}$$

For $j = i - 1$,

$$\|b_{i,i-1} - s_{i,i-1}\| = \left\| Z K_j - Z\overline{K} \right\| = O\left(\lambda_{\max}(T - \overline{K}Z)^j\right).$$

■

Proof [Theorem 14] Consider first an AR(d) model and the class

$$\mathcal{C}_{|\mathcal{F}|} = \{(u_0, u_1, u) \in \mathbb{R} \times \mathbb{R} \times \mathbb{R}^d : \mathbf{1}(|u_0 - a^\top u| > u_1) = 1\}.$$

This is the class contains sets of points u for which linear predictions of u_0 are made via a with threshold b. Rewrite $\mathcal{C}_{|\mathcal{F}|}$ as

$$\begin{aligned}
\mathcal{C}_{|\mathcal{F}|} = \Big\{ (u_0, u_1, u) : \max\{ &[\mathrm{sgn}((1 \ -a \ -u_1)^\top (u_0 \ u \ 1))], \\
&[\mathrm{sgn}((-1 \ a \ -u_1)^\top (u_0 \ u \ 1))] \} = 1 \Big\}.
\end{aligned}$$

Then $\mathcal{C}_{|\mathcal{F}|}$ is a 2-combination of $\mathcal{F}$ as in Anthony and Bartlett (1999, Theorem 7.3). As $\mathcal{F}$ is a linear function, it has solution set components bound $B = 1$ (Anthony and Bartlett, 1999, p. 91). We add an intercept to $\mathcal{F}$ so that it will be closed under addition. Then by (Anthony and Bartlett, 1999, Theorem 7.6),

$$G(n, \mathcal{C}_{|\mathcal{F}|}) \leq \left(\frac{2en}{d+1} \right)^{d+1}.$$

Then apply Lemma 25. The result for VARs is similar. Consider the class

$$\begin{aligned}
\mathcal{C}_{\mathcal{G}} = \Big\{ &[\mathrm{sgn}((1 \ -a_1 \ 0 \cdots 0 \ -u_1/k)^\top (u_0 \ u \ 1))] \vee [\mathrm{sgn}((-1 \ a_1 \ 0 \cdots 0 \ -u_1/k)^\top (u_0 \ u \ 1))] \\
&\vee [\mathrm{sgn}((1 \ 0 \ -a_2 \ 0 \cdots 0 \ -u_1/k)^\top (u_0 \ u \ 1))] \vee [\mathrm{sgn}((-1 \ 0 \ a_2 \ 0 \cdots 0 \ -u_1/k)^\top (u_0 \ u \ 1))] \\
&\vee \cdots \vee [\mathrm{sgn}((1 \ 0 \cdots 0 \ -a_k \ -u_1/k)^\top (u_0 \ u \ 1))] \\
&\vee [\mathrm{sgn}((-1 \ 0 \cdots 0 \ a_k \ -u_1/k)^\top (u_0 \ u \ 1))] \Big\}.
\end{aligned}$$

This is the class which correctly classifies at least one of the k coordinates. By (Anthony and Bartlett, 1999, Theorem 7.6),

$$G(n, \mathcal{C}_{\mathcal{G}}) \leq \left(\frac{2ken}{k(d+1)} \right)^{k(d+1)} = \left(\frac{2en}{d+1} \right)^{k(d+1)}.$$

Then apply Lemma 25. As $|\mathcal{F}| \subseteq \mathcal{G}$, we have the result. ■

Proof [Theorem 15 and Corollary 16] Let $\mathcal{F}_2$ be the class of predictors given by Algorithm 2. Let $\mathcal{F}_2'$ be the same class, but predictions are made based on the truncated memory length d. That is for $f_2 \in \mathcal{F}_2$,

$$\widehat{Y}_{d+1:n+1} = \mathbf{S} Y_{1:n}$$

where

$$\mathbf{S} = \begin{bmatrix} s_{d+1,1} & \cdots & s_{d+1,d} & & & 0 \\ s_{d+2,1} & \cdots & s_{d+2,d} & s_{d+2,d+1} & & \\ \vdots & & \vdots & & \ddots & \\ s_{n+1,1} & \cdots & s_{n+1.d} & s_{n+1,d+1} & \cdots & s_{n+1,n} \end{bmatrix} = \begin{bmatrix} s_{d-1} & \cdots & s_0 & & & 0 \\ s_d & \cdots & s_1 & s_0 & & \\ \vdots & & \vdots & & \ddots & \\ s_{n-1} & \cdots & s_{n-d} & s_{n-d+1} & \cdots & s_0 \end{bmatrix}$$

as in (3) (redefining $s_{i,j} \to s_{i-j-1}$), but for $f_2' \in \mathcal{F}_2'$,

$$\widehat{Y}'_{d+1:n+1} = \mathbf{S}' Y_{1:n}$$

where

$$\mathbf{S}' = \begin{bmatrix} s_{d-1} & s_{d-2} & \cdots & s_0 & 0 & & \\ 0 & s_{d-1} & \cdots & s_1 & s_0 & & \\ & & & & \ddots & & \\ & & 0 & & s_{d-1} & \cdots & s_0 \end{bmatrix}. \quad (11)$$

Note that by using Algorithm 2, predictions $\widehat{Y}_i'$ are made using the same linear combination of previous observations at all times i. Thus, truncating the memory, makes these predictions simple autoregressive models of order d.

Define $\widetilde{R}_n(f_2')$ to be the training error of this truncated predictor f_2'. Then, for any $f_2 \in \mathcal{F}_2$,

$$\begin{aligned} R_n(f_2) - \widehat{R}_n(f_2) &= (R_n(f_2) - R_n(f_2')) + (R_n(f_2') - \widetilde{R}_n(f_2')) + (\widetilde{R}_n(f_2') - \widehat{R}_n(f_2)) \\ &= \Delta_1(d) + (R_n(f_2') - \widetilde{R}_n(f_2')) + \Delta_2(d) \\ &= (R_n(f_2') - \widetilde{R}_n(f_2')) + \delta_d(f_2), \end{aligned}$$

where we have defined $\Delta_1(d)$, $\Delta_2(d)$, and $\delta_d(f_2)$ in the obvious way. For now, we will simply proceed with the analysis incorporating the approximation term $\delta_d(f_2)$, before showing that $\delta_d(f_2)$ decays rapidly for this class of models.

Now,

$$R_n(f_2) - \widehat{R}_n(f_2) - \delta_d(f_2) = R_n(f_2') - \widetilde{R}_n(f_2') = R_d(f_2') - \widetilde{R}_n(f_2').$$

Dividing through by $Q_d(f_2') = Q_d(f_2)$ and taking the supremum over $\mathcal{F}_2$ (and therefore over $\mathcal{F}_2'$) gives

$$\sup_{f_2 \in \mathcal{F}_2} \frac{R_n(f_2) - \widehat{R}_n(f_2) - \delta_d(f_2)}{Q_d(f_2)} \leq \sup_{f_2 \in \mathcal{F}_2} \frac{R_n(f_2') - \widetilde{R}_n(f_2')}{Q_d(f_2')}$$

Finally,

$$\mathbb{P}\left(\sup_{f \in \mathcal{F}} \frac{R_n(f) - \widehat{R}_n(f) - \delta_d(f)}{Q_d(f)} > \epsilon \right) \leq \mathbb{P}\left(\sup_{f' \in \mathcal{F}'} \frac{R_n(f') - \widetilde{R}_n(f')}{Q_d(f')} > \epsilon \right)$$

Since $\mathcal{F}'$ is a class with finite memory, we can apply Theorem 12 and Corollary 13 to get the results.

To show that the finite approximation $\delta_d(f)$ decays rapidly for Algorithm 2, consider both components separately. For the case of the difference in expected risks, we need only consider the last rows of $\mathbf{S}$ and $\mathbf{S}'$. As

$$\Delta_1(d) = R_n(f) - R_n(f') = \mathbb{E}[\ell\left(Y_{n+1} - \mathbf{s}_{n+1} Y_{1:n}\right)] - \mathbb{E}[\ell\left(Y_{n+1} - \mathbf{s}'_{n+1} Y_{1:n}\right)],$$

where $\mathbf{s}_{n+1}$ indicates the $(n+1)^{st}$ row of $\mathbf{S}$ and similarly for $\mathbf{s}'_{n+1}$,

$$\begin{aligned} \Delta_1(d) &\leq \mathbb{E}[\ell\left((\mathbf{s}_{n+1} - \mathbf{s}'_{n+1}) Y_{1:n}\right)] && \text{(by the Triangle inequality for } \ell) \\ &= \mathbb{E}\left[\sum_{j=1}^{n-d} \ell(s_{n-j} Y_j) \right] \\ &\leq \sum_{j=1}^{n-d} \ell^*(s_{n-j}) \mathbb{E}[\ell(Y_j)] && \left(\text{as } \ell \text{ is a norm, with } \ell^*(s) = \sup_{x \neq 0} \tfrac{\ell(sx)}{\ell(x)}\right) \\ &= \mathbb{E}[\ell(Y_1)] \left(\sum_{j=1}^{n-d} \ell^*(s_{n-j}) \right) && \text{(by stationarity)} \\ &= \mathbb{E}[\ell(Y_1)] \left(\sum_{j=1}^{n-d} O(\rho^{n-j}) \right) && \text{(Lemma 9 and equivalence of norms)} \\ &= O\left(\frac{\rho^d - \rho^n}{1-\rho} \right) = O(\rho^d). \end{aligned}$$

Similarly,

$$\begin{aligned} \Delta_2(d) &= \widetilde{R}_n(f') - \widehat{R}_n(f) \\ &\leq \frac{1}{n-d} \sum_{i=d+1}^{n} \ell((\mathbf{s}_i - \mathbf{s}'_i) Y_{1:i-1}) \\ &= \frac{1}{n-d} \sum_{i=d+2}^{n} \sum_{j=1}^{i-d-1} \ell(s_{i-j-1} Y_j) \\ &\leq \frac{1}{n-d} \sum_{i=d+2}^{n} O_p(1) \sum_{j=1}^{i-d-1} \ell^*(s_{i-j-1}) \\ &\leq O_{\mathbb{P}}((n-d)^{-1}) \sum_{i=d+2}^{n} \sum_{j=1}^{i-d-1} O(\rho^{i-j-1}) \\ &= O_{\mathbb{P}}((n-d)^{-1}) O\left[\frac{(n-d-1)\rho^d - (n-d)\rho^{d+1} + \rho^n}{(1-\rho)^2} \right] \\ &= O_{\mathbb{P}}(\rho^d) \end{aligned}$$

■

Proof [Theorem 17 and Corollary 18] Let $\mathcal{F}_1$ be the class of predictors given by Algorithm 1. Let $\mathcal{F}_2'$ be the class of predictors given by Algorithm 2 based on the truncated memory length d. That is for $f_1 \in \mathcal{F}_1$,

$$\widehat{Y}_{d+1:n+1} = \mathbf{B} Y_{1:n}$$

where

$$\mathbf{B} = \begin{bmatrix} b_{d,1} & \cdots & b_{d,d} & & \\ b_{d+1,1} & \cdots & b_{d+1,d} & b_{d+1,d+1} & \mathbf{0} \\ \vdots & & \vdots & & \ddots \\ b_{n+1,1} & \cdots & b_{n+1,d} & b_{n+1,d+1} & \cdots \quad b_{n+1,n} \end{bmatrix}$$

as in (2), but for f_2'

$$\widehat{Y}'_{d+1:n+1} = \mathbf{S}' Y_{1:n}$$

as in (11).

Then, proceeding as in the previous proof, define $\widetilde{R}_n(f_2')$ to be the training error of the associated truncated limiting predictor f_2' and write analogously

$$\begin{aligned} R_n(f_1) - \widehat{R}_n(f_1) &= (R_n(f_1) - R_n(f_2')) + (R_n(f_2') - \widetilde{R}_n(f_2')) + (\widetilde{R}_n(f_2') - \widehat{R}_n(f_1)) \\ &= \Delta_1(d) + (R_n(f_2') - \widetilde{R}_n(f_2')) + \Delta_2(d) \\ &= (R_n(f_2') - \widetilde{R}_n(f_2')) + \delta_d(f_1), \end{aligned}$$

where

$$\delta_d(f_1) = (R_n(f_1) - R_n(f_2')) + (\widetilde{R}_n(f_2') - \widehat{R}_n(f_1)) = \Delta_1(d) + \Delta_2(d).$$

Now, by Lemma 9, the truncated version of f_1, f_1' converges to f_2' so that $f_1'(Y_{n-d:n-1})$ converges in distribution to $f_2'(Y_{n-d:n-1})$. And because$\ell(y-x)^2$ is continuous in x, $\ell(Y_n - f_1'(Y_{n-d:n-1}))^2$ converges in distribution to $\ell(Y_n - f_2'(Y_{n-d:n-1}))^2$ by the continuous mapping theorem. By Assumption C, $\ell(Y_n - f_1'(Y_{n-d:n-1}))^2$ is uniformly integrable, so

$$\lim_{n\to\infty} \mathbb{E}\left[\ell(Y_n - f_1'(Y_{n-d:n-1}))^2\right] = \mathbb{E}\left[\ell(Y_n - f_2'(Y_{n-d:n-1}))^2\right].$$

Therefore,

$$\frac{R_n(f_1) - \widehat{R}_n(f_1) - \delta_d(f_1)}{\lim_{n\to\infty} Q_d(f_1)} = \frac{R_n(f_2') - \widetilde{R}_n(f_2')}{Q_d(f_2')}$$

For each $f_1 \in \mathcal{F}_1$ (parameterized by θ), there exists one function $f_2 \in \mathcal{F}_2$ which results from the same parameter vector, so that

$$\sup_{f_1\in\mathcal{F}_1} \frac{R_n(f_1) - \widehat{R}_n(f_1) - \delta_d(f_1)}{\lim_{n\to\infty} Q_d(f_1)} \leq \sup_{f_2'\in\mathcal{F}_2'} \frac{R_n(f_2') - \widetilde{R}_n(f_2')}{Q_d(f_2')}.$$

Therefore,

$$\mathbb{P}\left(\sup_{f_1\in\mathcal{F}_1} \frac{R_n(f_1) - \widehat{R}_n(f_1) - \delta_d(f_1)}{\lim_{n\to\infty} Q_d(f_1)} > \epsilon\right) \leq \mathbb{P}\left(\sup_{f_2'\in\mathcal{F}_2'} \frac{R_n(f_2') - \widetilde{R}_n(f_2')}{Q_d(f_2')} > \epsilon\right)$$

Since $\mathcal{F}_2'$ is a class with finite memory, we can apply Theorem 12 and Corollary 13 to get the results.

To show that the finite approximation $\delta_d(f_1)$ decays rapidly, consider both components separately. For the case of the difference in expected risks, we need only consider the last rows of $\mathbf{B}$ and $\mathbf{S}'$. As

$$\Delta_1(d) = R_n(f_1) - R_n(f_2') = \mathbb{E}[\ell\left(Y_{n+1} - \mathbf{b}_{n+1}Y_{1:n}\right)] - \mathbb{E}[\ell\left(Y_{n+1} - \mathbf{s}'_{n+1}Y_{1:n}\right)],$$

where $\mathbf{b}_{n+1}$ indicates the $(n+1)^{st}$ row of $\mathbf{B}$ and similarly for $\mathbf{s}'_{n+1}$. Then, as in the previous proof,

$$\begin{aligned} \Delta_1(d) &\leq \mathbb{E}[\ell\left((\mathbf{b}_{n+1} - \mathbf{s}'_{n+1})Y_{1:n}\right)] && (12) \\ &\leq \mathbb{E}\left[\left(\sum_{j=1}^{n-d} \ell(b_{n+1,j}Y_j) + \sum_{j=n-d+1}^{n} \ell((b_{n+1,j} - s_{n-j})Y_j)\right)\right] \\ &\leq \sum_{j=1}^{n-d} \ell^*(b_{n+1,j})\mathbb{E}[\ell(Y_j)] + \sum_{j=n-d+1}^{n} \ell^*(b_{n+1,j} - s_{n-j})\mathbb{E}[\ell(Y_j)] && (13) \\ &= \mathbb{E}[\ell(Y_1)]\left(\sum_{j=1}^{n-d} \ell^*(b_{n+1,j}) + \sum_{j=n-d+1}^{n} \ell^*(b_{n+1,j} - s_{n-j})\right) \\ &= \mathbb{E}[\ell(Y_1)]\left(\sum_{j=1}^{n-d} O(r^{n-j}) + \sum_{j=n-d+1}^{n} O(\rho^j)\right) \\ &= O\left(\frac{r^d - r^n}{1-r}\right) + O\left(\frac{\rho^{n-d+1}(1-\rho^d)}{1-\rho}\right) \\ &= O(r^d) + O(\rho^{n-d+1}). \end{aligned}$$

(12) and B are via the triangle inequality for ℓ, while (13) uses the fact that ℓ is a norm, with $\ell^*(b) = \sup_{x \neq 0} \frac{\ell(bx)}{\ell(x)}$. Similarly,

$$\begin{aligned}
\Delta_2(d) &= \widetilde{R}_n(f_2') - \widehat{R}_n(f_1) \\
&\leq \frac{1}{n-d} \sum_{i=d+1}^{n} \ell((\mathbf{b}_i - \mathbf{s}_i')Y_{1:n-1}) \\
&\leq \frac{1}{n-d} \sum_{i=d+1}^{n} \sum_{j=1}^{i-d-1} \ell(b_{ij} Y_j) + \frac{1}{n-d} \sum_{i=d+1}^{n} \sum_{j=i-d}^{i-1} \ell((b_{ij} - s_{n-j}) Y_j) \\
&\leq \frac{1}{n-d} \sum_{i=d+2}^{n} O_{\mathbb{P}}(1) \sum_{j=1}^{i-d-1} \ell^*(b_{ij}) + \frac{1}{n-d} \sum_{i=d+1}^{n} O_{\mathbb{P}}(1) \sum_{j=i-d}^{i-1} \ell^*(b_{ij} - s_{ij}) \\
&\leq O_{\mathbb{P}}((n-d)^{-1}) \sum_{i=d+2}^{n} \sum_{j=1}^{i-d-1} r^{i-j-1} + O_{\mathbb{P}}((n-d)^{-1}) \sum_{i=d+1}^{n} \sum_{j=i-d}^{i-1} \rho^j \\
&= O_{\mathbb{P}}((n-d)^{-1}) O\left[\frac{(n-d-1)r^d - (n-d)r^{d+1} + r^n}{(1-r)^2}\right] \\
&\quad + O_{\mathbb{P}}((n-d)^{-1}) \left[\frac{(1-r^d)(r^d - r^n)}{r^{d-1}(1-r)^2}\right] \\
&= O_{\mathbb{P}}(r^d) + O_{\mathbb{P}}((n-d)^{-1})
\end{aligned}$$

■

Proof [Theorem 20] Theorem 19 implies that simultaneously

$$\mathbb{P}\left(R_n(\widehat{f}_{erm}) - \widehat{R}_n(\widehat{f}_{erm}) > \epsilon\right) \leq 8G(n, \mathcal{C}_{\ell \circ \mathcal{F}}) \exp\left\{-\frac{\mu\epsilon^2}{\Upsilon}\right\} + 2\mu\beta_{a-d}$$

and

$$\mathbb{P}\left(R_n(f^*) - \widehat{R}_n(f^*) > \epsilon\right) \leq 8G(n, \mathcal{C}_{\ell \circ \mathcal{F}}) \exp\left\{-\frac{\mu\epsilon^2}{\Upsilon}\right\} + 2\mu\beta_{a-d}.$$

Since $\widehat{R}_n(\widehat{f}_{erm}) - \widehat{R}_n(f^*) \leq 0$, then

$$\mathbb{P}\left(R_n(\widehat{f}_{erm}) - R_n(f^*) > \epsilon\right) \leq 16G(n, \mathcal{C}_{\ell \circ \mathcal{F}}) \exp\left\{-\frac{\mu\epsilon^2}{\Upsilon}\right\} + 4\mu\beta_{a-d}.$$

For any nonnegative random variable Z, $\mathbb{E}[Z] = \int_0^\infty \mathbb{P}(Z > \epsilon) d\epsilon$, so

$$\begin{aligned}
L_n^2(\Pi) &= \int_0^\infty d\epsilon \mathbb{P}\left((R_n(\widehat{f}_{erm}) - R_n(f^*))^2 > \epsilon\right) \\
&= \int_0^\xi d\epsilon \mathbb{P}\left((R_n(\widehat{f}_{erm}) - R_n(f^*))^2 > \epsilon\right) + \int_\xi^\infty d\epsilon \mathbb{P}\left(R_n(\widehat{f}_{erm}) - R_n(f^*) > \epsilon\right) \\
&= \int_0^\xi d\epsilon \mathbb{P}\left((R_n(\widehat{f}_{erm}) - R_n(f^*))^2 > \epsilon\right) + \int_\xi^\infty d\epsilon \mathbb{P}\left((R_n(\widehat{f}_{erm}) - R_n(f^*))^2 > \sqrt{\epsilon}\right) \\
&= \int_0^\xi d\epsilon \mathbb{P}\left((R_n(\widehat{f}_{erm}) - R_n(f^*))^2 > \epsilon\right) + \int_\xi^{M^2} d\epsilon \mathbb{P}\left(R_n(\widehat{f}_{erm}) - R_n(f^*) > \sqrt{\epsilon}\right) \quad (14) \\
&\leq \xi + \int_\xi^{M^2} d\epsilon \left[16G(n, \mathcal{C}_{\ell \circ \mathcal{F}}) \exp\left\{-\frac{\mu\epsilon}{\Upsilon}\right\} + 4\mu\beta_{a-d}\right] \\
&\leq \xi + \int_\xi^\infty d\epsilon 16G(n, \mathcal{C}_{\ell \circ \mathcal{F}}) \exp\left\{-\frac{\mu\epsilon}{\Upsilon}\right\} + \int_\xi^{M^2} d\epsilon 4\mu\beta_{a-d} \\
&= \xi + \frac{16\Upsilon G(n, \mathcal{C}_{\ell \circ \mathcal{F}}) \exp\left\{-\frac{\mu\xi}{\Upsilon}\right\}}{\mu} + 4(M^2 - \xi)\mu\beta_{a-d}
\end{aligned}$$

for all $0 < \xi < \Upsilon$. Here, (14) follows because $\mathbb{P}\left(R_n(\widehat{f}_{erm}) - R_n(f^*) > M\right) = 0$. Using Assumption D, take

$$a = (n-d)^{1/(1+\kappa)} = O(n^{1/(1+\kappa)}), \qquad \mu = .5(n-d)^{\kappa/(1+\kappa)} = O(n^{\kappa/(1+\kappa)}),$$

and $\xi = \frac{\log G(n, \mathcal{C}_{\ell \circ \mathcal{F}})}{n^{\kappa/(1+\kappa)}}$ to balance the exponential and linear terms. Then

$$L_n(\Pi) = O\left(\sqrt{\frac{\log G(n, \mathcal{C}_{\ell \circ \mathcal{F}})}{n^{\kappa/(1+\kappa)}}}\right) = O\left(\sqrt{\frac{\mathrm{VCD}(\mathcal{C}_{\ell \circ \mathcal{F}}) \log n}{n^{\kappa/(1+\kappa)}}}\right).$$

as $\mathcal{C}_{\ell \circ \mathcal{F}}$ has finite VC-dimension.

For the lower bound, apply the i.i.d. version, as classification is a special case of bounded regression. The result follows. ■

References

Terrence M. Adams and Andrew B. Nobel. Uniform convergence of Vapnik-Chervonenkis classes under ergodic sampling. *Annals of Probability*, 38:1345–1367, 2010.

Alekh Agarwal and John C. Duchi. The generalization ability of online algorithms for dependent data. *IEEE Transactions on Information Theory*, 59(1):573–587, 2013.

Pierre Alquier, Olivier Wintenberger, et al. Model selection for weakly dependent time series forecasting. *Bernoulli*, 18(3):883–913, 2012.

Pierre Alquier, Xiaoyin Li, and Olivier Wintenberger. Prediction of time series by statistical learning: general losses and fast rates. *Dependence Modelling*, 1:65–93, 2014.

Brian D.O. Anderson and John B. Moore. *Optimal filtering*. Prentice-Hall, Englewood Cliffs, NJ, 2012.

Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, Cambridge, England, 1999.

Peter L. Bartlett and Shahar Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

Richard C. Bradley. Basic properties of strong mixing conditions: A survey and some open questions. *Probability Surveys*, 2:107–144, 2005.

Marine Carrasco and Xiaohong Chen. Mixing and moment properties of various GARCH and stochastic volatility models. *Econometric Theory*, 18:17–39, 2002.

Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, Cambridge, England, 2006.

Robert M. Corless, Gaston H. Gonnet, D. E. G. Hare, David J. Jeffrey, and Donald E. Knuth. On the Lambert W function. *Advances in Computational Mathematics*, 5:329–359, 1996.

Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weights. In John Lafferty, C. K. I. Williams, John Shawe-Taylor, Richard S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing (NIPS)*, pages 442–450. MIT Press, 2010.

David N. DeJong and Chetan Dave. *Structural Macroeconometrics*. Princeton University Press, Princeton, New Jersey, 2007.

David N. DeJong, Beth F. Ingram, and Charles H. Whiteman. A Bayesian approach to dynamic macroeconomics. *Journal of Econometrics*, 98:203–223, 2000.

Paul Doukhan. *Mixing: Properties and Examples*. Springer-Verlag, New York, 1995.

James Durbin and Siem Jam Koopman. *Time Series Analysis by State Space Methods*. Oxford University Press, Oxford, 2001.

Robert F. Engle. Autoregressive conditional heteroscedasticity with estimates of variance of United Kingdom inflation. *Econometrica*, 50:987–1008, 1982.

Robert F. Engle. Garch 101: The use of arch/garch models in applied econometrics. *Journal of Economic Perspectives*, 15:157–168, 2001.

Jesús Fernández-Villaverde. The econometrics of DSGE models. *SERIEs*, 1:3–49, 2010.

Andrew Harvey, Esther Ruiz, and Neil Shephard. Multivariate stochastic variance models. *The Review of Economic Studies*, 61:247–264, 1994.

Rudolf E. Kalman. A new approach to linear filtering and prediction problems. *ASME Transactions, Journal of Basic Engineering*, 82D:35–50, 1960.

Rajeeva L. Karandikar and Mathukumalli Vidyasagar. Rates of uniform convergence of empirical means with mixing processes. *Statistics and Probability Letters*, 58:297–307, 2002.

Michael J. Kearns and Dana Ron. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Neural Computation*, 11:1427–1453, 1999.

Bas J. K. Kleijn and Aad W. van der Vaart. Misspecification in infinite-dimensional Bayesian statistics. *Annals of Statistics*, 34:837–877, 2006.

Vladimir Koltchinskii and Dmitriy Panchenko. Empirical margin distributions and bounding the generalization error of combined classifiers. *Annals of Statistics*, 30(1):1–50, 02 2002.

Vitaly Kuznetsov and Mehryar Mohri. Generalization bounds for time series prediction with non-stationary processes. In Peter Auer, Alexander Clark, Thomas Zeugmann, and Sandra Zilles, editors, *Algorithmic Learning Theory (ALT)*, pages 260–274, 2014.

Finn E. Kydland and Edward C. Prescott. Time to build and aggregate fluctuations. *Econometrica*, 50(6):1345–1370, November 1982.

Soumendra N. Lahiri. Theoretical comparisons of block bootstrap methods. *Annals of Statistics*, 27:386–404, 1999.

Pascal Massart. *Concentration Inequalities and Model Selection*. Springer-Verlag, Berlin, 2007.

Daniel J. McDonald, Cosma Rohilla Shalizi, and Mark Schervish. Estimating β-mixing coefficients. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 516–524, 2011a.

Daniel J. McDonald, Cosma Rohilla Shalizi, and Mark Schervish. Risk bounds for autoregressive models. Technical report, Statistics Department, CMU, 2011b.

Daniel J. McDonald, Cosma Rohilla Shalizi, and Mark Schervish. Estimating beta-mixing coefficients via histograms. *Electronic Journal of Statistics*, 9:2855–2883, 2015.

Ron Meir. Nonparametric time series prediction through adaptive model selection. *Machine Learning*, 39:5–34, 2000.

Dharmendra S. Modha and Elias Masry. Memory-universal prediction of stationary random processes. *IEEE Transactions on Information Theory*, 44(1):117–133, 1998.

Mehryar Mohri and Afshin Rostamizadeh. Rademacher complexity bounds for non-I.I.D. processes. In Daphne Koller, D. Schuurmans, Y. Bengio, and Léon Bottou, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1104, 2009.

Mehryar Mohri and Afshin Rostamizadeh. Stability bounds for stationary ϕ-mixing and β-mixing processes. *Journal of Machine Learning Research*, 11, 2010.

Abdelkader Mokkadem. Mixing properties of ARMA processes. *Stochastic Processes and their Applications*, 29(2):309–315, 1988.

Ulrich K. Müller. Risk of Bayesian inference in misspecified models, and the sandwich covariance matrix. *Econometrica*, 81:1805–1849, 2013.

Andrew Nobel and Amir Dembo. A note on uniform laws of averages for dependent processes. *Statistics and Probability Letters*, 17:169–172, 1993.

David Pollard. *Convergence of Stochastic Processes*. Springer-Verlag, Berlin, 1984.

David Pollard. *Empirical Processes: Theory and Applications*, volume 2 of *NSF-CBMS Regional Conference Series in Probability and Statistics*. Institute of Mathematical Statistics, Hayward, California, 1990.

Jeff Racine. Consistent cross-validatory model-selection for dependent data: hv-block cross-validation. *Journal of Econometrics*, 99:39–61, 2000.

Alexander Rakhlin, Karthik Sridharan, and Ambuj Tewari. Online learning: Random averages, combinatorial parameters, and learnability. In John Lafferty, C. K. I. Williams, John Shawe-Taylor, Richard S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing (NIPS)*, pages 1984–1992. MIT Press, 2010.

David Romer. *Advanced Macroeconomics*. McGraw-Hill, New York, 4 edition, 2011.

Cosma Rohilla Shalizi. Dynamics of Bayesian updating with dependent data and misspecified models. *Electronic Journal of Statistics*, 3:1039–1074, 2009.

Cosma Rohilla Shalizi, Abigail Z. Jacobs, Kristina Lisa Klinkner, and Aaron Clauset. Adapting to non-stationarity with growing expert ensembles. Technical report, Statistics Department, CMU, 2011.

Christopher A. Sims. Solving linear rational expectations models. *Computational Economics*, 20:1–20, 2002.

Frank Smets and Rafael Wouters. Shocks and frictions in US business cycles: A Bayesian DSGE approach. *American Economic Review*, 97(3):586–606, June 2007.

Ingo Steinwart and Andreas Christmann. Fast learning from non-i.i.d. observations. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22, pages 1768–1776. MIT Press, Cambridge, MA, 2009.

Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Verlag, New York, 2008.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, 2nd edition, 2000.

Vladimir N. Vapnik and Alexey Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16: 264–280, 1971.

Mathukumalli Vidyasagar. *Learning and Generalization: With Applications to Neural Networks*. Springer-Verlag, Berlin, second edition, 2003.

Bin Yu. Rates of convergence for empirical processes of stationary mixing sequences. *Annals of Probability*, 22:94–116, 1994.

Journal of Machine Learning Research 18 (2017) 1-39 Submitted 5/14; Revised 4/16; Published 4/17

Online Bayesian Passive-Aggressive Learning*

Tianlin Shi TIANLINS@CS.STANFORD.EDU
Institute for Interdisciplinary Information Sciences
Tsinghua University
Beijing, 100084 China

Jun Zhu DCSZJ@TSINGHUA.EDU.CN
State Key Lab of Intelligent Technology and Systems
Tsinghua National Lab for Information Science and Technology
Department of Computer Science and Technology
Tsinghua University
Beijing, 100084 China

Editor: Amir Globerson

Abstract

We present online Bayesian Passive-Aggressive (BayesPA) learning, a generic online learning framework for hierarchical Bayesian models with max-margin posterior regularization. We show that BayesPA subsumes the standard online Passive-Aggressive (PA) learning and extends naturally to incorporate latent variables for both parametric and nonparametric Bayesian inference, therefore providing great flexibility for explorative analysis. As an important example, we apply BayesPA to topic modeling and derive efficient online learning algorithms for max-margin topic models. We further develop nonparametric BayesPA topic models to infer the unknown number of topics in an online manner. Experimental results on 20newsgroups and a large Wikipedia multi-label dataset (with 1.1 millions of training documents and 0.9 million of unique terms in the vocabulary) show that our approaches significantly improve time efficiency while achieving comparable accuracy with the corresponding batch algorithms.

1. Introduction

With the fast growth of data volume and variety, it is becoming increasingly important to develop scalable machine learning algorithms, which can be categorized into two major categories — online/stochastic methods on a single core and parallel/distributed algorithms on multiple-cores or multiple machines. This paper focuses on online learning, a process of answering a sequence of questions (e.g., which category does a document belong to?) given knowledge of the correct answers (e.g., the true category labels) to previous questions. Such a process is especially suitable for the applications with streaming data. For the applications with large datasets, online learning algorithms can effectively explore data redundancy relative to the model to be learned, by repeatedly subsampling the data; and they often lead to faster convergence to satisfactory results than the corresponding batch learning algorithms. Among the many popular algorithms, online Passive-Aggressive (PA) learning (Crammer et al., 2006) provides a generic framework of performing online learning for large-margin methods (e.g., SVMs), with many applications in natural language processing and text mining (McDonald et al., 2005; Chiang et al., 2008). Though enjoying strong discriminative ability that is preferable for predictive tasks, existing online PA methods are formulated as a point estimate problem by optimizing some deterministic objective function. This may lead to some potential shortcomings. For example, a single large-margin model could fail to capture the rich underlying structure of complex data.

On the other hand, Bayesian methods enjoy greater flexibility in describing the possible underlying structures of complex data by incorporating a hierarchy of latent variables. Moreover, the recent progress on nonparametric Bayesian methods (Hjort, 2010; Teh et al., 2006a) further provides an increasingly important framework that allows the Bayesian models to have an unbounded model complexity, e.g., an infinite number of components in a mixture model (Hjort, 2010) or an infinite number of units in a latent feature model (Ghahramani and Griffiths, 2005), and to adapt when the learning environment changes. In particular, adaptation to the changing environment is of great importance in online learning. For Bayesian models, one challenging problem is posterior inference, for which both variational and Monte Carlo methods can be too expensive to be applied to large-scale applications. To scale up Bayesian inference, much progress has been made on developing stochastic variational Bayes (Hoffman et al., 2013; Mimno et al., 2012) and stochastic gradient Monte Carlo (Welling and Teh, 2011; Ahn et al., 2012) methods, which repeatedly draw samples from a given finite dataset.[1] To deal with the potentially unbounded streaming data, streaming variational Bayes methods (Broderick et al., 2013) have been developed as a general framework, with an application to topic models for learning latent topic representations. However, due to the generative nature, Bayesian models usually does not perform as well as discriminative models in predictive tasks such as classification.

Successful attempts have been made to bring large-margin learning and Bayesian methods together to solve supervised learning tasks using flexible Bayesian latent variable models. For example, maximum entropy discrimination (MED) made a significant advance in conjoining max-margin learning and Bayesian generative models, in the context of univariate prediction (Jaakkola et al., 1999) and structured output prediction (Zhu and Xing, 2009). Recently, much attention has been devoted to generalizing MED to incorporate latent variables and perform nonparametric Bayesian inference in various contexts, including topic modeling (Zhu et al., 2012), matrix factorization (Xu et al., 2013), and multi-task learning (Jebara, 2011; Zhu et al., 2014c). Regularized Bayesian inference (RegBayes) (Zhu et al., 2014c) provides a unified framework for Bayesian models to perform max-margin learning, where the max-margin principle is incorporated through adding posterior constraints to an information-theoretical optimization problem. By solving this problem involving latent variables, RegBayes can uncover structures that support the supervised learning tasks. RegBayes subsumes the standard Bayes' rule and is more flexible in incorporating domain knowledge or max-margin constraints. Though flexible in discovering latent structures and powerful in discriminative predictions, posterior inference in such models remains a challenge. By exploring data augmentation techniques, recent progress has been made to develop efficient MCMC methods (Zhu et al., 2014b), which can also be implemented in distributed clusters (Zhu et al., 2013). However, these batch learning methods are not applicable to streaming data, and they do not explore the statistical redundancy in large-scale corpora either.

*. Correspondence should be addressed to J. Zhu. T. Shi is now with Stanford Artificial Intelligence Laboratory (SAIL), Stanford University, Stanford, CA, 94305

1. There are also many distributed methods for scalable Bayesian inference. Please see (Zhu et al., 2014a) for a review.

©2017 Tianlin Shi and Jun Zhu.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/14-188.html.

To address the above problems of both online PA on incorporating flexible latent structures and Bayesian max-margin models on scalable streaming inference, this paper presents online Bayesian Passive-Aggressive (BayesPA) learning, a general framework of performing online learning for Bayesian max-margin models. We show that online BayesPA subsumes the standard online PA when the underlying model is linear and the parameter prior is Gaussian (See Table 1 for its close relationships with streaming variational Bayes and RegBayes). We further show that one major significance of BayesPA is its natural generalization to incorporate a hierarchy of latent variables for both parametric and nonparametric Bayesian inference, therefore allowing online BayesPA to have the great flexibility of (nonparametric) Bayesian methods for explorative analysis as well as the strong discriminative ability of large-margin learning for predictive tasks. As concrete examples, we apply the theory of online BayesPA to topic modeling and derive efficient online learning algorithms for max-margin supervised topic models (Zhu et al., 2012). We further develop efficient online learning algorithms for the nonparametric max-margin topic models, an extension of the nonparametric topic models (Teh et al., 2006a) for predictive tasks. Extensive empirical results on real datasets demonstrate significant improvements on time efficiency and maintenance of comparable results with corresponding batch algorithms.

The paper is structured as follows. We discuss the related work in Section 2, and review the preliminary knowledge in Section 3. Then, we move on to the detailed description of BayesPA in Section 4, and present the theoretical analysis. Section 5 presents the concrete instantiations on topic modeling, and Section 6 presents the extensions to nonparametric topic models and multi-task learning. Section 7 presents experimental results. Finally, Section 8 concludes this paper with future directions discussed.

2. Related Work

As a well-established learning paradigm, online learning is of both theoretical and practical interest. The goal of online learning is to make a sequence of decisions, such as classification and regression, and use the knowledge extracted from previous correct answers to produce decisions on incoming ones. The root of online learning could be traced back to early studies of repeated games (Hannan, 1957), where an agent dynamically makes choices with the summary of past information. The idea became popular with the advent of Perceptron algorithms (Rosenblatt, 1958), which adopt an additive update rule for the classifier weights, and its multiplicative counterpart is the Winnow algorithm (Littlestone, 1988). The class of online multiplicative algorithms was further generalized by Adaboost (Freund and Schapire, 1997) in a decision theoretic sense and is now widely applied to various fields of study (Arora et al., 2012).

First presented by Crammer et al. (2006) as a weight updating method, online Passive-Aggressive (PA) algorithms provide a generic online learning framework for max-margin models. In particular, they considered loss functions that enforce max-margin constraints, and showed that surprisingly simple update rules could be derived in closed forms. Motivated by online PA learning and to handle unbalanced training sets, Dredze et al. (2008) proposed confidence-weighted learning, which maintains a Gaussian distribution of the classifier weights at each round and replaces the max-margin constraint in PA with a probabilistic constraint enforcing confidence of classification. Within the same framework, Crammer et al. (2008) derived a new convex form of the constraint and demonstrated performance improvements through empirical evaluations.

The theoretical analysis of online learning typically relies on the notion of *regret*, which is the average loss incurred by an adaptive online learner on streaming data versus the best achievable through a single fixed model having the hindsight of all data (Murphy, 2012). It can be shown that the notion of regret is closely related to *weak duality* in convex optimization, which brings online learning to the algorithmic framework of convex repeated games (Shalev-Shwartz and Singer, 2006).

Although the classical regime of online learning is based on decision theory, recently much attention has been paid to the theory and practice of online probabilistic inference in the context of Big Data. Rooted either in variational inference or Monte Carlo sampling methods, there are broadly two lines of work on the topic of online Bayesian inference. Stochastic variational inference (SVI) (Hoffman et al., 2013) is a stochastic approximation algorithm for mean-field variational inference. By approximating the natural gradients in maximizing the evidence lower bound with stochastic gradients sampled from data points, Hoffman et al. (2013) demonstrated scalable inference of topic models on large corpora. Mimno et al. (2012) showed the performance of SVI could be improved through structured mean-field assumptions and locally collapsed variational inference. SVI is also applicable to the stochastic inference of nonparametric Bayesian models, such as hierarchical Dirichlet process (Wang et al., 2011; Wang and Blei, 2012b).

There is also a large body of work on extending Monte Carlo methods to the online setting. A classic approach is sequential Monte Carlo methods (SMC) or particle filters (Doucet and Johansen, 2009), which arose from the numerical estimation of state-space models. Through Rao-Blackwellized particle filters (Doucet et al., 2000), one could obtain online inference algorithms for latent Dirichlet allocation (LDA) (Canini et al., 2009), and more recently Gao et al. (2016) presented a streaming (collapsed) Gibbs sampler with better accuracy. To tackle the sparsity issues and inadequate coverage of particles, Steinhardt and Liang (2014) leveraged "abstract particles" to represent entire regions of the sample space. Recently, Korattikara et al. (2014) introduced an approximate Metropolis-Hasting rule based on sequential hypothesis testing that allows accepting and rejecting samples using only a fraction of the data. As an alternative, Bardenet et al. (2014) proposed an adaptive sampling strategy of Metropolis-Hastings from a controlled perturbation of the target distribution. With elegant use of gradient information that Metropolis-Hastings algorithms neglected, a line of work (Welling and Teh, 2011; Ahn et al., 2012; Patterson and Teh, 2013) also developed stochastic gradient methods based on Langevin dynamics.

While most online Bayesian inference methods have adopted a *stochastic approximation* of the posterior distribution by sub-sampling a given finite dataset, in many applications data arrives in stream so that the dataset is changing over time and its size is unknown. To relax the previous request on knowing the data size, Broderick et al. (2013) made streaming updates to the estimated posterior and demonstrated the advantage of streaming variational Bayes (SVB) over stochastic variational inference. As will be discussed in this paper, BayesPA also does not impose assumptions on the size of dataset and works on streaming data.

The idea to discriminatively train univariate or structured output classifiers was popularized by the seminal work on support vector machines (Vapnik, 1995) and max-margin Markov networks (aka structural SVMs) (Taskar et al., 2003). Since then, research on developing max-margin models with latent variables has received increasing attention, because of the promise to capture underlying complex structures of the problems. A promising line of work focused on Bayesian approaches, and one representative is maximum entropy discrimination (MED) (Jaakkola et al., 1999; Jebara, 2001; Zhu and Xing, 2009), which learns distributions of model parameters discriminatively from labeled

data. MedLDA (Zhu et al., 2012) extends MED to infer latent topical structure from data with large margin constraints on the target posteriors. Similarly, nonparametric Bayesian max-margin models have also been developed, such as infinite SVMs (iSVM) (Zhu et al., 2011) for building SVM classifiers with a latent mixture structure, and infinite latent SVMs (iLSVM) (Zhu et al., 2014c) for automatic discovering predictive features for SVMs. Furthermore, the idea of nonparametric Bayesian learning has been applied to link prediction (Zhu, 2012), matrix factorization (Xu et al., 2013), etc. Regularized Bayesian inference (RegBayes) (Zhu et al., 2014c) provides a unified framework for performing max-margin learning of (nonparametric) Bayesian models, where the max-margin principle was incorporated through imposing posterior constraints to a variational formulation of the standard Bayes' rule.

Max-margin Bayesian learning in the batch mode has already been one of the common challenges facing this class of models. Despite its general intractability, efficient algorithms have been proposed under different settings. One way is to solve the problem via variational inference under a mean-field (Zhu et al., 2012) or structured mean-field (Jiang et al., 2012) assumption. Recently, Zhu et al. (2014b) provided a key insight in deriving efficient Monte Carlo methods without making strict assumptions. Their technique is based on a data augmentation formulation of the expected margin loss. Based on similar techniques, fast inference algorithms have also been developed for generalized relational topic models (Chen et al., 2015), matrix factorization (Xu et al., 2013), etc. Data augmentation (DA) refers to the method of introducing augmented variables along with the observed data to make their joint distribution tractable. The technique was popularized in the statistics community by the well-known expectation-maximization algorithm (EM) (Dempster et al., 1977) for maximum likelihood estimation with missing data. For posterior inference, the technique is popularized by Tanner and Wong (1987) in statistics and by Swendsen and Wang (1987) for the Ising and Potts models in physics. For a broader introduction to DA methods, we refer the readers to Van Dyk and Meng (2001).

Finally, our conference version of the paper (Shi and Zhu, 2014) has introduced some preliminary work, which we largely extend here.

3. Preliminaries

This section reviews the preliminary knowledge that is needed to develop online Bayesian Passive-Aggressive learning. The relationships with BayesPA will be summarized in Table 1 later.

3.1 Online Passive-Aggressive Learning

Based on a decision-theoretic view, the goal of online supervised learning is to minimize the cumulative loss for a certain prediction task from the sequentially arriving training samples. Online Passive-Aggressive (PA) learning (Crammer et al., 2006) achieves this goal by updating some parametric model $\boldsymbol{w} \in \mathbb{R}^K$ (e.g., the weights of a linear SVM) in an online manner with the instantaneous losses from arriving data $\{\boldsymbol{x}_t\}_{t\geq 0}$ ($\boldsymbol{x}_t \in \mathbb{R}^K$) and corresponding responses $\{y_t\}_{t\geq 0}$, where K is the number of input features. The loss $\ell_\epsilon(\boldsymbol{w}; \boldsymbol{x}_t, y_t)$, as they consider, could be the hinge loss $(\epsilon - y_t \boldsymbol{w}^\top \boldsymbol{x}_t)_+$ for binary classification ($y_t \in \{0, 1\}$) or the ϵ-insensitive loss $(|y_t - \boldsymbol{w}^\top \boldsymbol{x}_t| - \epsilon)_+$ for regression ($y_t \in \mathbb{R}$), where ϵ is a hyper-parameter and $(x)_+ = \max(0, x)$. The online Passive-Aggressive update rule is then derived by defining the new weight $\boldsymbol{w}_{t+1}$ as the solution to the following optimization problem:

$$\min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{w} - \boldsymbol{w_t}||^2 \quad \text{s.t.: } \ell_\epsilon(\boldsymbol{w}; \boldsymbol{x}_t, y_t) = 0, \tag{1}$$

where $\|\cdot\|$ is the Euclidean 2-norm. Intuitively, if $\boldsymbol{w_t}$ suffers no loss from the new data, i.e., $\ell_\epsilon(\boldsymbol{w}_t; \boldsymbol{x}_t, y_t) = 0$, the algorithm *passively* assigns $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t$; otherwise, it aggressively projects $\boldsymbol{w_t}$ to the feasible region of parameter vectors that attain zero loss on the new data. With provable regret bounds, Crammer et al. (2006) showed that online PA algorithms could achieve comparable results to the optimal classifier $\boldsymbol{w}^*$, which has the hindsight of all data. In practice, in order to account for inseparable training samples, soft margin constraints are often adopted and the resulting PA learning problem is

$$\min_{\boldsymbol{w}} \frac{1}{2}||\boldsymbol{w} - \boldsymbol{w_t}||^2 + 2c \cdot \ell_\epsilon(\boldsymbol{w}; \boldsymbol{x}_t, y_t), \tag{2}$$

where c is a positive regularization parameter and the constant factor 2 is included for simplicity as will be clear soon. For problems (1) and (2) with samples arriving one at a time, closed-form solutions can be derived (Crammer et al., 2006). For example, for the binary hinge loss the update rule for problem (2) is $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \tau_t y_t \boldsymbol{x}_t$, where $\tau_t = \min(2c, \max(0, \epsilon - y_t \boldsymbol{w}^\top \boldsymbol{x}_t)/||\boldsymbol{x}_t||^2)$; and for the ϵ-insensitive loss, the update rule is $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \text{sign}(y_t - \boldsymbol{w}^\top \boldsymbol{x}_t)\tau_t \boldsymbol{x}_t$, where $\tau_t = \max(0, \epsilon - y_t \boldsymbol{w}^\top \boldsymbol{x}_t)/||\boldsymbol{x}_t||^2$.

3.2 Streaming Variational Bayes

For Bayesian models, Bayes' rule naturally leads to a streaming update procedure for online learning. Specifically, suppose the data $\{\boldsymbol{x}_t\}_{t\geq 0}$ are generated i.i.d. according to a distribution $p(\boldsymbol{x}|\boldsymbol{w})$ and the prior $p(\boldsymbol{w})$ is given. Bayes' theorem implies that the posterior distribution of $\boldsymbol{w}$ given the first T samples ($T \geq 1$) satisfies

$$p(\boldsymbol{w}|\{\boldsymbol{x}_t\}_{t=0}^T) \propto p(\boldsymbol{w}|\{\boldsymbol{x}_t\}_{t=0}^{T-1}) p(\boldsymbol{x}_T|\boldsymbol{w}). \tag{3}$$

In other words, the posterior after observing the first $T-1$ samples is treated as the prior for the incoming data point.

This natural streaming Bayes' rule, however, is often intractable to compute, especially for complex models (e.g., when latent variables are present). Streaming variational Bayes (SVB) (Broderick et al., 2013) suggests that a variational approximation should be adopted and it practically works well. Specifically, let $\mathcal{A}$ be any algorithm that calculates an approximate posterior $q(\boldsymbol{w}) = \mathcal{A}(\boldsymbol{X}, q_0(\boldsymbol{w}))$ based on data $\boldsymbol{X}$ and prior $q_0(\boldsymbol{w})$. Then, the recursive formula for approximate streaming update is:

$$q(\boldsymbol{w}|\{\boldsymbol{x}_t\}_{t=0}^T) = \mathcal{A}\left(\boldsymbol{x}_T, q(\boldsymbol{w}|\{\boldsymbol{x}_t\}_{t=0}^{T-1})\right).$$

The choice of $\mathcal{A}$ can be problem-specific. For topic modeling, Broderick et al. (2013) showed that one may adopt mean-field variational Bayes (Wainwright and Jordan, 2008), expectation propagation (Minka, 2001), and one-pass posterior approximation algorithms using stochastic variational inference (Hoffman et al., 2013) or sufficient statistics update (Honkela and Valpola, 2003; Luts et al., 2013). By applying the streaming update in a distributed setting asynchronously, SVB could also be scaled up across multiple computer clusters (Broderick et al., 2013).

3.3 Regularized Bayesian Inference

The decision-theoretic and Bayesian view of learning can be reciprocal. For example, it would be beneficial to combine the flexibility of Bayesian models with the discriminative power of large-margin methods. The idea of regularized Bayesian inference (RegBayes) (Zhu et al., 2014c) is to treat Bayesian inference as an optimization problem with an imposed loss function for prediction tasks or a regularization term on the target posterior distribution in general. Mathematically, RegBayes for supervised learning can be formulated as

$$\min_{q\in\mathcal{P}} \ \mathbf{KL}\Big[q(\boldsymbol{w})||p_0(\boldsymbol{w})\Big] - \mathbb{E}_{q(\boldsymbol{w})}\Big[\log p(\boldsymbol{X}|\boldsymbol{w})\Big] + 2c\cdot\ell\Big(q(\boldsymbol{w});\boldsymbol{X},\boldsymbol{Y}\Big), \tag{4}$$

where $\mathcal{P}$ is the probability simplex, $p(\boldsymbol{X}|\boldsymbol{w})$ is the likelihood function and $\mathbf{KL}$ is the Kullback-Leibler divergence.[2] Note that if the loss is zero, i.e., $\ell(q(\boldsymbol{w});\boldsymbol{X},\boldsymbol{Y})=0$, then the optimal solution is $q^*(\boldsymbol{w}) \propto p_0(\boldsymbol{w})p(\boldsymbol{X}|\boldsymbol{w})$, which is just the Bayes' rule. However, when $\ell(q(\boldsymbol{w});\boldsymbol{X},\boldsymbol{Y})\neq 0$, RegBayes biases the inferred posterior towards discriminating the supervising side-information $\boldsymbol{Y}$, with the parameter c controlling the extent of regularization. If the posterior regularization term $\ell(\cdot)$ is a convex function of $q(\boldsymbol{w})$ through the linear expectation operator, Zhu et al. (2014c) presented a general representation theorem to characterize the solution to problem (4). To distinguish from the posterior distribution obtained via Bayes' rule, the solution to problem (4) is called *post-data posterior* (Zhu et al., 2014c). Many instantiations have been developed following the generic framework of RegBayes to demonstrate its superior performance than standard Bayesian models in various settings, such as topic modeling (Jiang et al., 2012; Zhu et al., 2014b), matrix factorization (Xu et al., 2013) and link prediction (Zhu, 2012), as well as the flexibility of performing robust Bayesian inference with noisy domain knowledge (Mei et al., 2014).

4. Bayesian Passive-Aggressive Learning

In this section, we present online Bayesian Passive-Aggressive (BayesPA) learning, a general perspective on online max-margin Bayesian inference. Without loss of generality, we consider binary classification. The techniques can be applied to other learning tasks. We provide an extension in Section 6.2.

4.1 Online BayesPA Learning

Instead of updating a point estimate of $\boldsymbol{w}$, online Bayesian PA (BayesPA) sequentially infers a new post-data posterior distribution $q_{t+1}(\boldsymbol{w})$, either parametric or nonparametric, on the arrival of a new data point $(\boldsymbol{x}_t, y_t)$ by solving the following optimization problem:

$$\begin{aligned}\min_{q(\boldsymbol{w})\in\mathcal{F}_t} \ & \mathbf{KL}\Big[q(\boldsymbol{w})||q_t(\boldsymbol{w})\Big] - \mathbb{E}_{q(\boldsymbol{w})}\Big[\log p(\boldsymbol{x}_t|\boldsymbol{w})\Big] \\ \text{s.t.:} \ & \ell_\epsilon\Big(q(\boldsymbol{w});\boldsymbol{x}_t,y_t\Big)=0,\end{aligned} \tag{5}$$

2. We assume that the model space $\boldsymbol{W}$ is a complete separable metric space endowed with its Borel σ-algebra $\mathcal{B}(\boldsymbol{W})$. Let P_0 and Q be probability measures on $\boldsymbol{W}$. The Kullback-Leibler (KL) divergence of the probability measure Q with respect to the measure P_0 is defined as $\mathbf{KL}[Q\|P_0] = \int \frac{\mathrm{d}Q}{\mathrm{d}P_0}(\boldsymbol{w})\log\frac{\mathrm{d}Q}{\mathrm{d}P_0}(\boldsymbol{w})\mathrm{d}P_0(\boldsymbol{w})$, where $\frac{\mathrm{d}Q}{\mathrm{d}P_0}(\boldsymbol{w})$ is the Radon-Nikodym derivative (Durret, 2010). It is required that Q is absolutely continuous with respect to P_0 such that this derivative exists. In this paper, we further assume that P_0 is absolutely continuous with respect to some background measure μ. Thus, there exists a density p_0 that satisfies $\mathrm{d}P_0 = p_0\mathrm{d}\mu$ and there also exists a density q that satisfies $\mathrm{d}Q = q\mathrm{d}\mu$. Then, the KL-divergence can be expressed as $\mathbf{KL}[q\|p_0] = \int q(\boldsymbol{w})\log\frac{q(\boldsymbol{w})}{p_0(\boldsymbol{w})}\mathrm{d}\mu(\boldsymbol{w})$.

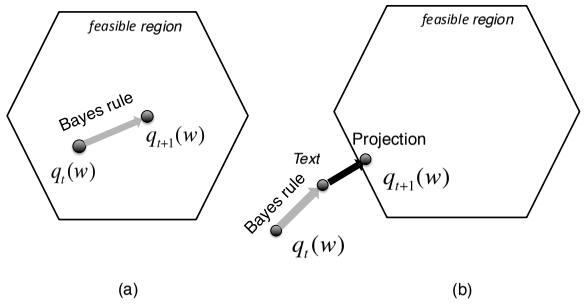


Figure 1: Graphical Illustration of BayesPA learning. **(a)**. Update *passively* by Bayes rule, if the resulting distribution suffer zero loss. **(b)** Otherwise, *aggressively* project the resulting distribution to the feasible region of weights with zero loss.

where $\mathcal{F}_t$ can be some family of distributions or the whole probability simplex $\mathcal{P}$. For notational convenience, we denote the objective as $\mathcal{L}(q(\boldsymbol{w}))$. In words, we find a post-data posterior distribution $q_{t+1}(\boldsymbol{w})$ in the feasible region that is not only close to the current posterior $q_t(\boldsymbol{w})$ in terms of KL-divergence, but also has a high likelihood of explaining the new data. As a result, if Bayes' rule already gives the posterior distribution $q_{t+1}(\boldsymbol{w}) \propto q_t(\boldsymbol{w})p(\boldsymbol{x}_t|\boldsymbol{w})$ that suffers no loss (i.e., $\ell_\epsilon = 0$), BayesPA *passively* updates the posterior following just Bayes' rule;[3] otherwise, BayesPA *aggressively* projects the new posterior to the feasible region of posteriors that attain zero loss. The passive and aggressive update cases are illustrated in Figure 1. We should note that when no likelihood is defined (e.g., $p(\boldsymbol{x}_t|\boldsymbol{w})$ is independent of $\boldsymbol{w}$), BayesPA will passively set $q_{t+1}(\boldsymbol{w}) = q_t(\boldsymbol{w})$ if $q_t(\boldsymbol{w})$ suffers no loss; otherwise, it will aggressively project $q_t(\boldsymbol{w})$ to the feasible region. We call it *non-likelihood* BayesPA.

In practical problems, the constraints in (5) could be unrealizable. To deal with such cases, we introduce the soft-margin version of BayesPA learning, which is equivalent to minimizing the objective function $\mathcal{L}(q(\boldsymbol{w}))$ in problem (5) with a regularization term, similar as in SVMs (Cortes and Vapnik, 1995):

$$q_{t+1}(\boldsymbol{w}) = \operatorname*{argmin}_{q(\boldsymbol{w})\in\mathcal{F}_t} \mathcal{L}\Big(q(\boldsymbol{w})\Big) + 2c\cdot\ell_\epsilon\Big(q(\boldsymbol{w});\boldsymbol{x}_t,y_t\Big). \tag{6}$$

For the max-margin classifiers that we consider, two types of loss functionals $\ell_\epsilon(q(\boldsymbol{w});\boldsymbol{x}_t,y_t)$ are common:

3. Under the Bayesian setting, we regard Bayes' rule as a natural update (passive), since no aggressive projection is applied.

Methods	Max-margin learning ?	Bayesian inference ?	Streaming update ?
PA	yes	no	yes
SVB	no	yes	yes
RegBayes	yes	yes	no
BayesPA	yes	yes	yes

Table 1: The comparison between BayesPA and its various precursors, including online PA, streaming variational Bayes (SVB) and regularized Bayesian inference (RegBayes), in three different aspects.

1. **Averaging classifier**: assume that a post-data posterior distribution $q(\boldsymbol{w})$ is given, then an averaging classifier makes predictions using the sign rule $\hat{y}_t = \text{sign}\ \mathbb{E}_{q(\boldsymbol{w})}[\boldsymbol{w}^\top \boldsymbol{x}_t]$ when the discriminant function has the simple linear form, $f(\boldsymbol{x}_t; \boldsymbol{w}) = \boldsymbol{w}^\top \boldsymbol{x}_t$. For this classifier, its hinge loss is therefore defined as:

$$\ell_\epsilon^{\text{ave}}\Big(q(\boldsymbol{w}); \boldsymbol{x}_t, y_t\Big) = \Big(\epsilon - y_t \mathbb{E}_{q(\boldsymbol{w})}\left[\boldsymbol{w}^\top \boldsymbol{x}_t\right]\Big)_+ .$$

2. **Gibbs classifier**: assume that a post-data posterior distribution $q(\boldsymbol{w})$ is given, then a Gibbs classifier randomly draws a weight vector $\boldsymbol{w} \sim q(\boldsymbol{w})$ to make predictions using the sign rule $\hat{y}_t = \text{sign}\ \boldsymbol{w}^\top \boldsymbol{x}_t$, when the discriminant function has the same linear form as above. For each single $\boldsymbol{w}$, we can measure its hinge loss $(\epsilon - y_t \boldsymbol{w}^\top \boldsymbol{x}_t)_+$. To account for the randomness of $\boldsymbol{w}$, the expected hinge loss of a Gibbs classifier is therefore defined as:

$$\ell_\epsilon^{\text{gibbs}}\Big(q(\boldsymbol{w}); \boldsymbol{x}_t, y_t\Big) = \mathbb{E}_{q(\boldsymbol{w})}\left[\Big(\epsilon - y_t \boldsymbol{w}^\top \boldsymbol{x}_t\Big)_+\right].$$

They are closely connected via the following lemma due to the convexity of the function $(x)_+$.

Lemma 1 *The expected hinge loss ℓ_ϵ^{gibbs} is an upper bound of the hinge loss ℓ_ϵ^{ave}, that is,*

$$\ell_\epsilon^{gibbs}\Big(q(\boldsymbol{w}); \boldsymbol{x}_t, y_t\Big) \geq \ell_\epsilon^{ave}\Big(q(\boldsymbol{w}); \boldsymbol{x}_t, y_t\Big).$$

BayesPA is deeply connected to its various precursors reviewed in Section 3, as summarized in Table 1. First, BayesPA is a natural Bayesian extension of online PA, which is explicated via the following theorem. Note that this theorem only serves as a connection to PA, and will not be relied on by our algorithms, where we are more interested in the cases with nontrivial likelihood models to describe the underlying structures (e.g., topic structures), which are typically out-of-reach for the standard PA. Nevertheless, the idea of the proof details would later be applied to develop practical BayesPA algorithms for topic models. Therefore, we include the complete proof here.

Theorem 2 *If the prior is normal $q_0(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}_0, I)$, $\mathcal{F}_t = \mathcal{P}$, and we use the averaging classifier ℓ_ϵ^{ave}, the non-likelihood BayesPA subsumes the online PA.*

Proof The soft-margin version of BayesPA learning can be reformulated using a slack variable ξ_t:

$$\begin{aligned} q_{t+1}(\boldsymbol{w}) = \quad & \underset{q(\boldsymbol{w})\in\mathcal{P}}{\text{argmin}}\ \text{KL}\Big[q(\boldsymbol{w})||q_t(\boldsymbol{w})\Big] + 2c\cdot\xi_t \\ & \text{s.t.}: y_t \mathbb{E}_{q(\boldsymbol{w})}\left[\boldsymbol{w}^\top \boldsymbol{x}_t\right] \geq \epsilon - \xi_t,\ \ \xi_t \geq 0. \end{aligned} \tag{7}$$

Similar to Corollary 5 in (Zhu et al., 2012), the optimal solution $q^*(\boldsymbol{w})$ of the above problem can be derived from its functional Lagrangian and has the following form:

$$q^*(\boldsymbol{w}) = \frac{1}{\Gamma(\tau_t^*, \boldsymbol{x}_t, y_t)} q_t(\boldsymbol{w}) \exp\Big(\tau_t^* y_t \boldsymbol{w}^\top \boldsymbol{x}_t\Big), \tag{8}$$

where the normalization term $\Gamma(\tau_t, \boldsymbol{x}_t, y_t) = \int_{\boldsymbol{w}} q_t(\boldsymbol{w}) \exp\Big(\tau_t y_t \boldsymbol{w}^\top \boldsymbol{x}_t\Big) d\boldsymbol{w}$, and τ_t^* is the optimal solution to the dual problem

$$\begin{aligned} & \max_{\tau_t}\ \epsilon\tau_t - \log\Gamma(\tau_t, \boldsymbol{x}_t, y_t) \\ & \text{s.t.}\ 0 \leq \tau_t \leq 2c. \end{aligned} \tag{9}$$

Using this primal-dual interpretation, we prove that for the normal prior $p_0(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}_0, I)$, the post-data posterior is also Gaussian: $q_t(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{\mu}_t, I)$ for some $\boldsymbol{\mu}_t$ in each round $t = 0, 1, 2, ...$ This can be shown by induction. By our assumption, $q_0(\boldsymbol{w}) = p_0(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}_0, I)$ is Gaussian. Assume for round $t \geq 0$, the distribution $q_t(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{\mu}_t, I)$. Then for round $t+1$, Eq. (8) suggests the distribution

$$q_{t+1}(\boldsymbol{w}) = \frac{\mathcal{C}}{\Gamma(\tau_t^*, \boldsymbol{x}_t, y_t)} \exp\left(-\frac{1}{2}||\boldsymbol{w} - (\boldsymbol{\mu}_t + \tau_t^* y_t \boldsymbol{x}_t)||^2\right),$$

where the constant $\mathcal{C} = \exp(y_t \tau_t^* \boldsymbol{\mu}_t^\top \boldsymbol{x}_t + \frac{1}{2}\tau_t^{*2} \boldsymbol{x}_t^\top \boldsymbol{x}_t)$. Therefore, the distribution $q_{t+1}(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{\mu}_t + \tau_t^* y_t \boldsymbol{x}_t, I)$, and the normalization term is $\Gamma(\tau_t, \boldsymbol{x}_t, y_t) = (\sqrt{2\pi})^K \exp(\tau_t y_t \boldsymbol{x}_t^\top \boldsymbol{\mu}_t + \frac{1}{2}\tau_t^2 \boldsymbol{x}_t^\top \boldsymbol{x}_t)$ for any $\tau_t \in [0, 2c]$.

Next, we show that $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + \tau_t^* y_t \boldsymbol{x}_t$ is the optimal solution of the online PA update rule (Crammer et al., 2006). To see this, we replace $\Gamma(\tau_t, \boldsymbol{x}_t, y_t)$ in problem (9) with our derived form. Ignoring constant terms, we obtain the dual problem

$$\begin{aligned} & \max_{\tau_t}\ \epsilon\tau_t - \tfrac{1}{2}\tau_t^2 \boldsymbol{x}_t^\top \boldsymbol{x}_t - y_t \tau_t \boldsymbol{\mu}_t^\top \boldsymbol{x}_t \\ & \text{s.t.:}\ 0 \leq \tau_t \leq 2c, \end{aligned} \tag{10}$$

which is exactly the dual form of the online PA update equation:

$$\begin{aligned} \boldsymbol{\mu}_{t+1}^{\text{PA}} = & \arg\min_{\boldsymbol{\mu}}\ \tfrac{1}{2}||\boldsymbol{\mu} - \boldsymbol{\mu}_t||^2 + 2c\cdot\xi_t \\ & \text{s.t.}\ y_t \boldsymbol{\mu}^\top \boldsymbol{x}_t \geq \epsilon - \xi_t,\ \ \xi_t \geq 0. \end{aligned}$$

The optimal solution is $\boldsymbol{\mu}_{t+1}^{\text{PA}} = \boldsymbol{\mu}_t + \tau_t^* y_t \boldsymbol{x}_t$. Note that τ_t^* is the optimal solution of dual problem (10) shared by both PA and BayesPA. Therefore, we conclude that $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_{t+1}^{\text{PA}}$. ■

Second, suppose some algorithm $\mathcal{A}$ is capable of solving problem (6), then it would produce streaming updates to the posterior distribution. For averaging classifiers, it is easy to modify the proof of theorem 2 to derive the update rule of BayesPA, which is presented in the following lemma.

Lemma 3 *If $\mathcal{F}_t = \mathcal{P}$ and we use the averaging classifier with loss functional ℓ_ϵ^{ave}, the update rule of online BayesPA is*

$$q_{t+1}(\boldsymbol{w}) = \frac{1}{\Gamma(\tau_t^*, \boldsymbol{x}_t, y_t)} q_t(\boldsymbol{w}) p(\boldsymbol{x}_t|\boldsymbol{w}) \exp\Big(\tau_t^* y_t \boldsymbol{w}^\top \boldsymbol{x}_t\Big), \tag{11}$$

where $\Gamma(\tau_t, \boldsymbol{x}_t, y_t)$ is the normalization term $\Gamma(\tau_t, \boldsymbol{x}_t, y_t) = \int_{\boldsymbol{w}} q_t(\boldsymbol{w}) p(\boldsymbol{x}_t|\boldsymbol{w}) \exp\Big(\tau_t y_t \boldsymbol{w}_t^\top \boldsymbol{x}_t\Big) d\boldsymbol{w}$, and τ_t^* is the optimal solution to the dual problem

$$\begin{array}{ll} \max\limits_{\tau_t} & \epsilon\tau_t - \log\Gamma(\tau_t, \boldsymbol{x}_t, y_t) \\ \text{s.t.} & 0 \le \tau_t \le 2c. \end{array}$$

For Gibbs classifiers, we have the following lemma to characterize its streaming update rule.

Lemma 4 *If $\mathcal{F}_t = \mathcal{P}$ and we use the Gibbs classifier with loss functional ℓ_ϵ^{gibbs}, the update rule of online BayesPA is*

$$q_{t+1}(\boldsymbol{w}) = \frac{q_t(\boldsymbol{w}) p(\boldsymbol{x}_t|\boldsymbol{w}) \exp\Big(-2c\big(\epsilon - y_t \boldsymbol{w}^\top \boldsymbol{x}_t\big)_+\Big)}{\Gamma(\boldsymbol{x}_t, y_t)}, \tag{12}$$

where $\Gamma(\boldsymbol{x}_t, y_t)$ is the normalization constant.

In both update rules (11) and (12), the post-data posterior $q_t(\boldsymbol{w})$ in the previous round t can be treated as a prior, while the newly observed data and the loss it incurs provide a likelihood and an un-normalized pseudo-likelihood respectively. Due to the analytical form, the BayesPA update rule (12) is sequential in nature, simiar as the convential Bayes rule (3). Therefore, the sequential posterior at time T is the same as the batch posterior that observes all the data instances up to T. For the case with averaging classifers, Eq. (11) reduces to the streaming Bayesian update problem if there is no loss functional (i.e., $\ell_\epsilon = 0$). Therefore, BayesPA is an extension to streaming variational Bayes (SVB) with imposed max-margin posterior constraints.

Finally, the update formulation (6) is essentially a RegBayes problem with a single data point $(\boldsymbol{x}_t, y_t)$. Although RegBayes inference is normally intractable, we would show later in the paper how to use variational approximation to bypass the difficulty for specific settings. This would lead to variational approximation algorithm $\mathcal{A}$ for the streaming update of post-data posterior.

Besides treating a single data point at a time, a useful technique in practice to reduce the noise in data is to use mini-batches. Suppose that we have a mini-batch of data points at time t with an index set B_t. Let $\boldsymbol{X}_t = \{\boldsymbol{x}_d\}_{d\in B_t}, \boldsymbol{Y}_t = \{y_d\}_{d\in B_t}$. The online BayesPA update equation for this mini-batch can be defined in a natural way:

$$\min_{q\in\mathcal{F}_t} \mathbf{KL}\Big[q(\boldsymbol{w})||q_t(\boldsymbol{w})\Big] - \mathbb{E}_{q(\boldsymbol{w})}\Big[\log p(\boldsymbol{X}_t|\boldsymbol{w})\Big] + 2c\cdot\ell_\epsilon\Big(q(\boldsymbol{w}); \boldsymbol{X}_t, \boldsymbol{Y}_t\Big),$$

where $\ell_\epsilon(q(\boldsymbol{w}); \boldsymbol{X}_t, \boldsymbol{Y}_t) = \sum_{d\in B_t} \ell_\epsilon(q(\boldsymbol{w}); \boldsymbol{x}_d, y_d)$ is the cumulative loos functional on the mini-batch B_t. Like the vanilla PA methods (Crammer et al., 2006), BayesPA on mini-batches may not have closed-form update rules, and numerical optimization methods are needed to solve this new formulation.

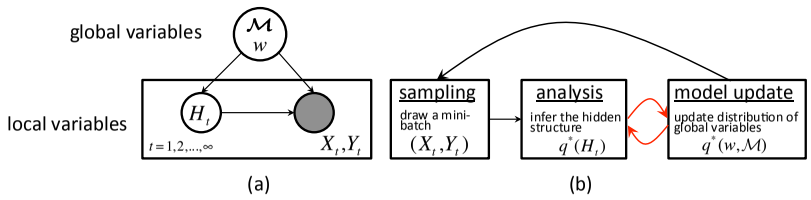


Figure 2: Graphical illustrations of: (a) the abstraction of models with latent structures; and (b) the procedure of BayesPA learning with latent structures.

4.2 BayesPA Learning with Latent Structures

To expressively explain complex real-word data, Bayesian models with latent structures have been extensively developed. The latent structures could typically be characterized by a hierarchy of variables, which are generally grouped into two sets—*local latent variables* $\boldsymbol{h}_d$ $(d \ge 0)$ that characterize the hidden structures of each observed data $\boldsymbol{x}_d$ and *global variables* $\boldsymbol{\mathcal{M}}$ that capture the common properties shared by all data.

As illustrated in Figure 2, BayesPA learning with latent structures aims to update the distribution of $\boldsymbol{\mathcal{M}}$ as well as the classifier weights $\boldsymbol{w}$, based on each incoming mini-batch $(\boldsymbol{X}_t, \boldsymbol{Y}_t)$ and their corresponding latent variables $\boldsymbol{H}_t = \{\boldsymbol{h}_d\}_{d\in B_t}$. Because of the uncertainty in $\boldsymbol{H}_t$, our posterior approximation algorithm $\mathcal{A}$ would first infer the joint posterior distribution $q_{t+1}(\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t)$ from

$$\min_{q\in\mathcal{F}_t} \mathcal{L}\Big(q(\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t)\Big) + 2c\cdot\ell_\epsilon\Big(q(\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t); \boldsymbol{X}_t, \boldsymbol{Y}_t\Big), \tag{13}$$

where $\mathcal{L}(q) = \mathrm{KL}[q(\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t)||q_t(\boldsymbol{w}, \boldsymbol{\mathcal{M}})p_0(\boldsymbol{H}_t)] - \mathbb{E}_{q(\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t)}[\log p(\boldsymbol{X}_t|\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t)]$ and $\ell_\epsilon(q(\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t); \boldsymbol{X}_t, \boldsymbol{Y}_t)$ is some cumulative loss functional on the min-batch data incurred by some classifiers on the latent variables $\boldsymbol{H}_t$ and/or global variables $\boldsymbol{\mathcal{M}}$. As in the case without latent variables, both averaging classifier and Gibbs classifier can be used.

Next, algorithm $\mathcal{A}$ produces the approximate posterior $q_{t+1}(\boldsymbol{w}, \boldsymbol{\mathcal{M}})$. In general we would not obtain a closed-form posterior distribution by marginalizing out $\boldsymbol{H}_t$, especially in dealing with some involved models like MedLDA (Zhu et al., 2012). The intractability is bypassed through the mean-field assumption $q(\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t) = q(\boldsymbol{w})q(\boldsymbol{\mathcal{M}})q(\boldsymbol{H}_t)$. Specifically, algorithm $\mathcal{A}$ solves problem (13) using an iterative procedure and obtain the optimal distribution $q^*(\boldsymbol{w})q^*(\boldsymbol{\mathcal{M}})q^*(\boldsymbol{H}_t)$. Then it sets $q_{t+1}(\boldsymbol{w}, \boldsymbol{\mathcal{M}}) = q^*(\boldsymbol{w})q^*(\boldsymbol{\mathcal{M}})$ and proceeds to next round. Concrete examples of this method will be discussed in Section 5 and Section 6.

5. Online Max-Margin Topic Models

In this section, we apply the theory of online BayesPA to topic modeling. We first review the basic ideas of max-margin topic models, and develop online learning algorithms based on BayesPA with averaging and Gibbs classifiers respectively.

5.1 Basics of MedLDA

A max-margin topic model consists of a latent Dirichlet allocation (LDA) model (Blei et al., 2003) for describing the underlying topic representations of document content and a max-margin classifier for making predictions. Specifically, LDA is a hierarchical Bayesian model that treats each document as an admixture of K topics, $\boldsymbol{\Phi} = \{\boldsymbol{\phi}_k\}_{k=1}^K$, where each topic $\boldsymbol{\phi}_k$ is a multinomial distribution over a given W-word vocabulary.[4] The generative process of the d-th document ($1 \leq d \leq D$) is described as follows:

- Draw a topic mixture proportion vector $\boldsymbol{\theta}_d|\boldsymbol{\alpha} \sim \text{Dir}(\boldsymbol{\alpha})$
- For the i-th word in document d, where $i = 1, 2, ..., n_d$,
  - draw a latent topic assignment $z_{di} \sim \text{Mult}(\boldsymbol{\theta}_d)$.
  - draw the word instance $x_{di} \sim \text{Mult}(\boldsymbol{\phi}_{z_{di}})$.

where $\text{Dir}(\cdot)$ is the Dirichlet distribution and $\text{Mult}(\cdot)$ is the multinomial distribution. For Bayesian LDA, the topics are also drawn from a Dirichlet distribution, i.e., $\boldsymbol{\phi}_k \sim \text{Dir}(\boldsymbol{\gamma})$.

Given a document set $\boldsymbol{X} = \{\boldsymbol{x}_d\}_{d=1}^D$. Let $\boldsymbol{Z} = \{\boldsymbol{z}_d\}_{d=1}^D$ and $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_d\}_{d=1}^D$ denote all the topic assignments and topic mixing vectors. LDA infers the posterior distribution $p(\boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z}|\boldsymbol{X}) \propto p_0(\boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z})p(\boldsymbol{X}|\boldsymbol{Z}, \boldsymbol{\Phi})$ via Bayes' rule. From a variational point of view, the posterior is identical to the solution of the optimization problem:

$$\min_{q \in \mathcal{P}} \mathbf{KL}\Big[q(\boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z})||p(\boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z}|\boldsymbol{X})\Big].$$

The advantage of the variational formulation of Bayesian inference lies in the convenience of posing restrictions on the post-data distribution with a regularization term. For supervised topic models (Blei and McAuliffe, 2010; Zhu et al., 2012), such a regularization term could be a loss function of a prediction model $\boldsymbol{w}$ on the data $\boldsymbol{X} = \{\boldsymbol{x}_d\}_{d=1}^D$ and response signals $\boldsymbol{Y} = \{y_d\}_{d=1}^D$. As a regularized Bayesian (RegBayes) model, MedLDA infers a distribution of the latent variables $\boldsymbol{Z}$ as well as classification weights $\boldsymbol{w}$ by solving the problem:

$$\min_{q \in \mathcal{P}} \mathcal{L}\Big(q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z})\Big) + 2c \sum_{d=1}^D \ell_\epsilon\Big(q(\boldsymbol{w}, \boldsymbol{z}_d); \boldsymbol{x}_d, y_d\Big),$$

where $\mathcal{L}(q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z})) = \mathbf{KL}[q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z})||p(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z}|\boldsymbol{X})]$. To specify the loss function, a linear discriminant function needs to be defined with respect to $\boldsymbol{w}$ and $\boldsymbol{z}_d$

$$f(\boldsymbol{w}, \boldsymbol{z}_d) = \boldsymbol{w}^\top \bar{\boldsymbol{z}}_d, \tag{14}$$

where $\bar{z}_{dk} = \frac{1}{n_d} \sum_i \mathbb{I}[z_{di} = k]$ is the average frequency of assigning the words in document d to topic k. Based on the discriminant function, both averaging classifiers with the hinge loss

$$\ell_\epsilon^{\text{ave}}(q(\boldsymbol{w}, \boldsymbol{z}_d); \boldsymbol{x}_d, y_d) = \big(\epsilon - y_d \mathbb{E}_{q(\boldsymbol{w}, \boldsymbol{z}_d)}[f(\boldsymbol{w}, \boldsymbol{z}_d)]\big)_+, \tag{15}$$

4. Without causing confusion, we slightly abused the notation K to denote the topic number (i.e., the latent dimension) in topic models.

and Gibbs classifiers with the expected hinge loss

$$\ell_\epsilon^{\text{gibbs}}(q(\boldsymbol{w}, \boldsymbol{z}_d); \boldsymbol{x}_d, y_d) = \mathbb{E}_{q(\boldsymbol{w}, \boldsymbol{z}_d)}\left[(\epsilon - y_d f(\boldsymbol{w}, \boldsymbol{z}_d))_+\right], \tag{16}$$

have been proposed, with extensive comparisons reported in Zhu et al. (2014b) using batch learning algorithms. As commented in (Chang and Blei, 2010), defining the classifier on $\bar{\boldsymbol{z}}_d$, instead of $\boldsymbol{\theta}_d$, enforces words and labels to share the same topics. Besides, it retains the conjugacy structure between the Dirichlet prior of $\boldsymbol{\theta}$ and the multinomial likelihood of generating $\boldsymbol{z}$; and it will allow us to integrate out $\boldsymbol{\theta}$ for efficient collapsed inference.

5.2 Online MedLDA

We first apply online BayesPA to MedLDA with averaging classifiers, which will be referred to as paMedLDA$^{\text{ave}}$ for convenience. During inference, we integrate out the local variables $\boldsymbol{\Theta}_t$ using the conjugacy between a Dirichlet prior and a multinomial likelihood (Griffiths and Steyvers, 2004; Teh et al., 2006b), which potentially improves the inference accuracy. Then we have the global variables $\boldsymbol{\mathcal{M}} = \boldsymbol{\Phi}$ and local variables $\boldsymbol{H}_t = \boldsymbol{Z}_t$. The latent BayesPA rule (13) becomes:

$$\begin{aligned} \min_{q, \xi_d} \quad & \mathbf{KL}\Big[q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t)||q_t(\boldsymbol{w}, \boldsymbol{\Phi})p_0(\boldsymbol{Z}_t)p(\boldsymbol{X}_t|\boldsymbol{\Phi}, \boldsymbol{Z}_t)\Big] + 2c \sum_{d \in B_t} \xi_d, \\ \text{s.t.:} \quad & y_d \mathbb{E}_{q(\boldsymbol{w}, \boldsymbol{z}_d)}[\boldsymbol{w}^\top \bar{\boldsymbol{z}}_d] \geq \epsilon - \xi_d, \quad \xi_d \geq 0, \quad \forall d \in B_t, \\ & q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t) \in \mathcal{P}. \end{aligned} \tag{17}$$

Since directly solving the above problem is intractable, we would impose a mild mean-field assumption $q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t) = q(\boldsymbol{w})q(\boldsymbol{\Phi})q(\boldsymbol{Z}_t)$. Now, problem (17) can be solved using an iterative procedure that alternately updates each factor distribution (Jordan et al., 1998), as detailed below:

1. **Update global $q(\boldsymbol{\Phi})$:** By fixing the distributions $q(\boldsymbol{Z}_t)$ and $q(\boldsymbol{w})$, we can ignore irrelevant terms and solve

$$\min_{q(\boldsymbol{\Phi})} \mathbf{KL}\Big[q(\boldsymbol{\Phi})q(\boldsymbol{Z}_t)||q_t(\boldsymbol{\Phi})p_0(\boldsymbol{Z}_t)p(\boldsymbol{X}_t|\boldsymbol{\Phi}, \boldsymbol{Z}_t)\Big].$$

The optimal solution has the following closed form:

$$q^*(\boldsymbol{\Phi}_k) \propto q_t(\boldsymbol{\Phi}_k) \exp\Big(\mathbb{E}_{q(\boldsymbol{Z}_t)}\Big[\log p_0(\boldsymbol{Z}_t)p(\boldsymbol{X}_t|\boldsymbol{Z}_t, \boldsymbol{\Phi})\Big]\Big), \quad k = 1, 2, ..., K. \tag{18}$$

If initially the prior is $q_0(\boldsymbol{\Phi}_k) = \text{Dir}(\Delta_{k1}^0, ..., \Delta_{kW}^0)$, then by induction the inferred distributions in each round are also in the family of Dirichlet distributions, namely, $q_t(\boldsymbol{\Phi}_k) = \text{Dir}(\Delta_{k1}^t, ..., \Delta_{kW}^t)$. Using equation (18), we can derive

$$q^*(\boldsymbol{\Phi}_k) = \text{Dir}(\Delta_{k1}^*, ..., \Delta_{kW}^*), \tag{19}$$

where $\Delta_{kw}^* = \Delta_{kw}^t + \sum_{d \in B_t} \sum_{i=1}^{n_d} \gamma_{di}^k \cdot \mathbb{I}[x_{di} = w]$ for all words w ($1 \leq w \leq W$) in the vocuabulary and $\gamma_{di}^k = \mathbb{E}_{q(\boldsymbol{z}_d)}\mathbb{I}[z_{di} = k]$ is the probability of assigning each word x_{di} to topic k.

2. **Update global weight $q(\boldsymbol{w})$:** Keeping all the other distributions fixed, $q(\boldsymbol{w})$ can be solved as

$$\begin{aligned} \min_{q(\boldsymbol{w})} \quad & \mathbf{KL}\Big[q(\boldsymbol{w})||q_t(\boldsymbol{w})\Big] + 2c\sum_{d\in B_t}\xi_d, \\ \text{s.t.:} \quad & y_d\mathbb{E}_{q(\boldsymbol{w})}[\boldsymbol{w}]^\top\widehat{\boldsymbol{z}}_d \geq \epsilon - \xi_d, \quad \xi_d \geq 0, \quad \forall d \in B_t, \end{aligned}$$

where $\widehat{\boldsymbol{z}}_d = \mathbb{E}_{q(\boldsymbol{z}_d)}[\bar{\boldsymbol{z}}_d]$ is the expectation of topic assignments under the fixed distribution $q(\boldsymbol{Z})$. Similar to Proposition 2 in MedLDA (Zhu et al., 2012), the optimal solution is attained by solving the Lagrangian form with respect to $q(\boldsymbol{w})$, which gives

$$q^*(\boldsymbol{w}) = \frac{1}{Z(\tau_d^*)}q_t(\boldsymbol{w})\exp\left(\sum_{d\in B_t}\tau_d^* y_d \boldsymbol{w}^\top\widehat{\boldsymbol{z}}_d\right), \tag{20}$$

where the Lagrange multipliers $\tau_d^*(d \in B_t)$ are obtained by solving the dual problem

$$\max_{0\leq\tau_d\leq 2c} \quad \epsilon\sum_{d\in B_t}\tau_d - \log Z(\tau_d).$$

For the common spherical Gaussian prior $q_0(\boldsymbol{w}) = \mathcal{N}(0, \sigma^2 I)$, by induction the distribution $q_t(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{\mu}_t, \sigma^2 I)$ at each round. So equation (20) gives $q^*(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{\mu}^*, \sigma^2 I)$, where

$$\boldsymbol{\mu}^* = \boldsymbol{\mu}_t + \sigma^2\sum_{d\in B_t}\tau_d^* y_d\widehat{\boldsymbol{z}}_d. \tag{21}$$

Furthermore, the dual problem becomes,

$$\max_{0\leq\tau_d\leq 2c} \quad \epsilon\sum_{d\in B_t}\tau_d - \sum_{d_1,d_2\in B_t}\frac{1}{2}\sigma^2\tau_{d_1}\tau_{d_2}\widehat{\boldsymbol{z}}_{d_1}^\top\widehat{\boldsymbol{z}}_{d_2} - \boldsymbol{\mu}_t^\top\sum_{d\in B_t}y_d\tau_d\widehat{\boldsymbol{z}}_d, \tag{22}$$

which is identical to the Lagrangian dual of the classical PA problem with mini-batch B_t (expressed in the equivalent constrained form by introducing slack variables)

$$\min_{\mu}\frac{||\boldsymbol{\mu} - \boldsymbol{\mu}_t||^2}{2\sigma^2} + 2c\sum_{d\in B_t}\Big(\epsilon - y_d\boldsymbol{\mu}^\top\widehat{\boldsymbol{z}}_d\Big)_+. \tag{23}$$

This equivalence suggests that we could rely on contemporary PA techniques to solve for $\boldsymbol{\mu}^*$. In particular, for instances coming one at a time (i.e., $B_t = \{t\},\ \forall t$), we have the closed-form solution

$$\tau_t^* = \min\left\{2c, \frac{(\epsilon - y_t\mu_t^\top\widehat{\boldsymbol{z}}_t)_+}{||\widehat{\boldsymbol{z}}_t||^2}\right\},$$

whose computation requires $O(K)$ time; for mini-batches, we could adapt methods solving linear SVM to either the dual (22) or primal (23) problem, which by state-of-the-arts require complexity $O(\text{poly}(\epsilon^{-1})K)$ per training instance in order to obtain ϵ-accurate solutions. Here we choose a gradient-based method similar to Shalev-Shwartz et al. (2011). Specifically, we first set $\boldsymbol{\mu}^1 = \boldsymbol{\mu}_t$, and then take the gradient steps $i = 2, 3, ...$ until $\boldsymbol{\mu}^i$ converges to $\boldsymbol{\mu}^*$. Let

A_t^i be the set of instances in B_t that suffer non-zero loss at step i, then we use the gradients to iteratively update

$$\boldsymbol{\mu}^{i+1} \leftarrow \boldsymbol{\mu}^i - \rho_i\nabla_i, \tag{24}$$

where annealing rate $\rho_i = \sigma^2 i^{-1}$ and

$$\nabla_i = \frac{\boldsymbol{\mu}^i - \boldsymbol{\mu}_t}{\sigma^2} - 2c\sum_{d\in A_t^i}y_d\widehat{\boldsymbol{z}}_d.$$

Correspondingly, we can derive the gradient-based update rule for the dual parameters. Imagine that we implicitly maintain the relationship $\boldsymbol{\mu} = \boldsymbol{\mu}_t + \sigma^2\sum_{d\in B_t}\tau_d y_d\widehat{\boldsymbol{z}}_d$. Then the following update rule for $\tau_d\ (d \in B_t)$ naturally implies the update rule (24) for $\boldsymbol{\mu}$:

$$\tau_d^i \leftarrow \begin{cases}(1-\frac{1}{i})\tau_d^i + \frac{2c}{i} & \text{for } d \in A_t^i \\ (1-\frac{1}{i})\tau_d^i & \text{for } d \notin A_t^i.\end{cases}$$

Therefore, the gradient steps adaptively adjust the contribution of each latent $\widehat{\boldsymbol{z}}_d$ to μ based on the loss it incurs. Furthermore, the annealing makes sure that $0 \leq \tau_d^i \leq 2c$ for all i. Since the problem (22) is concave, it can be guaranteed that τ_d^i converges to τ_d^*. This correspondence would be used in updating $q(\boldsymbol{Z}_t)$.

3. **Update local $q(\boldsymbol{Z}_t)$:** Fixing all the other distributions, we aim to solve

$$\begin{aligned} \min_{q(\boldsymbol{Z}_t)} \quad & \mathbf{KL}\Big[q(\boldsymbol{Z}_t)||p_0(\boldsymbol{Z}_t)p(\boldsymbol{X}_t|\boldsymbol{Z}_t, \boldsymbol{\Phi})\Big] + 2c\sum_{d\in B_t}\xi_d, \\ \text{s.t.:} \quad & y_d\boldsymbol{\mu}^{*\top}\mathbb{E}_{q(\boldsymbol{z}_d)}[\bar{\boldsymbol{z}}_d] \geq \epsilon - \xi_d, \quad \xi_d \geq 0, \quad \forall d \in B_t, \end{aligned}$$

where $\boldsymbol{\mu}^* = \mathbb{E}_{q(\boldsymbol{w})}[\boldsymbol{w}]$ is the expectation of $\boldsymbol{w}$ under the fixed distribution $q(\boldsymbol{w})$. Unlike the weight $\boldsymbol{w}$, the expectation over $\boldsymbol{Z}_t$ during optimization is intractable due to the combinatorial space of values. Instead, we adopt the same approximation strategy as MedLDA (Zhu et al., 2012): fix ξ, τ_d at the previous global step, and use the approximate solution

$$q^*(\boldsymbol{Z}_t) = p_0(\boldsymbol{Z}_t)p(\boldsymbol{X}_t|\boldsymbol{Z}_t, \boldsymbol{\Phi})\exp\left(\sum_{d\in B_t}\tau_d^* y_d\boldsymbol{\mu}^{*\top}\bar{\boldsymbol{z}}_d\right).$$

Then the expectation of $\bar{\boldsymbol{z}}_d$, as needed in the global updates, could be approximated by samples from the distribution $q^*(\boldsymbol{Z}_t)$. Specifically, we use Gibbs sampling with the conditional distribution

$$q(z_{di} = k \mid \boldsymbol{Z}_t^{\neg di}) \propto \left(\alpha + C_{dk}^{\neg di}\right)\exp\left(\Lambda_{k,x_{di}} + \sum_{d\in B_t}n_d^{-1}y_d\tau_d^*\mu_k^*\right). \tag{25}$$

where $\Lambda_{z_{di},x_{di}} = \mathbb{E}_{q(\boldsymbol{\Phi})}[\log(\Phi_{z_{di},x_{di}})] = \Psi(\Delta^*_{z_{di},x_{di}}) - \Psi(\sum_w\Delta^*_{z_{di},w})$ (note that $\Psi(\cdot)$ is the digamma function) and $\boldsymbol{C}_d^{\neg di}$ is a vector with the k-th entry being the number of words in document d (except the i-th word) that are assigned to topic k.

Then we draw $\mathcal{J}$ samples $\{\boldsymbol{Z}_t^{(j)}\}_{j=1}^{\mathcal{J}}$ using Eq. (25), discard the first β ($0 \leq \beta < \mathcal{J}$) burn-in samples, and approximate $\widehat{z}_{dk}$ with the empirical sum $(J-\beta)^{-1}\sum_{j=\beta+1}^{J}\sum_{d,i}\mathbb{I}[z_{di}^{(j)} = k]$.

Algorithm 1 Online MedLDA

Let $q_0(\boldsymbol{w}) = \mathcal{N}(0; \sigma^2 I), q_0(\boldsymbol{\phi}_k) = \text{Dir}(\gamma),\ \forall\, k$.
for $t = 0 \to \infty$ **do**
  Set $q(\boldsymbol{\Phi}, \boldsymbol{w}) = q_t(\boldsymbol{\Phi}) q_t(\boldsymbol{w})$. Initialize $\boldsymbol{Z}_t$.
  for $i = 1 \to \mathcal{I}$ **do**
    Draw samples $\{\boldsymbol{Z}_t^{(j)}\}_{j=1}^{\mathcal{J}}$ from Eq. (25).
    Discard the first β burn-in samples ($\beta < \mathcal{J}$).
    Use the rest $\mathcal{J} - \beta$ samples to update $q(\boldsymbol{\Phi}, \boldsymbol{w})$ following Eq.s (19, 20).
  end for
  Set $q_{t+1}(\boldsymbol{\Phi}, \boldsymbol{w}) = q(\boldsymbol{\Phi}, \boldsymbol{w})$.
end for

At each round t of BayesPA optimization during training, we run the global and local updates alternately until convergence, and assign $q_t(\boldsymbol{\Phi}, \boldsymbol{w}) = q^*(\boldsymbol{\Phi}) q^*(\boldsymbol{w})$, as outlined in Algorithm 1. To make predictions on testing data, we use the mean $\boldsymbol{\mu}$ as the classification weight and apply the prediction rule. The inference of $\bar{\boldsymbol{z}}$ for testing documents is similar to Zhu et al. (2014b). First, we draw a single sample of $\boldsymbol{\Phi}$, and for each test document d, we infer the MAP of $\boldsymbol{\theta}_d$. Then, we directly run the sampling of $\boldsymbol{z}_d$ until the burn-in stage is completed, and use the average of several samples to compute $\widehat{\boldsymbol{z}}_d$. Then the prediction rule is applied on $\boldsymbol{\mu}$ and $\widehat{\boldsymbol{z}}_d$.

5.3 Online Gibbs MedLDA

In this section, we apply the theory of BayesPA to Gibbs MedLDA. As shown in Zhu et al. (2014b), using Gibbs classifiers admits efficient inference algorithms by exploring *data augmentation* (DA) techniques (Tanner and Wong, 1987; Polson and Scott, 2011). Based on this insight, we will develop our efficient online inference algorithms for Gibbs MedLDA. We denote the model by paMedLDA$^{\text{gibbs}}$. Specifically, let $\zeta_d = \epsilon - y_d f(\boldsymbol{w}, \boldsymbol{z}_d)$ and $\psi(y_d|\boldsymbol{z}_d, \boldsymbol{w}) = e^{-2c(\zeta_d)_+}$. By Lemma 4, the optimal solution to problem (13) is

$$q_{t+1}(\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t) = \frac{q_t(\boldsymbol{w}, \boldsymbol{\mathcal{M}}) p_0(\boldsymbol{H}_t) p(\boldsymbol{X}_t|\boldsymbol{H}_t, \boldsymbol{\mathcal{M}}) \psi(\boldsymbol{Y}_t|\boldsymbol{H}_t, \boldsymbol{w})}{\Gamma(\boldsymbol{X}_t, \boldsymbol{Y}_t)}\ ,$$

where $\psi(\boldsymbol{Y}_t|\boldsymbol{H}_t, \boldsymbol{w}) = \prod_{d \in B_t} \psi(y_d|\boldsymbol{h}_d, \boldsymbol{w})$ and $\Gamma(\boldsymbol{X}_t, \boldsymbol{Y}_t)$ is a normalization constant. The basic idea of DA is to construct conjugacy between prior and data during inference by introducing augmented variables. Specifically, we would use the following equality (Zhu et al., 2014b):

$$\psi(y_d|\boldsymbol{z}_d, \boldsymbol{w}) = \int_0^\infty \psi(y_d, \lambda_d|\boldsymbol{z}_d, \boldsymbol{w}) d\lambda_d, \tag{26}$$

where $\psi(y_d, \lambda_d|\boldsymbol{z}_d, \boldsymbol{w}) = (2\pi\lambda_d)^{-1/2} \exp\left(-\frac{(\lambda_d + c\zeta_d)^2}{2\lambda_d}\right)$. Equality (26) essentially implies that the collapsed posterior $q_{t+1}(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t)$ is a marginal distribution of

$$q_{t+1}(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t, \boldsymbol{\lambda}_t) = \frac{p_0(\boldsymbol{Z}_t) q_t(\boldsymbol{w}, \boldsymbol{\Phi}) p(\boldsymbol{X}_t|\boldsymbol{Z}_t, \boldsymbol{\Phi}) \psi(\boldsymbol{Y}_t, \boldsymbol{\lambda}_t|\boldsymbol{Z}_t, \boldsymbol{w})}{\Gamma(\boldsymbol{X}_t, \boldsymbol{Y}_t)}\ ,$$

where $\psi(\boldsymbol{Y}_t, \boldsymbol{\lambda}_t|\boldsymbol{Z}_t, \boldsymbol{w}) = \prod_{d \in B_t} \psi(y_d, \lambda_d|\boldsymbol{z}_d, \boldsymbol{w})$ and $\boldsymbol{\lambda}_t = \{\lambda_d\}_{d \in B_t}$ are augmented variables locally associated with individual documents. In fact, the augmented distribution $q_{t+1}(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t, \boldsymbol{\lambda}_t)$ is the

solution to the problem:

$$\min_{q \in \mathcal{P}} \mathcal{L}\Big(q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t, \boldsymbol{\lambda}_t)\Big) - \mathbb{E}_q\Big[\log \psi(\boldsymbol{Y}_t, \boldsymbol{\lambda}_t|\boldsymbol{Z}_t, \boldsymbol{w})\Big], \tag{27}$$

where $\mathcal{L}(q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t, \boldsymbol{\lambda}_t)) = \mathbf{KL}[q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t, \boldsymbol{\lambda}_t) \| q_t(\boldsymbol{w}, \boldsymbol{\Phi}) p_0(\boldsymbol{Z}_t)] - \mathbb{E}_q[\log p(\boldsymbol{X}_t|\boldsymbol{Z}_t, \boldsymbol{\Phi})]$. In fact, this objective is an upper bound of that in the original problem (13) (See Appendix A for details).

Again, with the mild mean-field assumption that $q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t, \boldsymbol{\lambda}_t) = q(\boldsymbol{w}, \boldsymbol{\Phi}) q(\boldsymbol{Z}_t, \boldsymbol{\lambda}_t)$, we can solve problem (27) via an iterative procedure that alternately updates each factor distribution (Jordan et al., 1998), as detailed below.

1. **Global Update:** By fixing the distribution of local variables, $q(\boldsymbol{Z}_t, \boldsymbol{\lambda}_t)$, and ignoring irrelevant terms, the optimal distribution of $\boldsymbol{w}$ and $\boldsymbol{\Phi}$ can be shown to have the induced factorization form, $q(\boldsymbol{w}, \boldsymbol{\Phi}) = q(\boldsymbol{w}) q(\boldsymbol{\Phi})$. For $q(\boldsymbol{\Phi})$, the update rule is exactly (19). For $q(\boldsymbol{w})$, we have the update rule

$$q_{t+1}(\boldsymbol{w}) \propto q_t(\boldsymbol{w}) \exp\left(\mathbb{E}_{q(\boldsymbol{Z}_t, \boldsymbol{\lambda}_t)}\Big[\log p_0(\boldsymbol{Z}_t) \psi(\boldsymbol{Y}_t, \boldsymbol{\lambda}_t|\boldsymbol{Z}_t, \boldsymbol{w})\Big]\right).$$

   If the initial prior is normal $q_0(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}^0, \boldsymbol{\Sigma}^0)$, by induction we can show that the inferred distribution in each round is also a normal distribution, namely, $q_t(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}^t, \boldsymbol{\Sigma}^t)$. Indeed, the optimal solution of $q(\boldsymbol{w})$ to problem (27) is

$$q^*(\boldsymbol{w}) = \mathcal{N}(\boldsymbol{w}; \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*), \tag{28}$$

   where the posterior paramters are computed as

$$\boldsymbol{\Sigma}^* = \left((\boldsymbol{\Sigma}^t)^{-1} + c^2 \sum_{d \in B_t} \mathbb{E}_{q(\boldsymbol{z}_d, \lambda_d)}\left[\lambda_d^{-1} \bar{\boldsymbol{z}}_d \bar{\boldsymbol{z}}_d^\top\right]\right)^{-1},$$

$$\boldsymbol{\mu}^* = \boldsymbol{\Sigma}^* \left((\boldsymbol{\Sigma}^t)^{-1} \boldsymbol{\mu}^t + c \sum_{d \in B_t} \mathbb{E}_{q(\boldsymbol{z}_d, \lambda_d)}\left[y_d \left(1 + c\epsilon\lambda_d^{-1}\right) \bar{\boldsymbol{z}}_d\right]\right).$$

   For the sequential update rule, we simply set $\boldsymbol{\mu}^{t+1} = \boldsymbol{\mu}^*$ and $\boldsymbol{\Sigma}^{t+1} = \boldsymbol{\Sigma}^*$.

2. **Local Update:** Given the distribution of global variables, $q(\boldsymbol{\Phi}, \boldsymbol{w})$, the mean-field update equation for $(\boldsymbol{Z}_t, \boldsymbol{\lambda}_t)$ is

$$q(\boldsymbol{Z}_t, \boldsymbol{\lambda}_t) \propto p_0(\boldsymbol{Z}_t) \prod_{d \in B_t} \frac{1}{\sqrt{2\pi\lambda_d}} \exp\left(\sum_{i \in [n_d]} \Lambda_{z_{di}, x_{di}} - \mathbb{E}_{q(\boldsymbol{\Phi}, \boldsymbol{w})}\left[\frac{(\lambda_d + c\zeta_d)^2}{2\lambda_d}\right]\right),$$

   where Λ admits the same definition as in (25). But it is impossible to evaluate the expectation in the global update using $q(\boldsymbol{Z}_t, \boldsymbol{\lambda}_t)$ because of the huge number of configurations for $(\boldsymbol{Z}_t, \boldsymbol{\lambda}_t)$. As a result, we turn to Gibbs sampling and estimate the required expectations using multiple empirical samples. This hybrid strategy has shown promising performance for LDA (Mimno et al., 2012). Specifically, the conditional distributions used in the Gibbs sampling are as follows:

Algorithm 2 Online Gibbs MedLDA

Let $q_0(\boldsymbol{w}) = \mathcal{N}(0; \sigma^2 I), q_0(\boldsymbol{\phi}_k) = \text{Dir}(\gamma),\ \forall\, k$.
for $t = 0 \to \infty$ **do**
  Set $q(\boldsymbol{\Phi}, \boldsymbol{w}) = q_t(\boldsymbol{\Phi}) q_t(\boldsymbol{w})$. Initialize $\boldsymbol{Z}_t$.
  for $i = 1 \to \mathcal{I}$ **do**
    Draw samples $\{\boldsymbol{Z}_t^{(j)}, \boldsymbol{\lambda}_t^{(j)}\}_{j=1}^{\mathcal{J}}$ from Eq.s (29, 30).
    Discard the first β burn-in samples ($\beta < \mathcal{J}$).
    Use the rest $\mathcal{J} - \beta$ samples to update $q(\boldsymbol{\Phi}, \boldsymbol{w})$ following Eq.s (19, 28).
  end for
  Set $q_{t+1}(\boldsymbol{\Phi}, \boldsymbol{w}) = q(\boldsymbol{\Phi}, \boldsymbol{w})$.
end for

- **For $\boldsymbol{Z}_t$:** By canceling out common factors, the conditional distribution of one variable z_{di} given $\boldsymbol{Z}_t^{\neg di}$ and $\boldsymbol{\lambda}_t$ is

$$q(z_{di}{=}k|\boldsymbol{Z}_t^{\neg di}, \boldsymbol{\lambda}_t) \propto (\alpha + C_{dk}^{\neg di}) \exp\Big(\frac{c y_d (c\epsilon + \lambda_d) \mu_k^*}{n_d \lambda_d} + \Lambda_{k, x_{di}} - \frac{c^2(\mu_k^{*2} + \Sigma_{kk}^* + 2(\mu_k^* \boldsymbol{\mu}^* + \boldsymbol{\Sigma}_{\cdot,k}^*)^\top \boldsymbol{C}_d^{\neg di})}{2 n_d^2 \lambda_d} \Big), \tag{29}$$

  where $\boldsymbol{\Sigma}_{\cdot,k}^*$ is the k-th column of $\boldsymbol{\Sigma}^*$.

- **For $\boldsymbol{\lambda}_t$:** Let $\bar{\zeta}_d = \epsilon - y_d \bar{\boldsymbol{z}}_d^\top \boldsymbol{\mu}^*$. The conditional distribution of each variable λ_d given $\boldsymbol{Z}_t$ is

$$\begin{aligned} q(\lambda_d|\boldsymbol{Z}_t) &\propto \frac{1}{\sqrt{2\pi\lambda_d}} \exp\left(-\frac{c^2 \bar{\boldsymbol{z}}_d^\top \boldsymbol{\Sigma}^* \bar{\boldsymbol{z}}_d + (\lambda_d + c\bar{\zeta}_d)^2}{2\lambda_d} \right) \\ &= \mathcal{GIG}\left(\lambda_d; \frac{1}{2}, 1, c^2 \left(\bar{\zeta}_d^2 + \bar{\boldsymbol{z}}_d^\top \boldsymbol{\Sigma}^* \bar{\boldsymbol{z}}_d \right) \right), \end{aligned} \tag{30}$$

  a generalized inverse gaussian distribution (Devroye, 1986). Therefore, λ_d^{-1} follows an inverse gaussian distribution, that is,

$$q(\lambda_d^{-1}|\boldsymbol{Z}_t) = \mathcal{IG}\left(\lambda_d^{-1}; \frac{1}{c\sqrt{\bar{\zeta}_d^2 + \bar{\boldsymbol{z}}_d^\top \boldsymbol{\Sigma}^* \bar{\boldsymbol{z}}_d}}, 1 \right),$$

  from which we can draw a sample in constant time (Michael et al., 1976).

For training, we run the global and local updates alternately until convergence at each round of PA optimization, as outlined in Alg. 2. To make predictions on testing data, we then draw one sample of $\hat{\boldsymbol{w}}$ as the classification weight and apply the prediction rule. The inference of $\bar{\boldsymbol{z}}$ for testing documents is the same as online MedLDA.

6. Extensions

In the above topic models, we assume that the number of topics (i.e., K) is pre-specified, which is unfortunately unknown in practice. An extra procedure is often applied to select a good K value.

We now present extensions of online MedLDA to automatically determine the unknown K values by leveraging nonparametric Bayesian techniques. We also present an extension of these models for multi-task learning.

6.1 Online Nonparametric MedLDA

We first present online nonparametric MedLDA for resolving the unknown number of topics, based on the theory of hierarchical Dirichlet process (HDP) (Teh et al., 2006a).

6.1.1 BATCH MEDHDP

A two-level HDP provides an extension to LDA that allows for a nonparametric inference of the unknown topic numbers. The generative process of HDP can be summarized using a stick-breaking construction (Wang and Blei, 2012b), where the stick lengths $\boldsymbol{\pi} = \{\pi_k\}_{k=1}^\infty$ are generated as:

$$\pi_k = \bar{\pi}_k \prod_{i<k} (1 - \bar{\pi}_i),\ \bar{\pi}_k \sim \text{Beta}(1, \eta),\ \text{for } k = 1, ..., \infty,$$

and the topic mixing proportions are generated as $\boldsymbol{\theta}_d \sim \text{Dir}(\alpha\boldsymbol{\pi}),$ for $d = 1, ..., D$. Each topic $\boldsymbol{\phi}_k$ is a sample from a Dirichlet base distribution, i.e., $\boldsymbol{\phi}_k \sim \text{Dir}(\boldsymbol{\gamma})$. After we get the topic mixing proportions $\boldsymbol{\theta}_d$, the generation of words is the same as in the standard LDA.

To extend the HDP topic model for predictive tasks, we introduce a classifier $\boldsymbol{w}$ that is drawn from a Gaussian process, $\mathcal{GP}(0, \boldsymbol{\Sigma})$, where the covariance function is $\Sigma(\boldsymbol{w}, \boldsymbol{w}') = \sigma^2 \mathbb{I}[\boldsymbol{w} = \boldsymbol{w}']$. We still define the linear discriminant function in the same form as Eq. (14). Since the number of words in a document is finite, the average topic assignment vector $\bar{\boldsymbol{z}}_d$ has only a finite number of non-zero elements, and the dot product in Eq. (14) is in fact finite. Therefore, given the latent topic assignments, the conditoinal posterior of $\boldsymbol{w}$ is in fact a multivariate Gaussian distribution.

Let $\bar{\boldsymbol{\pi}} = \{\bar{\pi}_k\}_{k=1}^\infty$. We define maximum entropy discrimination HDP (MedHDP) topic model as solving the following RegBayes problem to infer the joint post-data posterior $q(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z})$:[5]

$$\min_{q \in \mathcal{P}} \mathcal{L}\Big(q(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z}) \Big) + 2c \sum_{d=1}^D \ell_\epsilon\Big(q(\boldsymbol{w}, \boldsymbol{z}_d); \boldsymbol{x}_d, \boldsymbol{y}_d \Big), \tag{31}$$

where $\mathcal{L}(q(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z})) = \mathbf{KL}[q(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z}) || p(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \boldsymbol{Z}|\boldsymbol{X})]$ is the objective corresponding to the standard Bayesian inference under the variational formulation of Bayes' rule. The loss function could be either (15) or (16). We call the resulting model with the averaging classifier MedHDP$^{\text{ave}}$ and that with the Gibbs classifier MedHDP$^{\text{gibbs}}$.

Since MedHDP is a new model, we would briefly discuss the corresponding inference problem. For the inference of MedHDP$^{\text{gibbs}}$, we can use Gibbs sampling based on Chinese Restaurant Franchise (Teh et al., 2006a; Wang and Blei, 2012a) with modifications similar to the techniques introduced in Zhu et al. (2014b). For MedHDP$^{\text{ave}}$, the current state-of-the-art for inferring max-margin Bayesian models with averaging loss resorts to mean-field assumptions and variational inference. Notice that classical mean-field derivation would fail due to the potentially unbounded space of variables. However, it is possible to incorporate Gibbs sampling into mean-field update equations to explore the unbounded space (Welling et al., 2008; Wang and Blei, 2012b) and therefore bypass the difficulty. In this paper, we would not focus on developing inference algorithms for MedHDP,

5. Given $\bar{\boldsymbol{\pi}}$, $\boldsymbol{\pi}$ can be computed via the stick breaking process.

but instead attain batch MedHDP algorithms from the corresponding BayesPA methods, as will be clear at the end of each subsection below.

6.1.2 ONLINE MEDHDP

To apply the ideas of BayesPA to develop online MedHDP algorithms, we have the global variables $\boldsymbol{\mathcal{M}} = (\bar{\boldsymbol{\pi}}, \boldsymbol{\Phi})$, and the local variables $\boldsymbol{H}_t = (\boldsymbol{\Theta}_t, \boldsymbol{Z}_t)$. As in online MedLDA, we marginalize out $\boldsymbol{\Theta}_t$ by conjugacy. Furthermore, to simplify the sampling scheme, we introduce another set of auxiliary latent variables $\boldsymbol{S}_t = \{\boldsymbol{s}_d\}_{d\in B_t}$, where $\boldsymbol{s}_d = \{s_{dk}\}_{k=1}^{\infty}$ and each element s_{dk} represents the number of occupied tables serving dish k in a Chinese restaurant process (CRP) (Teh et al., 2006a; Wang and Blei, 2012b). By definition, we have $p(\boldsymbol{Z}_t, \boldsymbol{S}_t|\bar{\boldsymbol{\pi}}) = \prod_{d\in B_t} p(\boldsymbol{s}_d, \boldsymbol{z}_d|\bar{\boldsymbol{\pi}})$ and

$$p(\boldsymbol{s}_d, \boldsymbol{z}_d|\bar{\boldsymbol{\pi}}) \propto \prod_{k=1}^{\infty} S(n_d\tilde{z}_{dk}, s_{dk})(\alpha\pi_k)^{s_{dk}}, \tag{32}$$

where $S(a, b)$ are unsigned Stirling numbers of the first kind (Antoniak, 1974). It is not hard to verify that $p(\boldsymbol{z}_d|\bar{\boldsymbol{\pi}}) = \sum_{\boldsymbol{s}_d} p(\boldsymbol{s}_d, \boldsymbol{z}_d|\bar{\boldsymbol{\pi}})$. After this "collapse-and-augment" procedure, we now have the local variables $\boldsymbol{H}_t = (\boldsymbol{Z}_t, \boldsymbol{S}_t)$. The global variables remain intact. The new BayesPA problem is now:

$$\min_{q\in\mathcal{F}_t} \mathcal{L}\Big(q(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi}, \boldsymbol{H}_t)\Big) + 2c\sum_{d\in B_t} \ell_\epsilon\Big(\boldsymbol{w}; \boldsymbol{x}_t, y_t\Big), \tag{33}$$

where $\mathcal{L}(q(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi}, \boldsymbol{H}_t)) = \mathbf{KL}[q(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi}, \boldsymbol{H}_t)||q_t(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi})p(\boldsymbol{Z}_t, \boldsymbol{S}_t|\bar{\boldsymbol{\pi}})p(\boldsymbol{X}_t|\boldsymbol{Z}_t, \boldsymbol{\Phi})]$. As in online MedLDA, we adopt the mild mean field assumption $q(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi}, \boldsymbol{H}_t) = q(\boldsymbol{w})q(\bar{\boldsymbol{\pi}})q(\boldsymbol{\Phi})q(\boldsymbol{H}_t)$ and solve problem (33) via an iterative procedure detailed below.

1. **Global Update:** By fixing the distribution of local variables, $q(\boldsymbol{H}_t)$, and ignoring the irrelevant terms, we have same mean-field update equations (19) for $\boldsymbol{\Phi}$ and (21) for $\boldsymbol{w}$ with the averaging loss. For global variable $\bar{\boldsymbol{\pi}}$, we have

$$q^*(\bar{\pi}_k) \propto q_t(\bar{\pi}_k) \prod_{d\in B_t} \exp\left(\mathbb{E}_{q(\boldsymbol{h}_d)}\Big[\log p(\boldsymbol{s}_d, \boldsymbol{z}_d|\bar{\boldsymbol{\pi}})\Big]\right). \tag{34}$$

   By induction, we can show that $q_t(\bar{\pi}_k) = \text{Beta}(u_k^t, v_k^t)$ is a Beta distribution at each step, and the update equation is

$$q^*(\bar{\pi}_k) = \text{Beta}(u_k^*, v_k^*), \tag{35}$$

   where $u_k^* = u_k^t + \sum_{d\in B_t} \mathbb{E}_{q(\boldsymbol{s}_d)}[s_{dk}]$ and $v_k^* = v_k^t + \sum_{d\in B_t} \mathbb{E}_{q(\boldsymbol{s}_d)}[\sum_{j>k} s_{dj}]$ for $k = \{1, 2, ...\}$ and $u_k^0 = 1, v_k^0 = \eta$.

   Since $\boldsymbol{Z}_t$ contains only a finite number of discrete variables, we only need to maintain and update the above global distributions for a finite number of topics.

2. **Local Update:** Fixing the global distribution $q(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi})$, we get the mean-field update equation for $(\boldsymbol{Z}_t, \boldsymbol{S}_t)$:

$$q^*(\boldsymbol{Z}_t, \boldsymbol{S}_t) \propto \tilde{q}(\boldsymbol{Z}_t, \boldsymbol{S}_t)\hat{q}(\boldsymbol{Z}_t) \tag{36}$$

where

$$\tilde{q}(\boldsymbol{Z}_t, \boldsymbol{S}_t) = \exp\left(\mathbb{E}_{q^*(\boldsymbol{\Phi})q^*(\bar{\boldsymbol{\pi}})}[\log p(\boldsymbol{X}|\boldsymbol{\Phi}, \boldsymbol{Z}_t) + \log p(\boldsymbol{Z}_t, \boldsymbol{S}_t|\bar{\boldsymbol{\pi}})]\right),$$
$$\hat{q}(\boldsymbol{Z}_t) = \exp\left(\sum_{d\in B_t} \tau_d y_d \mathbb{E}[\boldsymbol{w}]^\top \bar{\boldsymbol{z}}_d\right),$$

and $\tau_d (d \in B_t)$ are the dual variables computed in the global update. The most cumbersome point to tackle is the potentially unbounded sample space of $\boldsymbol{Z}_t$ and $\boldsymbol{S}_t$. We take the ideas from (Wang and Blei, 2012b) and adopt an approximation for $\tilde{q}(\boldsymbol{Z}_t, \boldsymbol{S}_t)$:

$$\tilde{q}(\boldsymbol{Z}_t, \boldsymbol{S}_t) \approx \mathbb{E}_{q^*(\boldsymbol{\Phi})q^*(\bar{\boldsymbol{\pi}})}\left[p(\boldsymbol{X}|\boldsymbol{\Phi}, \boldsymbol{Z}_t)p(\boldsymbol{Z}_t, \boldsymbol{S}_t|\bar{\boldsymbol{\pi}})\right]. \tag{37}$$

Computing the expectation regarding $\bar{\boldsymbol{\pi}}$ in (37) turns out to be difficult. However, imagine that the expectation operator is essentially collapsing $\bar{\boldsymbol{\pi}}$ out from the joint distribution

$$\tilde{q}(\bar{\boldsymbol{\pi}}, \boldsymbol{Z}_t, \boldsymbol{S}_t) \approx \mathbb{E}_{q^*(\boldsymbol{\Phi})}\left[q^*(\bar{\boldsymbol{\pi}})p(\boldsymbol{X}|\boldsymbol{\Phi}, \boldsymbol{Z}_t)p(\boldsymbol{Z}_t, \boldsymbol{S}_t|\bar{\boldsymbol{\pi}})\right]. \tag{38}$$

Now we propose to uncollapse $\bar{\boldsymbol{\pi}}$ and sample the local variables from

$$q^*(\bar{\boldsymbol{\pi}}, \boldsymbol{Z}_t, \boldsymbol{S}_t) \propto \tilde{q}(\bar{\boldsymbol{\pi}}, \boldsymbol{Z}_t, \boldsymbol{S}_t)\hat{q}(\boldsymbol{Z}_t). \tag{39}$$

Notice in the local updates, $\bar{\boldsymbol{\pi}}$ is only an auxilary variable. Putting all the pieces together, we have the following sampling scheme.

- **For $\boldsymbol{Z}_t$:** Let K be the current inferred number of topics. The conditional distribution of one variable z_{di} given all other local variables can be derived from (36) with $\boldsymbol{s}_d$ marginalized out for convenience.

$$q(z_{di} = k|\boldsymbol{Z}_t^{\neg di}, \bar{\boldsymbol{\pi}}) \propto \frac{(\alpha\pi_k + C_{dk}^{\neg di})(C_{kx_{di}}^{\neg di} + \Delta_{kx_{di}}^t)}{\sum_w (C_{kw}^{\neg di} + \Delta_{kw}^t)} \exp\left(\sum_{d\in B_t} n_d^{-1} y_d \tau_d \mu_k^*\right). \tag{40}$$

  Besides, for $k > K$ and symmetric Dirichlet prior $\boldsymbol{\gamma}$, (40) converge to a single rule $q(z_{di} = k|\boldsymbol{Z}_t^{\neg di}, \bar{\boldsymbol{\pi}}) \propto \alpha\pi_k/W$, and therefore the total probability of assigning a new topic is

$$q(z_{di} > K|\boldsymbol{Z}_t^{\neg di}, \bar{\boldsymbol{\pi}}) \propto \alpha\left(1 - \sum_{k=1}^{K} \pi_k\right)/W.$$

- **For $\boldsymbol{S}_t$:** The conditional distribution of s_{dk} given $(\boldsymbol{Z}_t, \bar{\boldsymbol{\pi}}, \boldsymbol{\lambda}_t)$ can be derived from the joint distribution (32):

$$q(s_{dk}|\boldsymbol{Z}_t, \bar{\boldsymbol{\pi}}) \propto S(n_d\tilde{z}_{dk}, s_{dk})(\alpha\pi_k)^{s_{dk}}. \tag{41}$$

- **For $\bar{\boldsymbol{\pi}}$:** It can be derived from (36) that given $(\boldsymbol{Z}_t, \boldsymbol{S}_t)$, each $\bar{\pi}_k$ follows the beta distribution,

$$\bar{\pi}_k \sim \text{Beta}(a_k, b_k), \tag{42}$$

  where $a_k = u_k^* + \sum_{d\in B_t} s_{dk}$ and $b_k = v_k^* + \sum_{d\in B_t}\sum_{j>k} s_{dj}$.

Similar to online MedLDA, we iterate the above steps till convergence for training. For testing, the learned model is essentially a finite MedLDA, and we use the same scheme as that of online MedLDA.

Notice that if we run online MedHDP for only one round ($T = 1$) and use the entire dataset as mini-batch ($|B| = D$), iterating the above steps till converge in fact solves the batch MedHDP problem Eq. (31). We call this batch version MedHDP$^{\text{ave}}$, and will use it as a baseline algorithm.

6.1.3 ONLINE GIBBS MEDHDP

For Gibbs MedHDP, the only difference is the loss functional ℓ_ϵ, which is reflected in the sampling of local variables. As in online Gibbs MedLDA, we can facilitate more efficient inference by adopting the same data augmentation technique with the augmented variables $\boldsymbol{\lambda}_t$. Then the local variables are $(\boldsymbol{Z}_t, \boldsymbol{S}_t, \boldsymbol{\lambda}_t)$ and the global variables are unchanged. We then use the mean field assumption $q(\boldsymbol{w}, \bar{\boldsymbol{\pi}}, \boldsymbol{\Phi}, \boldsymbol{H}_t) = q(\boldsymbol{w})q(\bar{\boldsymbol{\pi}})q(\boldsymbol{\Phi})q(\boldsymbol{H}_t)$ and compute the iterative steps as follows.

1. **Global Update:** The same as online MedHDP, except that the update rule for $\boldsymbol{w}$ is now (28) for the Gibbs classifier.

2. **Local Update:** This step involves drawing samples of the local variables. We develop a Gibbs sampler, which iteratively draws $\boldsymbol{S}_t$ from the local conditional in (41), draws $\bar{\boldsymbol{\pi}}$ from the conditional in (42), and draws the augmented variables $\boldsymbol{\lambda}_t$ from the conditional in (30). For $\boldsymbol{Z}_t$, we explain the sampling procedure in detail. Specifically, we infer $\boldsymbol{Z}_t$ through

$$q^*(\boldsymbol{Z}_t, \boldsymbol{S}_t, \boldsymbol{\lambda}_t, \bar{\boldsymbol{\pi}}) \propto \tilde{q}(\bar{\boldsymbol{\pi}}, \boldsymbol{Z}_t, \boldsymbol{S}_t)\hat{q}(\boldsymbol{Z}_t, \boldsymbol{\lambda}_t), \tag{43}$$

where

$$\hat{q}(\boldsymbol{Z}_t, \boldsymbol{\lambda}_t) = \prod_{d \in B_t} \frac{1}{\sqrt{2\pi\lambda_d}} \exp\left(\sum_{i \in [n_d]} \Lambda_{z_{di}, x_{di}} - \mathbb{E}_{q(\boldsymbol{\Phi}, \boldsymbol{w})}\left[\frac{(\lambda_d + c\zeta_d)^2}{2\lambda_d} \right] \right),$$

The Gibbs sampling for each variable z_{di} is

$$q(z_{di} = k | \boldsymbol{Z}_t^{\neg di}, \boldsymbol{\lambda_t}, \bar{\boldsymbol{\pi}}) \propto \frac{(\alpha\pi_k + C_{dk}^{\neg di})(C_{kx_{di}}^{\neg di} + \Delta_{kx_{di}}^t)}{\sum_w (C_{kw}^{\neg di} + \Delta_{kw}^t)} \exp\left(\frac{c y_d (c\epsilon + \lambda_d)\mu_k^*}{n_d \lambda_d} - \frac{c^2(\mu_k^{*2} + \Sigma_{kk}^* + 2(\mu_k^* \boldsymbol{\mu}^* + \boldsymbol{\Sigma}_{\cdot,k}^*)^\top \boldsymbol{C}_d^{\neg di})}{2 n_d^2 \lambda_d} \right), \tag{44}$$

while the probability of sampling a new topic is

$$q(z_{di} > K | \boldsymbol{Z}_t^{\neg di}, \bar{\boldsymbol{\pi}}) \propto \alpha \left(1 - \sum_{k=1}^K \pi_k \right) / W.$$

Again, we iterate the above steps till convergence for training and the testing is the same as online MedHDP. A batch version algorithm can be attained by setting $T = 1$ and $|B| = D$, which we denote as MedHDP$^{\text{gibbs}}$.

6.2 Multi-task Learning

The above models have been presented for classification. The basic ideas can be applied to solve other learning tasks, such as regression and multi-task learning (MTL). We use multi-task learning an one example. The primary assumption of multi-task learning is that by sharing statistical strength in a joint learning procedure, multiple related tasks can be mutually enhanced or some main tasks can be improved. MTL has many applications. We consider one scenario for multi-label classification. In this task, a set of binary classifiers are trained, each of which identifies whether a document $\boldsymbol{x}_d$ belongs to a specific category $y_d^q \in \{+1, -1\}$. These binary classifiers are allowed to share common latent representations and therefore could be attained via a modified BayesPA update equation:

$$\min_{q \in \mathcal{F}_t} \mathcal{L}\Big(q(\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t)\Big) + 2c \sum_{q=1}^{Q} \ell_\epsilon\Big(q(\boldsymbol{w}, \boldsymbol{\mathcal{M}}, \boldsymbol{H}_t); \boldsymbol{X}_t, \boldsymbol{Y}_t^\tau\Big),$$

where Q is the total number of tasks. We can then derive the multi-task version of Passive-Aggressive topic models, denoted by paMedLDAmt$^{\text{ave}}$ and paMedLDAmt$^{\text{gibbs}}$, in a way similar as in Section 5. We can further develop the nonparametric multi-task MedLDA topic models in a way similar as in Section 6.1 and the online PA learning algorithms. We denote the nonparametric online models by paMedHDP$^{\text{ave}}$ and paMedHDP$^{\text{gibbs}}$, according to whether the task-specific classifier is averaging or Gibbs.

7. Experiments

We demonstrate the efficiency and prediction accuracy of online MedLDA, online Gibbs MedLDA and their extensions on the 20Newsgroup (20NG) and a large Wikipedia dataset. A sensitivity analysis of the key parameters is also provided. Following the same setting in Zhu et al. (2012), we remove a standard list of stop words. All of the experiments are done on a normal computer with single-core clock rate up to 2.4 GHz.

7.1 Classification on 20Newsgroup

We perform multi-class classification on the entire 20NG dataset with all the 20 categories. The training set contains 11,269 documents, with the smallest category having 376 documents and the biggest category having 599 documents. The testing set contains 7,505 documents, with the smallest and biggest categories having 259 and 399 documents respectively. We adopt the "one-vs-all" strategy (Rifkin and Klautau, 2004) to combine binary classifiers for multi-class prediction tasks.

We use shorthand notations paMedLDA$^{\text{ave}}$ and paMedLDA$^{\text{gibbs}}$ for online MedLDA and online Gibbs MedLDA respectively. The corresponding batch learning algorithms are MedLDA (Zhu et al., 2012) and Gibbs MedLDA (MedLDA$^{\text{gibbs}}$) (Zhu et al., 2014b), which is a MedLDA model with Gibbs classifiers. We use collapsed Gibbs sampling to solve MedLDA, which is exactly the gMedLDA model proposed by Jiang et al. (2012). We also choose a state-of-the-art online unsupervised topic model as the baseline, the sparse inference for LDA (spLDA) (Mimno et al., 2012), which has been demonstrated to be superior than the stochastic variational LDA (Hoffman et al., 2013) in prediction performance. To perform the supervised tasks, we learn a linear SVM with the topic representations using LIBSVM (Chang and Lin, 2011). We refer the readers to (Zhu et al., 2012) for the performance of other batch-learning-based supervised topic models, such as sLDA

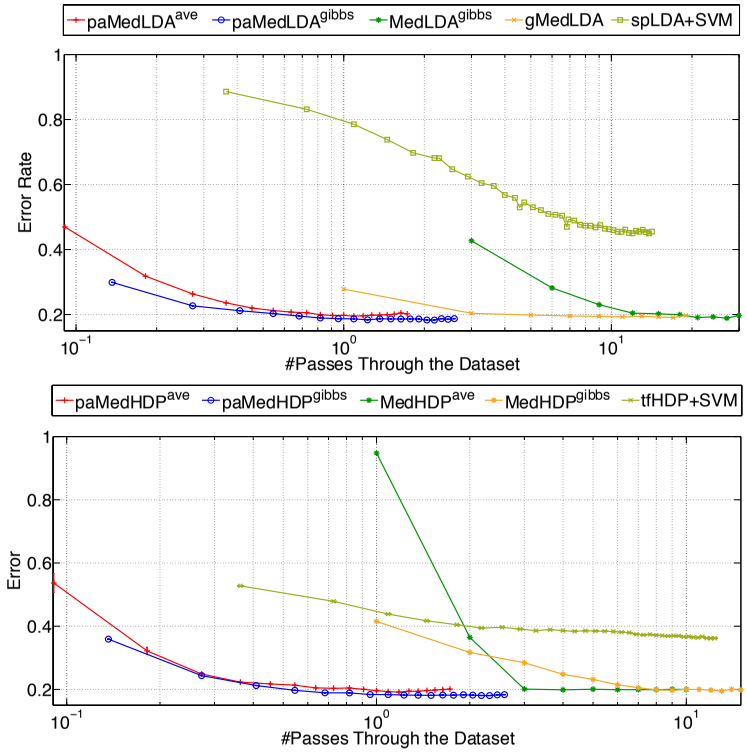


Figure 3: Test errors of different models with respect to the number of passes through the 20NG training dataset.

(Blei and McAuliffe, 2010) and DiscLDA (Lacoste-Julien et al., 2008), as well as SVM classifiers on the raw inputs, which are generally either inferior in accuracy or slower in efficiency than MedLDA. For all the LDA-based topic models, we use symmetric Dirichlet priors $\boldsymbol{\alpha} = 1/K \cdot \mathbf{1}$ and $\boldsymbol{\gamma} = 0.45 \cdot \mathbf{1}$. For BayesPA with Gibbs classifiers, the parameters were set at $\epsilon = 164$, $c = 1$, and $\sigma^2 = 1$. The models' performance is not sensitive to the choice of these parameters in wide ranges as shown in Zhu et al. (2014b). For BayesPA with averaging classifiers, the parameters determined by cross validation are $\epsilon = 16$, $c = 500$, and $\sigma^2 = 10^{-3}$. For reasons explained in section 7.3, we set the mini-batch size $|B| = 1$ for the averaging classifier and $|B| = 512$ for the Gibbs classifier.

We first analyze how many processed documents are sufficient for each model to converge. Figure 3 shows the test errors with respect to the number of passes through the 20NG training set, where the number of topics is set at $K = 80$ and the other parameters of BayesPA are set at

$(\mathcal{I}, \mathcal{J}, \beta) = (1, 2, 0)$. As we can observe, by solving a series of latent BayesPA problems, both paMedLDA$^{\text{ave}}$ and paMedLDA$^{\text{gibbs}}$ fully explore the redundancy of documents and converge in less than one pass, while their corresponding batch algorithms (i.e., MedLDA and MedLDA$^{\text{gibbs}}$) need many passes as burn-in steps. Besides, compared with the online learning algorithms for unsupervised topic models (i.e., spLDA+SVM), BayesPA topic models use supervising side information from each mini-batch, and therefore exhibit a faster convergence rate in discrimination ability. The convergence performance of BayesPA models is significantly better than that of the unsupervised spLDA.

Next, we study each model's best performance possible and the corresponding training time. To allow for a fair comparison, we train each model until the relative change of its objective is less than 10^{-4}. Figure 4 shows the prediction accuracy and training time of LDA-based models on the whole dataset with varying numbers of topics. We can see that BayesPA topic models are more than an order of magnitude faster than their corresponding batch algorithms in training time due to the power of online learning. paMedLDA$^{\text{ave}}$ is faster than paMedLDA$^{\text{gibbs}}$, because it does not need to update the covariance matrix of classifier weights $\boldsymbol{w}$. But the tradeoff is that for averaging models, they are more sensitive to the initial choice of σ^2, and therefore we need to use cross-validation to determine the best choice of variance beforehand. Furthermore, thanks to the merits of structured mean-field inference, which does not impose strict assumptions on the independence of latent variables, BayesPA topic models parallel their batch alternatives in accuracy. Moreover, all the supervised models are significantly better than the unsupervised spLDA in classification, reaching the state-of-the-art performance on the tested datasets.

Table 2 visualizes the learnt topic representation by paMedLDA$^{\text{ave}}$ and paMedLDA$^{\text{gibbs}}$. For the displayed categories, we plot the corresponding classifier's topic distribution averaged over the positive examples and top words from the topic matrix. As we can see, the average topic distributions become increasingly sparse as more and more data are observed. Eventually, the averaged topic distribution for each category contains only 1$\sim$2 non-zero entries and meanwhile different categories have quite diverse average topic distributions, therefore showing strong discriminative ability of the topic representations in distinguishing different categories. Such sparse and discriminatrive patterns are similar to what have been shown in batch settings (Zhu et al., 2012, 2014b).

7.2 Extensions

We now present the experimental results on the extensions of BayesPA topic models. We first present the results of nonparametric topic modeling on the same 20NG dataset. Then we demonstrate multi-task learning on a large Wikipedia dataset with more than 1 million documents and about 1 million unique terms.

7.2.1 NONPARAMETRIC TOPIC MODELING

Recall the nonparametric extensions of BayesPA topic models paMedHDP$^{\text{ave}}$ and paMedHDP$^{\text{gibbs}}$. To validate the advantage of online learning, we test them against their corresponding batch algorithms (i.e., the models MedHDP$^{\text{ave}}$ and MedHDP$^{\text{gibbs}}$) on the 20NG corpus. We also include an unsupervised baseline as the comparison model, the truncation-free HDP topic model (tfHDP) (Wang and Blei, 2012b), which is first used to discover the latent topic representations, and then combined with a linear SVM classifier for document categorization. For all HDP-based models, the following parameter setting is used: $\boldsymbol{\alpha} = 5 \cdot \mathbf{1}, \boldsymbol{\gamma} = 0.5 \cdot \mathbf{1}$ and $\eta = 1$. As an initial number of topic numbers

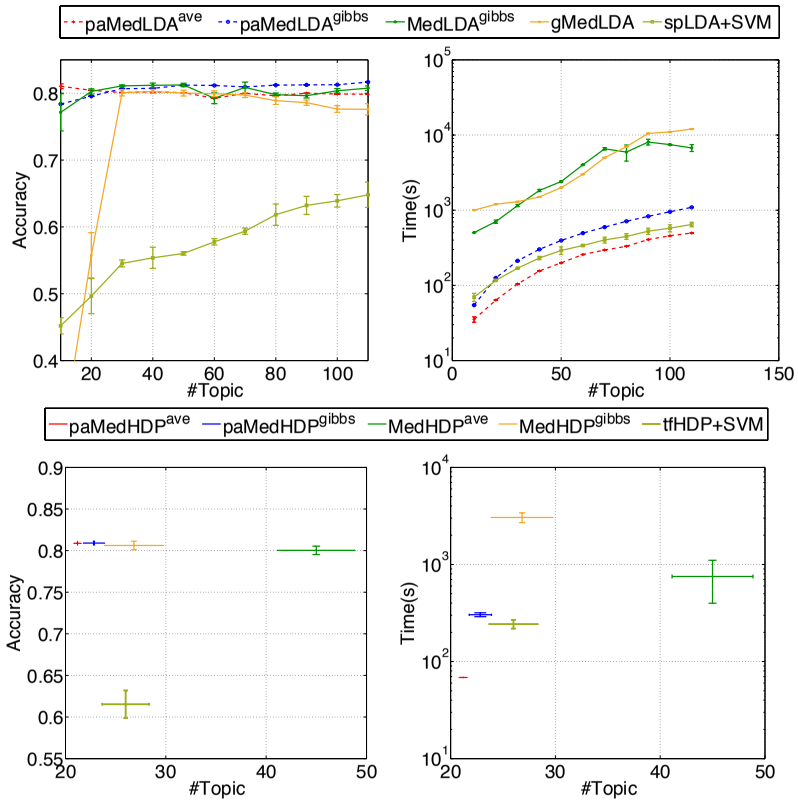
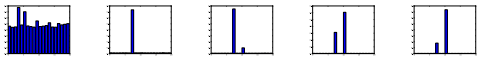
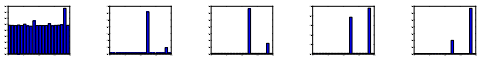


Figure 4: Classification accuracy and running time of various models with respect to the number of topics on the 20NG dataset.

for HDP to start with, we choose $K = 20$. We observed that the training time and the prediction accuracy do not depend heavily on the initial number of topics. The other parameters of BayesPA are the same.

Figure 3 shows the convergence of paMedHDP$^{\text{ave}}$ and paMedHDP$^{\text{gibbs}}$, and Figure 4 plots the accuracy and time together with the inferred topic numbers, where the length of the horizontal bars represents the variance of the inferred topic numbers. The results are summarized from five different runs. As we can see, the nonparametric extensions of BayesPA topic models also dramatically improve time efficiency and converge to corresponding batch algorithms in prediction performance. Furthermore, the averaging models are again faster to train because they do not need to update the covariance matrix of classifier weights.

Results by paMedLDA$^{\text{ave}}$								
Category	Visualization							
alt.altheism	#Observation	1		64		512	4096	11269
	Topic distribution							
	Top words	T8	writes	article	don	people	time	good
		T11	god	people	writes	don	article	atheism
comp.graphics	#Observation	1		64		512	4096	11269
	Topic distribution							
	Top words	T19	image	graphics	file	jpeg	files	software
		T13	writes	article	people	don	time	good
misc.forsale	#Observation	1		64		512	4096	11269
	Topic distribution							
	Top words	T4	sale	offer	shipping	mail	dos	price
		T8	writes	article	people	don	time	good

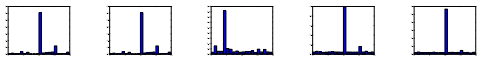
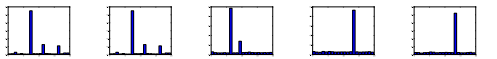
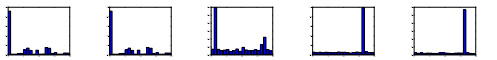
Results by paMedLDA$^{\text{gibbs}}$								
Category	Visualization							
alt.altheism	#Observation	1		64		512	4096	11269
	Topic distribution							
	Top words	T11	god	people	writes	article	don	time
		T16	god	people	don	writes	jesus	time
comp.graphics	#Observation	1		64		512	4096	11269
	Topic distribution							
	Top words	T14	graphics	image	file	program	images	writes
		T13	entry	space	don	people	system	writes
misc.forsale	#Observation	1		64		512	4096	11269
	Topic distribution							
	Top words	T17	sale	mail	dos	good	offer	shipping
		T18	don	writes	article	time	good	work

Table 2: Visualization of the learnt topics by paMedLDA$^{\text{ave}}$ and paMedLDA$^{\text{gibbs}}$. See text for details.

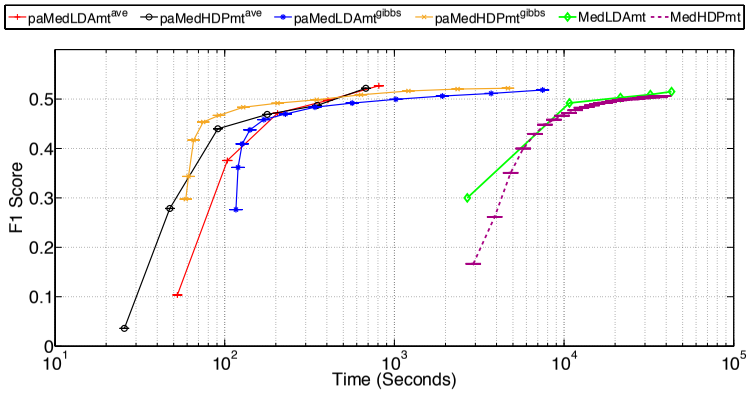


Figure 5: F1 scores of various multi-task topic models with respect to the running time on the 1.1M wikipedia dataset.

7.2.2 MULTI-TASK CLASSIFICATION

We test paMedLDAmt$^{\text{ave}}$, paMedLDAmt$^{\text{gibbs}}$ and their nonparametric exentions on a large Wiki dataset. The Wiki dataset is built from the Wikipedia set used in PASCAL LSHC challenge 2012 [6]. The Wiki dataset is a collection of documents with labels up to 20 different kinds, while the data distribution among the labels is balanced. The training set consists of 1.1 millions of wikipedia documents and the testing test consists of 5,000 documents. The vocabulary contains 917,683 unique terms. To measure performance, we use F1 score, the harmonic mean of precision and recall.

As baseline batch algorithms, we include MedLDAmt, a recent multi-task extention of Gibbs MedLDA (Zhu et al., 2013). Since MedHDP is a new model, there is no exising implementation of multi-task batch versions. So we instead extended MedHDP to support multi-task inference. We call this model MedHDPmt.

We use the same validation scheme as previous to select batchsize $|B| = 1, c = 5000, \sigma^2 = 10^{-6}$ for paMedLDAmt$^{\text{ave}}$; We choose $|B| = 512, c = 1, \sigma^2 = 1$ for paMedLDAmt$^{\text{gibbs}}$. For both models, the Dirichlet parameters are $\boldsymbol{\alpha} = 0.8 \cdot \mathbf{1}, \boldsymbol{\gamma} = 0.5 \cdot \mathbf{1}$, and $\epsilon = 1$. We use $K = 40$ topics for MedLDA models. The nonparametric extensions use exactly the same parameter settings except that $\boldsymbol{\alpha} = 5 \cdot \mathbf{1}, \eta = 1$ and we do not need to specify the topic number K.

Figure 5 shows the F1 scores of various models as a function of training time. We find that BayesPA topic models produce comparable results with their batch counterparts, but the training time is significantly less. With either Gibbs or averaging classifiers, BayesPA is about two orders of magnitude faster than their batch counterparts. Therefore, BayesPA topic models could potentially be applied to large-scale multi-class settings.

6. See http://lshtc.iit.demokritos.gr/.

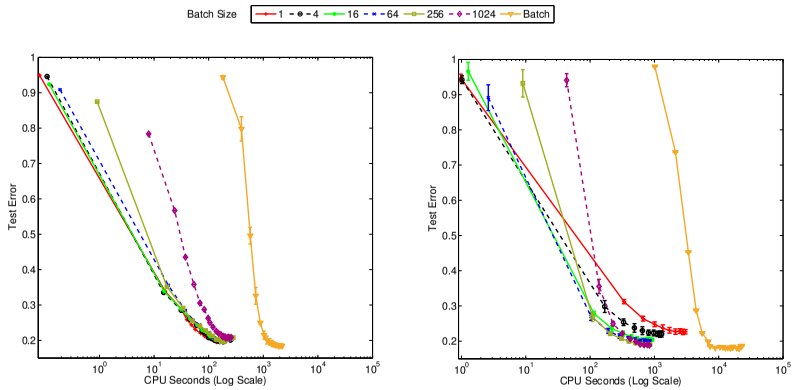


Figure 6: Test errors of paMedLDA$^{\text{ave}}$ (left) and paMedLDA$^{\text{gibbs}}$ (right) with different batch sizes on the 20NG dataset.

7.3 Sensitivity Analysis

We provide further discussions on BayesPA learning for topic models. We analyze the models' sensitivity to some key parameters.

Batch Size $|B|$: Figure 6 presents the test errors of BayesPA topic models (paMedLDA$^{\text{ave}}$, paMedLDA$^{\text{gibbs}}$) as a function of training time on the entire 20NG dataset with various batch sizes. The number of topics is fixed at $K = 40$. We can see that the convergence speeds of different algorithms vary. First of all, the batch algorithms suffer from multiple passes through the dataset and therefore are much slower than the online alternatives. Second, we could observe that algorithms with medium batch sizes ($|B| = 64$ or 256) converge faster. If we choose a batch size too small, for example, $|B| = 1$, each iteration would not provide sufficient evidence for the update of global variables; if the batch size is too large, each mini-batch becomes redundant and the convergence rate also decreases. By comparing the two figures, we find that paMedLDA$^{\text{ave}}$ runs faster than paMedLDA$^{\text{gibbs}}$. This is because for averaging classifers, we do not update the covaraince of the classifier weights, which requires frequent matrix inverse operations. Furthermore, paMedLDA$^{\text{ave}}$ appears to be more robust against change in batchsize. Similarly, Figure 7 shows the sensitivity experiment of the batchsize parameter in paMedHDP models. The results are similar to paMedLDA models, that is, a moderate batchsize leads to faster convergence.

Number of iterations $\mathcal{I}$ and samples $\mathcal{J}$: Since the time complexity of Algorithm 2 is linear in both $\mathcal{I}$ and $\mathcal{J}$, we would like to know how these parameters influence the quality of the trained models. First, we analyze which setting of $(\mathcal{I}, \mathcal{J})$ guarantees good performance. Fix $\beta = 0, K = 40$. Figure 8 presents the results. First, the number of samples $\mathcal{J}$ does not have a large effect on the accuracy. Second, the peformance of paMedLDA$^{\text{gibbs}}$ and paMedHDP$^{\text{gibbs}}$ is not sensitive the number of optimization round, but paMedLDA$^{\text{ave}}$ and paMedHDP$^{\text{ave}}$ suffers largely if $\mathcal{I} = 1$. This is because with averaging classifiers, the sampling of latent variable $\boldsymbol{Z}$ relies not only on global

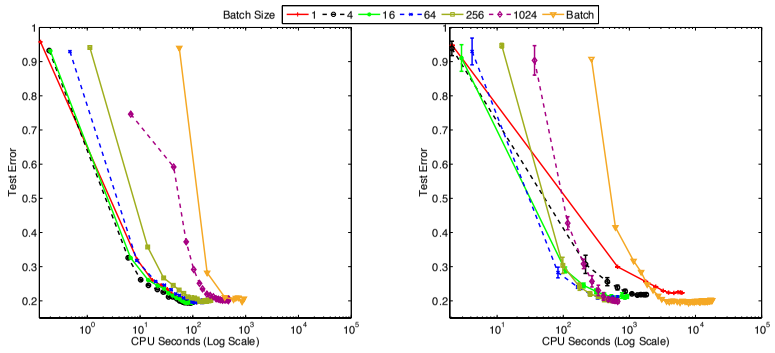


Figure 7: Test errors of paMedHDP$^{\text{ave}}$ (left) and paMedHDP$^{\text{gibbs}}$ (right) with different batch sizes on the 20NG dataset.

parameters, but also on a local variable τ, so more optimization rounds are needed. The training time of all models scale linearly in terms of $\mathcal{I}$ and $\mathcal{J}$.

$\mathcal{J}$ \ β	0	2	4
1	0.802		
3	0.803	0.803	
5	0.805	0.805	0.803

(a)

$\mathcal{J}$ \ β	0	2	4
1	0.783		
3	0.803	0.799	
5	0.808	0.803	0.792

(b)

Table 3: Effect of the number of local samples and burn-in steps for **(a).** paMedLDA$^{\text{ave}}$; and **(b).** paMedLDA$^{\text{gibbs}}$.

Notice that the first β samples are discarded as burn-in steps. To understand how large β is sufficient, we consider the settings of the pairs $(\mathcal{J}, \beta)$ and check the prediction accuracy of Algorithm 2 for $K = 40$. Based on the sensitivity analysis of $\mathcal{I}$ and $\mathcal{J}$, we fix $\mathcal{I} = 1$ for paMedLDA$^{\text{gibbs}}$, paMedHDP$^{\text{gibbs}}$ and $\mathcal{I} = 2$ for paMedLDA$^{\text{ave}}$, paMedHDP$^{\text{ave}}$. The results are shown in Table 3. We can see that accuracy scores closer to the diagonal of the table are relatively lower, while settings with the same number of kept samples, e.g. $(\mathcal{J}, \beta) = (3, 0), (5, 2), (9, 6)$, yield similar results. Therefore, the number of kept samples exhibits a more significant role in the performance of BayesPA topic models than the number of burn-in steps.

Number of Topics. Finally, we observed an accuracy degradation phenomenon when running one-pass BayesPA with a large number of topics on small datasets. To see this effect, we test on a small binary classification dataset, which consists of the documents from the two groups $alt.atheism$ and $talk.religion.misc$ in the 20 newsgroup dataset. The training set contains 1,597

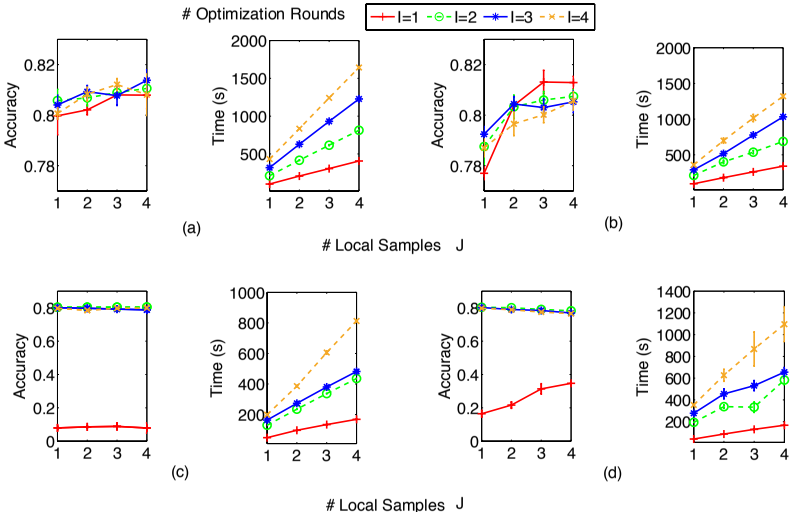


Figure 8: Classification accuracies and training time of **(a)**: paMedLDA$^{\text{gibbs}}$, **(b):** paMedHDP$^{\text{gibbs}}$, **(c)**: paMedLDA$^{\text{ave}}$, and **(d):** paMedHDP$^{\text{ave}}$, with different combinations of $(\mathcal{I}, \mathcal{J})$ on the 20NG dataset. The x-axis includes different values of $\mathcal{J}$, while different values of $\mathcal{I}$ are shown as separate lines.

documents with a split of 795/802 over the two categories and the test set contains 401 documents with a split of 204/197. For clarity, we report the results of the online BayesPA with a Gibbs MedLDA classifier, whose hyper-parameters are set as in the multi-class setting.

As shown in Figure 9 (the red curve), when the number of topics increases to relatively large, the classification accuracy drops significantly for Gibbs MedLDA with a single pass of the dataset. We believe this phenomenon is due to the fact that with a limited number of data points and a large number of parameters, the online BayesPA fails to converge in just one pass. Two possible strategies can be used to make this algorithm practically useful:

- **Multiple Scans**: In this strategy, we simply scan the dataset for multiple passes. As shown in Figure 9, this method works well — with five passes, BayesPA can learn the topic model with more than 100 topics well and lead to improved accuracy when K is in the range of $[50, 100]$. However, one disadvantage of this strategy is that it takes more computational resources, linear to the number of scans.

- **Ghost Samples**: In this strategy, we create ghost copies of the data, that is, for every data point $(\boldsymbol{x}_d, y_d)$, we make $L - 1$ exact clones. Then in our global update equation, each aggregation

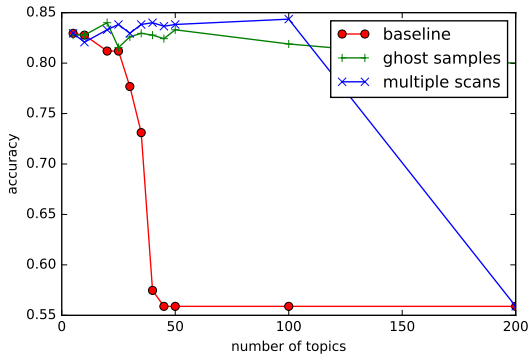


Figure 9: Solving the performance degradation issue with a couple of practical approaches: **baseline**: the model without any modifications, where degradation issues emerge. **ghost copies**: create $L = 10$ copies for every mini-batch. **multiple scan**: run the model with total of five passes over the dataset.

of sufficient statistics will be multiplied by an extra factor of L. This strategy works well too, as shown in Figure 9, although slightly worse than the strategy of multiple scans. One advantage of this strategy is that it does not increase the computation burden.

8. Conclusions and Future Work

We present online Bayesian Passive-Aggressive (BayesPA) learning as a new framework for max-margin Bayesian inference on streaming data. For fixed but large-scale datasets, online BayesPA effectively explores the statistical redundancy by repeatedly drawing samples and leads to faster convergence. We show that BayesPA subsumes the online PA, and more significantly generalizes naturally to incorporate latent variables and to perform nonparametric Bayesian inference, therefore providing great flexibility for explorative analysis. Based on the ideas of BayesPA, we develop efficient online learning algorithms for max-margin topic models as well as their nonparametric extensions which can automatically infer the unknown topic numbers. Empirical experiments on 20Newsgroups and a large-scale Wikipedia multi-label dataset demonstrate significant improvements on time efficiency, while maintaining comparable results.

As for future work, we are interested in developing highly scalable, distributed (Broderick et al., 2013) BayesPA learning paradigms, which will better meet the demand of processing massive real data available today. We are also interested in applying BayesPA to develop efficient algorithms for more sophisticated max-margin Bayesian models, such as the latent feature relational models (Zhu, 2012). Finally, a theoretical analysis of the regret bounds for BayesPA needs to be systematically carried out for both averaging and Gibbs classifiers.

Acknowledgements

This work is supported by National Key Foundation R&D Projects (No. 2013CB329403), the National Natural Science Foundation of China (NSFC) (Nos. 61620106010, 61322308, 61332007), the NSFC and German Research Foundation (DFG) in Project Crsossmodal Learning (No. NSFC 61621136008 / DFC TRR-169), and the National Youth Top-notch Talent Support Program. J.Z is also partly supported by the collaborative projects from Intel and Tencent.

Appendix A.

We show the objective in (27) is an upper bound of that in (13), that is,

$$\mathcal{L}\Big(q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t, \boldsymbol{\lambda}_t)\Big) - \mathbb{E}_q\Big[\log(\psi(\boldsymbol{Y}_t, \boldsymbol{\lambda}_t | \boldsymbol{Z}_t, \boldsymbol{w}))\Big] \geq \mathcal{L}\Big(q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t)\Big) + 2c \sum_{d \in B_t} \mathbb{E}_q\Big[(\xi_d)_+\Big], \quad (45)$$

where $\mathcal{L}(q) = \mathbf{KL}[q || q_t(\boldsymbol{w}, \boldsymbol{\Phi}) q_0(\boldsymbol{Z}_t)]$.

Proof We first have

$$\mathcal{L}\Big(q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t, \boldsymbol{\lambda}_t)\Big) = \mathbb{E}_q\left[\log \frac{q(\boldsymbol{\lambda}_t \mid \boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t) q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t)}{q_t(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t)}\right],$$

and

$$\mathcal{L}\Big(q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t)\Big) = \mathbb{E}_q\left[\log \frac{q(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t)}{q_t(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t)}\right].$$

Comparing these two equations and canceling out common factors, we know that in order for (45) to make sense, it suffices to prove

$$\mathbb{H}[q'] - \mathbb{E}_{q'}[\log(\psi(\boldsymbol{Y}_t, \boldsymbol{\lambda}_t | \boldsymbol{Z}_t, \boldsymbol{w})] \geq 2c \sum_{d \in B_t} \mathbb{E}_{q'}[(\xi_d)_+] \quad (46)$$

is uniformly true for any given $(\boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t)$, where $\mathbb{H}(\cdot)$ is the entropy operator and $q' = q(\boldsymbol{\lambda}_t \mid \boldsymbol{w}, \boldsymbol{\Phi}, \boldsymbol{Z}_t)$. The inequality (46) can be reformulated as

$$\mathbb{E}_{q'}\left[\log \frac{q'}{\psi(\boldsymbol{Y}_t, \boldsymbol{\lambda}_t | \boldsymbol{Z}_t, \boldsymbol{w})}\right] \geq 2c \sum_{d \in B_t} \mathbb{E}_{q'}[(\xi_d)_+] \quad (47)$$

Exploiting the convexity of the function $\log(\cdot)$, i.e.

$$-\mathbb{E}_{q'}\left[\log \frac{\psi(\boldsymbol{Y}_t, \boldsymbol{\lambda}_t | \boldsymbol{Z}_t, \boldsymbol{w})}{q'}\right] \geq -\log \int_{\boldsymbol{\lambda}_t} \psi(\boldsymbol{Y}_t, \boldsymbol{\lambda}_t | \boldsymbol{Z}_t, \boldsymbol{w})\, d\boldsymbol{\lambda}_t,$$

and utilizing the equality (26), we then have (47) and therefore prove (45). ■

References

Sungjin Ahn, Anoop Korattikara, and Max Welling. Bayesian posterior sampling via stochastic gradient Fisher scoring. In *International Conference on Machine Learning (ICML)*, 2012.

Charles E. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *Annals of Statistics*, pages 1152–1174, 1974.

Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

Rémi Bardenet, Arnaud Doucet, and Chris Holmes. Towards scaling up Markov chain Monte Carlo: an adaptive subsampling approach. In *International Conference on Machine Learning (ICML)*, 2014.

David M. Blei and Jon D. McAuliffe. Supervised topic models. In *Neural Information Processing Systems (NIPS)*, 2010.

David M. Blei, Andrew Ng, and Michael I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

Tamara Broderick, Nicholas Boyd, Andre Wibisono, Ashia C Wilson, and Michael Jordan. Streaming variational bayes. In *Neural Information Processing Systems (NIPS)*, pages 1727–1735, 2013.

Kevin R. Canini, Lei Shi, and Thomas Griffiths. Online inference of topics with latent Dirichlet allocation. In *International Conference on Artificial Intelligence and Statistics*, pages 65–72, 2009.

Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

Jonathan Chang and David M Blei. Hierarchical relational models for document networks. *The Annals of Applied Statistics*, pages 124–150, 2010.

Ning Chen, Jun Zhu, Fei Xia, and Bo Zhang. Discriminative relational topic models. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 37(5):973–986, 2015.

David Chiang, Yuval Marton, and Philip Resnik. Online large-margin training of syntactic and structural translation features. In *Conference on Empirical Methods on Natural Language Processing (EMNLP)*, 2008.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalel-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research (JMLR)*, 7:551–585, 2006.

Koby Crammer, Mark Dredze, and Fernando Pereira. Exact convex confidence-weighted learning. In *Neural Information Processing Systems (NIPS)*, 2008.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal statistical Society*, 39(1):1–38, 1977.

Luc Devroye. *Non-Uniform Random Variate Generation*. Springer, 1986.

Arnaud Doucet and Adam M Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, 12:656–704, 2009.

Arnaud Doucet, Nando De Freitas, Kevin Murphy, and Stuart Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 176–183. Morgan Kaufmann Publishers Inc., 2000.

Mark Dredze, Koby Crammer, and Fernando Pereira. Confidence-weighted linear classification. In *International Conference on Machine Learning (ICML)*, pages 264–271, 2008.

Rick Durret. *Probabiilty: Theory and Examples*. Cambridge University Press, 2010.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

Yang Gao, Jianfei Chen, and Jun Zhu. Streaming gibbs sampling for LDA model. *arXiv preprint arXiv:1601.01142*, 2016.

Zoubin Ghahramani and Thomas Griffiths. Infinite latent feature models and the Indian buffet process. In *Neural Information Processing Systems (NIPS)*, 2005.

Thomas Griffiths and Mark Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101(Suppl 1):5228–5235, 2004.

James Hannan. Approximation to Bayes risk in repeated play. *Contributions to the Theory of Games*, 3:97–139, 1957.

Nils Lid Hjort. *Bayesian Nonparametrics*. Cambridge University Press, 2010.

Matthew Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research (JMLR)*, 14(1):1303–1347, 2013.

Antti Honkela and Harri Valpola. Online variational Bayesian learning. In *4th International Symposium on Independent Component Analysis and Blind Signal Separation*, pages 803–808, 2003.

Tommi Jaakkola, Marina Meila, and Tony Jebara. Maximum entropy discrimination. In *Neural Information Processing Systems (NIPS)*, 1999.

Tony Jebara. *Discriminative, Generative and Imitative learning*. PhD thesis, Massachusetts Institute of Technology, 2001.

Tony Jebara. Multitask sparsity via maximum entropy discrimination. *Journal of Machine Learning Research (JMLR)*, 12:75–110, 2011.

Qixia Jiang, Jun Zhu, Maosong Sun, and Eric Xing. Monte Carlo methods for maximum margin supervised topic models. In *Neural Information Processing Systems (NIPS)*, 2012.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. *An Introduction to Variational Methods for Graphical Models*. Springer, 1998.

Anoop Korattikara, Yutian Chen, and Max Welling. Austerity in MCMC land: Cutting the Metropolis-Hastings budget. In *International Conference on Machine Learning (ICML)*, 2014.

Simon Lacoste-Julien, Fei Sha, and Michael I Jordan. DiscLDA: Discriminative learning for dimensionality reduction and classification. In *Neural Information Processing Systems (NIPS)*, 2008.

Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine learning*, 2(4):285–318, 1988.

Jan Luts, Tamara Broderick, and Matt Wand. Real-time semiparametric regression. *arXiv:1209.3550*, 2013.

Ryan McDonald, Koby Crammer, and Fernando Pereira. Online large-margin training of dependency parsers. In *Annual Meeting of the Association for Computational Linguistics*, 2005.

Shike Mei, Jun Zhu, and Xiaojin Zhu. Robust RegBayes: Selectively incorporating first-order logic domain knowledge into Bayesian models. In *International Conference on Machine Learning*, 2014.

John R. Michael, William R. Schucany, and Roy W. Haas. Generating random variates using transformations with multiple roots. *Journal of the American Statistician*, 30(2):88–90, 1976.

David Mimno, Matthew Hoffman, and David M. Blei. Sparse stochastic inference for latent Dirichlet allocation. *International Conference on Machine Learning (ICML)*, 2012.

Thomas P. Minka. Expectation propagation for approximate Bayesian inference. In *Uncertainty in Artificial Intelligence (UAI)*. Morgan Kaufmann Publishers Inc., 2001.

Kevin P. Murphy. *Machine Learning: a Probabilistic Perspective*. MIT Press, 2012.

Sam Patterson and Yee Whye Teh. Stochastic gradient Riemannian Langevin dynamics on the probability simplex. In *Neural Information Processing Systems (NIPS)*, pages 3102–3110, 2013.

Nicholas G. Polson and Steven L. Scott. Data augmentation for support vector machines. *Bayesian Analysis*, 6(1):1–23, 2011.

Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research (JMLR)*, 5:101–141, 2004.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 1958.

Shai Shalev-Shwartz and Yoram Singer. Convex repeated games and Fenchel duality. In *Neural Information Processing Systems (NIPS)*, pages 1265–1272, 2006.

Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for SVM. *Mathematical Programming*, 127(1):3–30, 2011.

Tianlin Shi and Jun Zhu. Online Bayesian passive-aggressive learning. In *International Conference on Machine Learning (ICML)*, 2014.

Jacob Steinhardt and Percy Liang. Filtering with abstract particles. In *International Conference on Machine Learning (ICML)*, 2014.

Robert H. Swendsen and Jian-Sheng Wang. Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, 58(2):86–88, 1987.

Martin A. Tanner and Wing Hung Wong. The calculation of posterior distributions by data augmentation. *Journal of the American statistical Association*, 82(398):528–540, 1987.

Ben Taskar, Carlos Guestrin, and Daphne Koller. Max-margin Markov networks. In *Neural Information Processing Systems (NIPS)*, 2003.

Yee Whye Teh, M.I. Jordan, M.J. Beal, and David M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476), 2006a.

Yee Whye Teh, David Newman, and Max Welling. A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation. In *Neural Information Processing Systems (NIPS)*, 2006b.

David Van Dyk and Xiao-Li Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 10(1), 2001.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008.

Chong Wang and David M. Blei. A split-merge MCMC algorithm for the hierarchical Dirichlet process. *arXiv preprint arXiv:1201.1657*, 2012a.

Chong Wang and David M. Blei. Truncation-free online variational inference for Bayesian nonparametric models. In *Neural Information Processing Systems (NIPS)*, 2012b.

Chong Wang, John Paisley, and David M. Blei. Online variational inference for the hierarchical Dirichlet process. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *International Conference on Machine Learning (ICML)*, pages 681–688, 2011.

Max Welling, Yee Whye Teh, and Hilbert J. Kappen. Hybrid variational/Gibbs collapsed inference in topic models. In *Uncertainty in Artificial Intelligence (UAI)*, 2008.

Minjie Xu, Jun Zhu, and Bo Zhang. Fast max-margin matrix factorization with data augmentation. In *International Conference on Machine Learning (ICML)*, pages 978–986, 2013.

Jun Zhu. Max-margin nonparametric latent feature models for link prediction. In *International Conference on Machine Learning (ICML)*, 2012.

Jun Zhu and Eric P. Xing. Maximum entropy discrimination Markov networks. *Journal of Machine Learning Research (JMLR)*, 10:2531–2569, 2009.

Jun Zhu, Ning Chen, and Eric P Xing. Infinite SVM: a Dirichlet process mixture of large-margin kernel machines. In *International Conference on Machine Learning (ICML)*, pages 617–624, 2011.

Jun Zhu, Amr Ahmed, and Eric P Xing. MedLDA: maximum margin supervised topic models. *Journal of Machine Learning Research (JMLR)*, 13:2237–2278, 2012.

Jun Zhu, Xun Zheng, Li Zhou, and Bo Zhang. Scalable inference in max-margin topic models. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2013.

Jun Zhu, Jianfei Chen, and Wenbo Hu. Big learning with Bayesian methods. *arXiv preprint arXiv:1411.6370*, 2014a.

Jun Zhu, Ning Chen, Hugh Perkins, and Bo Zhang. Gibbs max-margin topic models with data augmentation. *Journal of Machine Learning Research (JMLR)*, 15:1073–1110, 2014b.

Jun Zhu, Ning Chen, and Eric P Xing. Bayesian inference with posterior regularization and applications to infinite latent SVMs. *Journal of Machine Learning Research (JMLR)*, 15:1799–1847, 2014c.

Journal of Machine Learning Research 18 (2017) 1-41 Submitted 3/15; Revised 2/17; Published 4/17

Asymptotic Analysis of Objectives Based on Fisher Information in Active Learning

Jamshid Sourati sourati@ece.neu.edu
Department of Electrical and Computer Engineering
Northeastern University
Boston, MA 02115 USA

Murat Akcakaya akcakaya@pitt.edu
Department of Electrical and Computer Engineering
University of Pittsburgh
Pittsburgh, PA 15261 USA

Todd K. Leen todd.leen@georgetown.edu
Georgetown University
Washington D.C. 20057 USA

Deniz Erdogmus erdogmus@ece.neu.edu
Jennifer G. Dy jdy@ece.neu.edu
Department of Electrical and Computer Engineering
Northeastern University
Boston, MA 02115 USA

Editor: Qiang Liu

Abstract

Obtaining labels can be costly and time-consuming. Active learning allows a learning algorithm to intelligently query samples to be labeled for a more efficient learning. Fisher information ratio (FIR) has been used as an objective for selecting queries. However, little is known about the theory behind the use of FIR for active learning. There is a gap between the underlying theory and the motivation of its usage in practice. In this paper, we attempt to fill this gap and provide a rigorous framework for analyzing existing FIR-based active learning methods. In particular, we show that FIR can be asymptotically viewed as an upper bound of the expected variance of the log-likelihood ratio. Additionally, our analysis suggests a unifying framework that not only enables us to make theoretical comparisons among the existing querying methods based on FIR, but also allows us to give insight into the development of new active learning approaches based on this objective.

Keywords: classification active learning, Fisher information ratio, asymptotic log-loss, upper-bound minimization.

1. Introduction

In supervised learning, a *learner* is a model-algorithm pair that is optimized to (semi) automatically perform tasks, such as classification, or regression using information provided by an external source (oracle). In *passive learning*, the learner has no control over the information given. In *active learning*, the learner is permitted to *query* certain types of information from the oracle (Cohn et al., 1994). Usually there is a cost associated with obtaining information from an oracle; therefore an active learner will need to maximize the information gained from queries within a fixed budget or minimize the cost of gaining a desired level of information. A majority of the existing algorithms restrict to the former problem, to get the most efficiently trained learner by querying a fixed amount of knowledge (Settles, 2012; Fu et al., 2013).

Active learning is the process of coupled querying/learning strategies. In such an algorithm, one needs to specify a query quality measure in terms of the learning method that uses the new information gained at each step of querying. For instance, information theoretic measures are commonly employed in classification problems to choose training samples whose class labels, considered as random variables, are most informative with respect to the labels of the remaining unlabeled samples. This family of measures is particularly helpful when probabilistic approaches are used for classification. Among these objectives, Fisher information criterion is very popular due to its relative ease of computation compared to other information theoretic objectives, desirable statistical properties and existence of effective optimization techniques. However, as we discuss in this manuscript, this objective is not well-studied in the classification context and there seems to be a gap between the underlying theory and the motivation of its usage in practice. *This paper is an attempt to fill this gap and also provide a rigorous framework for analyzing the existing querying methods based on Fisher information.*

From the statistical point of view, we characterize the process of constructing a classifier in three steps as follows: (1) choosing the loss and risk functions, (2) building a decision rule that minimizes the risk, and (3) modeling the discriminant functions of the decision rule. For instance, choosing the simple 0/1 loss and its a posteriori expectation as the risk, incurs the Bayes rule as the optimal decision (Duda et al., 1999), where the discriminant function is the posterior distribution of the class labels given the covariates. For this type of risk, discriminative models that directly parametrize the posteriors, such as logistic regression, are popularly used to learn the discriminant functions (Bishop, 2006). In order to better categorize the existing techniques, we break an active learning algorithm into the following sub-problems:

(i) *(Query Selection)* Sampling a set of covariates $\{\mathbf{x}_1, ..., \mathbf{x}_n\}$ from the *training* marginal[1], whose labels $\{y_1, ..., y_n\}$ are to be requested from an external source of knowledge (the *oracle*). The queried covariates together with their labels form the *training* data set.

(ii) *(Inference)* Estimating parameters of the posterior model based on the training data set formed in the previous step.

(iii) *(Prediction)* Making decisions regarding class labels of the *test* covariates sampled from the *test* marginal.

These three steps can be carried out iteratively. Note that the query selection subproblem is formulated in terms of the distribution from which the queries will be drawn.

1. Throughout this paper, marginal distribution or simply distribution refers to the distribution of covariates, while joint distribution is used for pairs of the covariates and their class labels.

©2017 Jamshid Sourati, Murat Akcakaya, Todd K. Leen, Deniz Erdogmus and Jennifer G. Dy.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-104.html.

Ideally, queries (or the query distribution) are chosen such that they increase the expected quality of the classification performance measured by a particular objective function. This objective can be constructed from two different perspectives: based on the accuracy of the parameter inference or the accuracy of label prediction. In the rest of the manuscript, accordingly, we refer to the algorithms that use these two types of objectives as *inference-based* or *prediction-based* algorithms, respectively.

Most of the inference-based querying algorithms in classification aim to choose queries that maximize the expected change in the objective of the inference step (Settles et al., 2008; Guo and Schuurmans, 2008) or Fisher information criterion (Hoi et al., 2006; Settles and Craven, 2008; Hoi et al., 2009; Chaudhuri et al., 2015b). On the other hand, the wide range of studies in prediction-based active learning includes a more varied set of objectives: for instance, the prediction error probability[2] (Cohn et al., 1994; Freund et al., 1997; Zhu et al., 2003; Nguyen and Smeulders, 2004; Dasgupta, 2005; Dasgupta et al., 2005; Balcan et al., 2006; Dasgupta et al., 2007; Beygelzimer et al., 2010; Hanneke et al., 2011; Hanneke, 2012; Awasthi et al., 2013; Zhang and Chaudhuri, 2014), variance of the predictions (Cohn et al., 1996; Schein and Ungar, 2007; Ji and Han, 2012), uncertainty of the learner with respect to the unknown labels as evaluated by the entropy function (Holub et al., 2008), mutual information (Guo and Greiner, 2007; Krause et al., 2008; Guo, 2010; Sourati et al., 2016), and margin of the samples with respect to the trained hyperplanar discriminant function (Schohn and Cohn, 2000; Tong and Koller, 2002).

In this paper, we focus on the Fisher information criterion used in classification active learning algorithms. These algorithms use a scalar function of the Fisher information matrices computed for parametric models of training and test marginals. In the classification context, this scalar is sometimes called *Fisher information ratio (FIR)* (Settles and Craven, 2008) and its usage is motivated by older attempts in optimal experiment design for statistical regression methods (Fedorov, 1972; MacKay, 1992; Cohn, 1996; Fukumizu, 2000).

Among the existing FIR-based classification querying methods, only the very first one proposed by Zhang and Oles (2000) approached the FIR objective from a parameter inference point of view. Using a maximum likelihood estimator (MLE), they claimed (with the proof skipped) that FIR is asymptotically equal to the expectation of the log-likelihood ratio with respect to both test and training samples (see sub-problem i above). Later on, Hoi et al. (2006) and Hoi et al. (2009), inspired by Zhang and Oles (2000), used FIR in connection with a logistic regression classifier with the motivation of decreasing the labels' uncertainty and hence the prediction error. Settles and Craven (2008) employed this objective with the same motivation, but using a different approximation and optimization technique. More recently, Chaudhuri et al. (2015b) showed that even finite-sample FIR is closely related to the expected log-likelihood ratio of an MLE-based classifier. However, their results are derived under a different and rather restricting set of conditions and assumptions: they focused on the finite-sample case where the test marginal is a uniform PMF and the proposal marginal is a general PMF (to be determined) over a finite pool of unlabeled samples. Moreover, they assumed that the conditional Fisher information matrix is assumed to be independent of the class labels. Here, in a framework similar to Zhang and Oles (2000) but with a more

2. Prediction error probability is indeed the frequentist risk function of 0/1 loss, and is also known as *generalization error*.

expanded and different derivation, we discuss a novel theoretical result based on which FIR is related to an MLE-based inference step for a large number of training data. More specifically, under certain regularity conditions required for consistency of MLE and in the absence of model mis-specification, and with no restricting assumptions on the form of test or training marginals, we show that FIR can be viewed as an upper bound for the expected variance of the asymptotic distribution of the log-likelihood ratio. Inspired by Chaudhuri et al. (2015b), we also show that under certain extra conditions, this relationship holds even in finite-sample case.

There are two practical issues in employing FIR as a query selection objective: its computation and optimization. First, computing the Fisher information matrices is usually intractable, except for very simple distributions; also FIR depends on the true marginal, which is usually unknown. Therefore, even if the computations are tractable, approximations have to be used for evaluating FIR. Second, the optimization of FIR is straightforward only if a single query is to be selected per iteration, or when the optimization has continuous domain (e.g., optimizing to get the real parameters of the query marginal as in Fukumizu, 2000). However, the optimization becomes NP-hard when multiple queries are to be selected from a countable set of unlabeled samples (*pool-based batch* active learning). Heuristics have been used to approximate such combinatorial optimization, such as greedy methods (Settles and Craven, 2008) and relaxation to continuous domains (Guo, 2010). Another strategy is to take advantage of *monotonic submodularity* of the objective set functions. If the objective is shown to be monotonically submodular, efficient greedy algorithms can be used for optimization with guaranteed tight bounds (Krause et al., 2008; Azimi et al., 2012; Chen and Krause, 2013). Regarding FIR, Hoi et al. (2006) proved that, when a logistic regression model is used, a Monte-Carlo simulation of this objective is a monotone and submodular set function in terms of the queries.

In addition to our theoretical contribution in asymptotically relating FIR to the log-likelihood ratio, we clarify the differences between some of the existing FIR-based querying methods according to the techniques that they use to address the evaluation and optimization issues. Furthermore, we show that monotonicity and submodularity of Monte-Carlo approximation of FIR can be extended from logistic regression models to *any* discriminative classifier. Here is a summary of our contributions in this paper:

- Establishing a relationship between the Fisher information matrix of the query distribution and the asymptotic distribution of the log-likelihood ratio (Section 4.1);
- Showing that FIR can be viewed as an upper bound of the expected asymptotic variance of the log-likelihood ratio, implying that minimizing FIR, as an active learning objective, is asymptotically equivalent to upper-bound minimization of the expected variance of the log-likelihood ratio, as a measure of inference performance (Section 4.2);
- Proving that under certain assumptions, the above-mentioned asymptotic relationship also holds for finite-sample estimation of FIR (Section 5.1.1);
- Discussing different existing methods for coping with practical issues in using FIR in querying algorithms (Section 5.1), and accordingly providing a unifying framework for existing FIR-based active learning methods (Section 5.2).

- Proving submodularity for the Monte-Carlo simulation of FIR under *any* discriminative classifier, assuming a pool-based active learning which enables access to approximations of Fisher information matrices of both test and training distributions (Lemma 7 and Theorem 8).

Before going through the main discussion in Section 4, we formalize our classification model assumptions, set the notations and review the basics and some of the key properties of our inference method, maximum likelihood estimation, in Sections 2 and 3. The statistical background required to follow the remaining sections is given in Appendix A.

2. The Framework and Assumptions

In this paper, we deal with classification problems, where each covariate, represented by a feature vector $\mathbf{x}$ in vector space X, is associated with a numerical class label y. Assuming that there are $1 < c < \infty$ classes, y can take any integer among the set $\{1, ..., c\}$. Suppose that the pairs $(\mathbf{x}, y)$ are distributed according to a parametric joint distribution $p(\mathbf{x}, y|\,\boldsymbol{\theta})$, with the parameter space denoted by $\Omega \subseteq \mathbb{R}^d$. Using a set of observed pairs as the training data, $\mathcal{L}_n := \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$, we can estimate $\boldsymbol{\theta}$ and predict the class labels of the unseen test samples, e.g., by maximizing $p(y|\,\mathbf{x}, \boldsymbol{\theta})$. In active learning, the algorithm is permitted to take part in designing $\mathcal{L}_n$ by choosing a set of data points $\{\mathbf{x}_1, ..., \mathbf{x}_n\}$, for which the class labels are then generated using an external oracle.

In addition to the framework described in the last section (see subproblems i to iii), we make the following assumptions regarding the oracle, our classification model and the underlying data distribution:

(A0). The dependence of the joint distribution to the parameter $\boldsymbol{\theta}$ comes only from the class-conditional distribution and the marginal distribution does not depend on $\boldsymbol{\theta}$, that is,

$$p(\mathbf{x}, y|\,\boldsymbol{\theta}) \;=\; p(y|\,\mathbf{x}, \boldsymbol{\theta}) p(\mathbf{x}). \tag{1}$$

Zhang and Oles (2000) referred to joint distributions with such parameter dependence as type-II models, as opposed to type-I models which have parameter dependence in both class conditionals and marginal. They argue that active learning is more suitable for type-II models. Moreover, maximizing the joint with respect to the parameter vector in this model, becomes equivalent to maximizing the posterior $p(y|\,\mathbf{x}, \boldsymbol{\theta})$ (inference step in sub-problem ii).

(A1). *(Identifiability):* The joint distribution $P_{\boldsymbol{\theta}}$, whose density is given by $p(\mathbf{x}, y|\,\boldsymbol{\theta})$, is identifiable for different parameters. Meaning that for every distinct parameter vectors $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ in Ω, $P_{\boldsymbol{\theta}_1}$ and $P_{\boldsymbol{\theta}_2}$ are also distinct. That is,

$$\forall\, \boldsymbol{\theta}_1 \neq \boldsymbol{\theta}_2 \in \Omega \;\; \exists A \subseteq X \times \{1, ..., c\} \quad \text{s.t.} \quad P_{\boldsymbol{\theta}_1}(A) \neq P_{\boldsymbol{\theta}_2}(A).$$

(A2). The joint distribution $P_{\boldsymbol{\theta}}$ has common support for all $\boldsymbol{\theta} \in \Omega$.

(A3). *(Model Faithfulness):* For any $\mathbf{x} \in X$, we have access to an oracle that generates a label y according to the conditional $p(y|\,\mathbf{x}, \boldsymbol{\theta}_0)$. That is, the posterior parametric model matches the oracle distribution. We call $\boldsymbol{\theta}_0$ the true model parameter.

(A4). *(Training joint):* The set of observations in $\mathcal{L}_n := \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$ are drawn independently from the *training/proposal/query* joint distribution of the form $p(y|\,\mathbf{x}, \boldsymbol{\theta}_0)$ $q(\mathbf{x})$, where q is the training marginal with no dependence on the parameter.

(A5). *(Test joint):* The unseen test pairs are distributed according to the *test/true* joint distribution of the form $p(y|\,\mathbf{x}, \boldsymbol{\theta}_0)p(\mathbf{x})$ where p is the test marginal with no dependence on the parameter.

(A6). *(Differentiability):* The log-conditional $\log p(y|\,\mathbf{x}, \boldsymbol{\theta})$ is of class $\mathcal{C}^3(\Omega)$ for all $(\mathbf{x}, y) \in X \times \{1, ..., c\}$, when being viewed as a funtion of the parameter.[3]

(A7). The parameter space Ω is compact and there exists an open ball around the true parameter of the model $\boldsymbol{\theta}_0 \in \Omega$.

(A8). *(Invertibility):* The Fisher information matrix (reviewed in Section 3.2) of the joint distribution is positive definite and therefore invertible for all $\boldsymbol{\theta} \in \Omega$, and for any type of marginal that is used under assumption (A0).

Regarding assumptions (A4) and (A5), note that the training and test marginals are not necessarily equal. The test marginal is usually not known beforehand and q cannot be set equal to p in practice, hence q can be viewed as a proposal distribution. Such inconsistency is what Shimodaira (2000) called *covariate shift in distribution*. In the remaining sections of the report, we use subscripts p and q for the statistical operators that consider $p(\mathbf{x})$ and $q(\mathbf{x})$ as the marginal in the joint distribution, respectively. We explicitly mention $\mathbf{x}$ as the input argument in order to refer to marginal operators. For instance, $\mathbb{E}_q$ denotes the joint expectation with respect to $q(\mathbf{x})p(y|\,\boldsymbol{\theta}, \mathbf{x})$, whereas $\mathbb{E}_{q(\mathbf{x})}$ denotes the marginal expectation with respect to $q(\mathbf{x})$.

3. Background

Here, we provide a short review of maximum likelihood estimation (MLE) as our inference method, and briefly introduce Fisher information of a parametric distribution. These two basic concepts enable us to explain some of the key properties of MLE, upon which our further analysis of FIR objective relies. Note that our focus in this section is on subproblem (ii) with the assumptions listed above.

3.1 Maximum Likelihood Estimation

Here, we review maximum likelihood estimation in the context of classification problem. Given a training data set $\mathcal{L}_n = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\}$, a maximum likelihood estimate (MLE) is obtained by maximizing the log-likelihood function over all pairs inside $\mathcal{L}_n$, with respect to the parameter $\boldsymbol{\theta}$:

$$\hat{\boldsymbol{\theta}}_n \;=\; \arg\max_{\boldsymbol{\theta}} \;\; \log p\left(\mathcal{L}_n \,|\, \boldsymbol{\theta}\right). \tag{2}$$

3. We say that a function $f : X \to Y$ is of $\mathcal{C}^p(X)$, for an integer $p > 0$, if its derivatives up to the p-th order exist and are continuous at all points of X.

Under the assumptions (A0) and (A4), the optimization in (2) can be written as

$$\hat{\boldsymbol{\theta}}_n \;=\; \arg\max_{\boldsymbol{\theta}} \;\sum_{i=1}^{n} \log p(y_i|\,\mathbf{x}_i, \boldsymbol{\theta}). \tag{3}$$

Equation (3) shows that MLE does not depend on the marginal when using type-II model. Hence, in our analysis we focus on the conditional log-likelihood as the classification objective, and simply call it the log-likelihood function when viewed as a function of the parameter vector $\boldsymbol{\theta}$, for any given pair $(\mathbf{x}, y) \in X \times \{1, ..., c\}$:

$$\ell(\boldsymbol{\theta}; \mathbf{x}, y) \;:=\; \log p(y|\,\mathbf{x}, \boldsymbol{\theta}). \tag{4}$$

Moreover, for any set of pairs independently generated from the joint distribution of the training data, such as $\mathcal{L}_n$ mentioned in (A4), the log-likelihood function will be

$$\ell(\boldsymbol{\theta}; \mathcal{L}_n) \;=\; \sum_{i=1}^{n} \ell(\boldsymbol{\theta}; \mathbf{x}_i, y_i) \;=\; \sum_{i=1}^{n} \log p(y_i|\,\mathbf{x}_i, \boldsymbol{\theta}). \tag{5}$$

Hence, the MLE can be rewritten as

$$\hat{\boldsymbol{\theta}}_n \;=\; \arg\max_{\boldsymbol{\theta}} \;\sum_{i=1}^{n} \ell(\boldsymbol{\theta}; \mathbf{x}_i, y_i). \tag{6}$$

Doing this maximization usually involves the computation of the stationary points of the log-likelihood, which requires calculating $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathcal{L}_n) = \sum_{i=1}^{n} \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathbf{x}_i, y_i)$. For models assumed in (A0), each of the derivations in the summation is equal to the *score function* defined as the gradient of the joint log-likelihood:

$$\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathbf{x}, y) \;=\; \nabla_{\boldsymbol{\theta}} \log p(y|\,\mathbf{x}, \boldsymbol{\theta}) \;=\; \nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}, y|\,\boldsymbol{\theta}). \tag{7}$$

Equation (7) implies that the score will be the same no matter whether we choose the training or test distribution as our marginal. Furthermore, under regularity conditions (A6), the score is always a zero-mean random variable.[4]

Finally, using MLE to estimate $\hat{\boldsymbol{\theta}}_n$, class label of a test sample $\mathbf{x}$ will be predicted as the class with the highest log-likelihood value:

$$\hat{y}(\mathbf{x}) \;=\; \arg\max_{y} \ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y). \tag{8}$$

3.2 Fisher Information

In this section, we give a very short introduction to Fisher information. More detailed descriptions about this well-known criterion can be found in various textbooks, such as Lehmann and Casella (1998).

Fisher information of a parametric distribution is a measure of information that the samples generated from that distribution provide regarding the parameter. It owes part of its importance to the Cramér-Rao Theorem (see Appendix A.2, Theorem 19), which guarantees a lower-bound for the covariance of the parameter estimators.

Fisher information, denoted by $\mathbf{I}(\boldsymbol{\theta})$, is defined as the expected value of the outer-product of the score function with itself, evaluated at some $\boldsymbol{\theta} \in \Omega$. In our classification context, taking the expectation with respect to the training or test distributions gives us the training or test Fisher information criteria, respectively:

$$\begin{aligned} \mathbf{I}_q(\boldsymbol{\theta}) \;&:=\; \mathbb{E}_q\left[\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}, y|\,\boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}}^{\top} \log p(\mathbf{x}, y|\,\boldsymbol{\theta})\right], \\ \mathbf{I}_p(\boldsymbol{\theta}) \;&:=\; \mathbb{E}_p\left[\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x}, y|\,\boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}}^{\top} \log p(\mathbf{x}, y|\,\boldsymbol{\theta})\right]. \end{aligned} \tag{9}$$

Here, we focus on $\mathbf{I}_q$ to further explain Fisher information criterion. Our descriptions can be directly generalized to $\mathbf{I}_p$ as well. First, note that from equation (7) and that the score function is always zero-mean, one can reformulate the definition as

$$\begin{aligned} \mathbf{I}_q(\boldsymbol{\theta}) \;&=\; \mathbb{E}_q\left[\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathbf{x}, y) \cdot \nabla_{\boldsymbol{\theta}}^{\top} \ell(\boldsymbol{\theta}; \mathbf{x}, y)\right] \\ &=\; \mathrm{Cov}_q\left[\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathbf{x}, y)\right]. \end{aligned} \tag{10}$$

Under the differentiability conditions (A6), it is easy to show that we can also write the Fisher information in terms of the Hessian matrix of the log-likelihood:

$$\mathbf{I}_q(\boldsymbol{\theta}) \;=\; -\mathbb{E}_q\left[\nabla_{\boldsymbol{\theta}}^2 \ell(\boldsymbol{\theta}; \mathbf{x}, y)\right]. \tag{11}$$

Recall that the subscript q in equations (10) and (11) indicates that the expectations are taken with respect to the joint distribution that uses $q(\mathbf{x})$ as the marginal, that is $p(\mathbf{x}, y|\,\boldsymbol{\theta}) = q(\mathbf{x}) p(y|\,\mathbf{x}, \boldsymbol{\theta})$. Expansion of the expectation in (11) results

$$\begin{aligned} \mathbf{I}_q(\boldsymbol{\theta}) \;&=\; -\mathbb{E}_{q(\mathbf{x})}\left[\mathbb{E}_{y|\,\mathbf{x},\boldsymbol{\theta}}\left[\nabla_{\boldsymbol{\theta}}^2 \ell(\boldsymbol{\theta}; \mathbf{x}, y)|\,\mathbf{x}, \boldsymbol{\theta}\right]\right] \\ &=\; -\int_{\mathbf{x} \in X} q(\mathbf{x}) \left[\sum_{y=1}^{c} p(y|\,\mathbf{x}, \boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}}^2 \ell(\boldsymbol{\theta}; \mathbf{x}, y)\right] d\mathbf{x}\,. \end{aligned} \tag{12}$$

3.3 Some Properties of MLE

In this section, we formalize some of the key properties of MLE, which make this estimator popular in various fields. They are also very useful in the theoretical analysis of FIR, provided in the next section. More detailed descriptions of these properties, together with the proofs that are skipped here, can be found in different sources, such as Wasserman (2004) and Lehmann and Casella (1998).

Note that a full understanding of the properties described in this section requires the knowledge of different modes of statistical convergence, specifically, convergence in probability ($\xrightarrow{P}$), and convergence in law ($\xrightarrow{L}$). A brief overview of these concepts are given in Appendix A.

Theorem 1 (Lehmann and Casella (1998), Theorem 5.1) *If the assumptions (A0) to (A7) hold, then there exists a sequence of solutions* $\left\{\hat{\boldsymbol{\theta}}_n^*\right\}_{n=1}^{\infty}$ *to* $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathcal{L}_n) = 0$ *that converges to the true parameter* $\boldsymbol{\theta}_0$ *in probability.*

4. Score function is actually zero-mean even under weaker regularity conditions.

Note that Theorem 1 does not imply that convergence holds for *any* sequence of MLEs. Hence, if there are multiple solutions to equation $\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta};\mathcal{L}_n)=0$ (the equation to solve for finding the stationary points) for every n, it is not obvious which root to select as $\hat{\boldsymbol{\theta}}_n^*$ to sustain the convergence. Therefore, while consistency of the MLE is guaranteed for models with a unique root of the score function evaluated at $\mathcal{L}_n$, it is not trivial how to build a consistent sequence when multiple roots exist. Here, in order to remove this ambiguity, we assume that either the roots are unique, or become asymptotically unique, or we have access to an external procedure guiding us to select the proper roots so that $\hat{\boldsymbol{\theta}}_n \xrightarrow{P} \boldsymbol{\theta}_0$. We will denote the selected roots the same as $\hat{\boldsymbol{\theta}}_n$ from now on.

Theorem 2 (Lehmann and Casella 1998, Theorem 5.1) *Let $\hat{\boldsymbol{\theta}}_n$ be the maximum likelihood estimator based on the training data set $\mathcal{L}_n$. If the assumptions (A0) to (A8) hold, then the MLE $\hat{\boldsymbol{\theta}}_n$ has a zero-mean normal asymptotic distribution with the covariance equal to the inverse Fisher information matrix, and with the convergence rate of $1/2$:*

$$\sqrt{n}(\hat{\boldsymbol{\theta}}_n-\boldsymbol{\theta}_0) \;\xrightarrow{L}\; \mathcal{N}\left(\mathbf{0},\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}\right). \tag{13}$$

Theorems 2 and Cramér-Rao bound (see Appendix A), together with the consistency assumption, i.e., $\hat{\boldsymbol{\theta}}_n \xrightarrow{P} \boldsymbol{\theta}_0$, imply that MLE is an asymptotically efficient estimator with the efficiency equal to the training Fisher information. One can rewrite (13) as

$$\sqrt{n}\cdot\mathbf{I}_q(\boldsymbol{\theta}_0)^{1/2}(\hat{\boldsymbol{\theta}}_n-\boldsymbol{\theta}_0) \;\xrightarrow{L}\; \mathcal{N}(\mathbf{0},\mathbb{I}_d). \tag{14}$$

In the following corollary, we see that if we substitute $\mathbf{I}_q(\boldsymbol{\theta}_0)$ with $\mathbf{I}_q(\hat{\boldsymbol{\theta}}_n)$, the new sequence still converges to a normal distribution:

Corollary 3 (Wasserman 2004, Theorem 9.18) *Under the assumptions given in Theorem 2, we get*

$$\sqrt{n}\cdot\mathbf{I}_q(\hat{\boldsymbol{\theta}}_n)^{1/2}(\hat{\boldsymbol{\theta}}_n-\boldsymbol{\theta}_0) \;\xrightarrow{L}\; \mathcal{N}\left(\mathbf{0},\mathbb{I}_d\right). \tag{15}$$

4. Fisher Information Ratio as an Upper Bound

In this section, we give our main theoretical analysis to relate FIR to the asymptotic distribution of the parameter log-likelihood ratio. Using the established relationship, we then show that FIR can be viewed as an asymptotic upper-bound of the expected variance of the loss function.

4.1 Asymptotic Distribution of MLE-Based Classifier

Recall that the estimated parameter $\hat{\boldsymbol{\theta}}_n$ is obtained from a given proposal distribution $q(\mathbf{x})$. The log-likelihood ratio function, at a given pair $(\mathbf{x},y)$, is defined as

$$\ell(\hat{\boldsymbol{\theta}}_n;\mathbf{x},y) \;-\; \ell(\boldsymbol{\theta}_0;\mathbf{x},y). \tag{16}$$

This ratio can be viewed as an example of the classification loss function whose expectation with respect to the test joint distribution of $\mathbf{x}$ and y, results in the *discrepancy* between the

true conditional $p(y|\mathbf{x},\boldsymbol{\theta}_0)$ and MLE conditional $p(y|\mathbf{x},\hat{\boldsymbol{\theta}}_n)$ (Murata et al., 1994). Here, we analyze this measure asymptotically as $(n\to\infty)$. Primarily, note that based on continuity of the log-likelihood function (A6) and consistency of MLE (Theorem 1), equation (16) converges in probability to zero for any $(\mathbf{x},y)$.

Furthermore, equation (16) is dependent on both the true marginal $p(\mathbf{x})$ (through the test pairs, where it should be evaluated) and the proposal marginal $q(\mathbf{x})$ (through the MLE $\hat{\boldsymbol{\theta}}_n$). In the classification context, Zhang and Oles (2000) claimed that the expected value of this ratio with respect to both marginals converges to $\mathrm{tr}[\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}\mathbf{I}_p(\boldsymbol{\theta}_0)]$ with the convergence rate equal to unity. In the scalar case, $\mathrm{tr}[\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}\mathbf{I}_p(\boldsymbol{\theta}_0)]$ is equal to the ratio of the Fisher information of the true and proposal distributions, the reason why it is sometimes referred to as the *Fisher information ratio* (Settles and Craven, 2008). This objective has been widely studied in linear and non-linear regression problems (Fedorov, 1972; MacKay, 1992; Murata et al., 1994; Cohn, 1996; Fukumizu, 2000). However, it is not as fully analyzed in classification.

Zhang and Oles (2000) and many papers following them (Hoi et al., 2006; Settles and Craven, 2008; Hoi et al., 2009), used this function as an *asymptotic* objective in active learning to be optimized with respect to the proposal q. Here, we show that this objective can also be viewed as an *upper bound* for the expected variance of the asymptotic distribution of (16).

First, we investigate the asymptotic distribution of the log-likelihood ratio in two different cases:

Theorem 4 *If the assumptions (A0) to (A8) hold, then, at any given $(\mathbf{x},y)\in X\times\{1,...,c\}$:*

(I) In case $\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}_0;\mathbf{x},y)\neq\mathbf{0}$, the log-likelihood ratio follows an asymptotic normality with convergence rate equal to $1/2$. More specifically,

$$\sqrt{n}\cdot\Big(\ell(\hat{\boldsymbol{\theta}}_n;\mathbf{x},y)-\ell(\boldsymbol{\theta}_0;\mathbf{x},y)\Big) \;\xrightarrow{L}\; \mathcal{N}\Big(0,\mathrm{tr}\big[\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}_0;\mathbf{x},y)\cdot\nabla_{\boldsymbol{\theta}}^{\top}\ell(\boldsymbol{\theta}_0;\mathbf{x},y)\cdot\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}\big]\Big). \tag{17}$$

(II) In case $\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}_0;\mathbf{x},y)=\mathbf{0}$ and $\nabla^2_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}_0;\mathbf{x},y)$ is non-singular, the asymptotic distribution of the log-likelihood ratio is a mixture of Chi-square distributions with one degree of freedom, and the convergence rate is one. More specifically,

$$n\cdot\Big(\ell(\hat{\boldsymbol{\theta}}_n;\mathbf{x},y) \;-\; \ell(\boldsymbol{\theta}_0;\mathbf{x},y)\Big) \;\xrightarrow{L}\; \sum_{i=1}^{d}\lambda_i\cdot\chi^2_1, \tag{18}$$

where λ_i's are eigenvalues of $\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1/2}\nabla^2_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}_0;\mathbf{x},y)\,\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1/2}$.

Proof Due to assumptions (A0) to (A7), Theorem 2 holds and therefore we have $\sqrt{n}\cdot(\hat{\boldsymbol{\theta}}_n-\boldsymbol{\theta}_0)\xrightarrow{L}\mathcal{N}(\mathbf{0},\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1})$. The rest of the proof is based on the Delta method in the two modes described in Appendix A (Theorems 16 and 17):

(I) $\nabla_{\boldsymbol{\theta}}\ell(\boldsymbol{\theta}_0;\mathbf{x},y)\neq\mathbf{0}$:

Since the expected log-likelihood function, evaluated at a given pair $(\mathbf{x}, y)$, is assumed to be continuously differentiable (A6) and that $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \neq \mathbf{0}$, we can apply Theorem 16 to $\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)$ to write:

$$\sqrt{n} \cdot \Big(\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) \;-\; \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \Big) \;\xrightarrow{L}\; \mathcal{N}\Big(0 \;,\; \nabla_{\boldsymbol{\theta}}^\top \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \cdot \mathbf{I}_q(\boldsymbol{\theta}_0)^{-1} \cdot \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \Big), \tag{19}$$

where the scalar variance can also be written in a trace format.

(II) $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) = \mathbf{0}$ and $\nabla^2_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)$ non-singular :

In this case, the conditions in Theorem 17 are satisfied (with $\boldsymbol{\Sigma} = \mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}$ and $g(\boldsymbol{\theta}) = \ell(\boldsymbol{\theta}; \mathbf{x}, y)$), and therefore we can directly write (18) from equations (51).

■

Theorem 4 regards the log-likelihood ratio (16) evaluated at any arbitrary pair $(\mathbf{x}, y)$. Note that if we consider the training pairs in $\mathcal{L}_n$, which are used to obtain $\hat{\boldsymbol{\theta}}_n$, it is known that the ratio evaluated at the training set converges to a single Chi-square distribution with degree one, that is,

$$\ell(\hat{\boldsymbol{\theta}}_n; \mathcal{L}_n) \;-\; \ell(\boldsymbol{\theta}_0; \mathcal{L}_n) \;\xrightarrow{L}\; \frac{1}{2}\chi^2_1. \tag{20}$$

Theorem 4 implies that variance of the asymptotic distribution of the log-likelihood ratio in case (I) is $\operatorname{tr}\big[\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \cdot \nabla_{\boldsymbol{\theta}}^\top \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \cdot \mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}\big]$, whereas in case (II), from Theorem 17 (see Appendix A), the variance is $\frac{1}{2} \left\| \mathbf{I}_q(\boldsymbol{\theta}_0)^{-1/2} \nabla^2_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \, \mathbf{I}_q(\boldsymbol{\theta}_0)^{-1/2} \right\|_F^2$. Therefore, it is evident that the variance of the log-likelihood ratio at any $(\mathbf{x}, y)$ is reciprocally dependent on the training Fisher information. From this point of view, one can set the training distribution such that it leads to a Fisher information that minimizes this variance. Unless the parameter and hence the Fisher information is univariate, it is not clear what objective to optimize with respect to q such that the resulting Fisher information minimizes the variance.

4.2 Establishing the Upper Bound

In the next theorem, we show that the Fisher information ratio , $\operatorname{tr}\big[\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1} \mathbf{I}_p(\boldsymbol{\theta}_0)\big]$, is a reasonable candidate objective to minimize in order to get a training distribution q for the multivariate case.

Theorem 5 *If the assumptions (A0) to (A8) hold, then*

$$\mathbb{E}_p \Big[\operatorname{Var}_q \Big(\lim_{n\to\infty} \sqrt{n} \cdot [\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)] \Big) \Big] \;\leq\; \operatorname{tr}\Big[\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1} \mathbf{I}_p(\boldsymbol{\theta}_0) \Big]. \tag{21}$$

The equality holds if the set of pairs $(\mathbf{x}, y)$ where we have zero score function at $\boldsymbol{\theta}_0$, i.e., $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) = \mathbf{0}$, has measure zero under the true joint distribution $P_{\boldsymbol{\theta}_0}$ in $X \times \{1, ..., c\}$.

Proof First note that $\operatorname{Var}_q \Big(\lim_{n\to\infty} \sqrt{n} \cdot [\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)] \Big)$ is well-defined for any given pair $(\mathbf{x}, y)$. We consider the two cases where the score $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)$ is non-zero and zero. In the former case, Theorem 4 shows that the sequence $\sqrt{n} \cdot [\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)]$

is asymptotically distributed according to a zero-mean normal distribution with variance shown in (17). On the other hand, when the score is zero, the rate of convergence of log-likelihood ratio is shown to be one. More specifically, Theorem 4 derives the asymptotic (non-degenerate) distribution of the sequence $n \cdot [\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)]$. Based on Proposition 26, any sequence of random variables converging in law to a non-degenerate distribution is of $O_p(1)$, that is, $n \cdot [\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)] = O_p(1)$ and therefore, $\sqrt{n} \cdot [\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)] = O_p\left(\frac{1}{\sqrt{n}}\right)$. Now, from Proposition 25, one can conclude that the sequence $\sqrt{n} \cdot [\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)]$ is of $o_p(1)$ and, by definition, converges in probability to zero. In other words, in case of zero score, the sequence $\sqrt{n} \cdot [\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)]$ converges in law to a degenerate distribution concentrated at zero and hence has zero asymptotic variance.

According to the analysis above, we can compute the expected value of the asymptotic variance by only considering regions with non-zero score. Define the region $R_0 \subseteq X \times \{1, ..., c\}$ as

$$R_0 \;:=\; \{(\mathbf{x}, y) | \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) = 0\}. \tag{22}$$

By this definition, the case of having zero score, $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) = 0$, happens with probability $P_{\boldsymbol{\theta}_0}(R_0)$, and non-zero score, $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \neq 0$, happens with probability $1 - P_{\boldsymbol{\theta}_0}(R_0)$. Considering these two cases and the analysis made above, variance of the asymptotic distribution of $\sqrt{n} \cdot [\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)]$ can be written as

$$\begin{aligned} &\operatorname{Var}\Big(\lim_{n\to\infty} \sqrt{n} \cdot [\ell(\hat{\boldsymbol{\theta}}_n; \mathbf{x}, y) - \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)] \Big) \\ &\quad= [1 - P_{\boldsymbol{\theta}_0}(R_0)] \cdot \operatorname{tr}\big[\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \cdot \nabla^\top_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \cdot \mathbf{I}_q(\boldsymbol{\theta}_0)^{-1} \big] \;+\; P_{\boldsymbol{\theta}_0}(R_0) \cdot 0 \\ &\quad\leq\; \operatorname{tr}\big[\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \cdot \nabla^\top_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) \cdot \mathbf{I}_q(\boldsymbol{\theta}_0)^{-1} \big]. \end{aligned} \tag{23}$$

Taking the expectation of both sides with respect to the true joint gives the inequality (21). If the set of pairs $(\mathbf{x}, y)$ where $\nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) = \mathbf{0}$ form a zero measure set under $P_{\boldsymbol{\theta}_0}$, then $P_{\boldsymbol{\theta}_0}(R_0) = 0$ and we get equality in (23), hence an equality in (21). ■

Theorem 5 implies that minimizing the Fisher information ratio with respect to q, is indeed the upper-bound minimization of the expected variance of the asymptotic distribution of the log-likelihood ratio.

5. Fisher Information Ratio in Practice

In this section, we explain how inequality (21) can be utilized in practice as an objective function for active learning. The left-hand-side is the objective that is more reasonable to minimize from classification point of view. However, its optimization is intractable and FIR-based methods approximate it by its upper-bound minimization. Querying can be done with this objective by first learning the optimal proposal distribution q that minimizes the left-hand-side of inequality (21) and then drawing the queries from this optimal distribution:

$$q^* \;=\; \arg\min_q \;\operatorname{tr}[\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1} \mathbf{I}_p(\boldsymbol{\theta}_0)] \tag{24a}$$

$$X_q \;\sim\; q^*(\mathbf{x}), \tag{24b}$$

where X_q is the set of queries whose samples are drawn from q^*. Note that in (24), due to the sampling process, X_q cannot be deterministically determined even by fixing all the other parameters leading to a fixed query distribution q^* (ignoring the uncertainties in the numerical optimization processes). Hence, this setting is sometimes called *probabilistic* active learning. Notice that in pool active learning, q should be constrained to be a PMF over the unlabeled pool from which the queries are to be chosen. Relaxing q to continuous distributions leads to *synthetic* active learning, since each time an unseen sample will be *synthesized* by sampling from q. We will see later that in some pool-based applications, the *objective functional* of q is approximated as a *set function* of X_q, and therefore a combinatorial optimization is performed directly with respect to the query set.

As mentioned, q^* is an upper-bound minimization of the expected asymptotic loss variance. Moreover, there are a number of unknown variables involved in FIR objective, such as $\mathbf{I}_p$ and $\boldsymbol{\theta}_0$. In practice, estimations of these unknown variables are used in the optimization process for active learning. Therefore, although the derivations in the previous section (Theorem 5) are made based on one querying of infinitely many samples, in active learning a finite-sample approximation of the cost function is used in an iterative querying process. As the number of querying iterations in active learning increases, the parameter estimates get more accurate and so does the approximate FIR objective. In the next section, we show that under certain assumptions the optimization with respect to proposal distribution in each iteration is yet another upper-bound minimization similar to (21). More specifically, Remark 6 (see Section 5.1.1) shows that although the proposal distribution is optimized separately in each iteration of an FIR-based active learning algorithm, minimizing the approximate FIR at each iteration is still an upper-bound minimization of the original cost function (i.e. left-hand-side of inequality 21).

Algorithm 0 shows steps of a general discriminative classification with active learning. We assume an initial training set $\mathcal{L}_{n_0} = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_{n_0}, y_{n_0})\}$ is given based on which an initial MLE $\hat{\boldsymbol{\theta}}_{n_0}$ can be obtained. The initial MLE enables us to approximate the active learning objective function and therefore select queries for building the new training set. After obtaining the query set X_q, for each individual sample $\mathbf{x} \in X_q$, we request its labels $y(\mathbf{x})$ from the oracle, or equivalently, sample it from the true conditional, $y(\mathbf{x}) \sim p(y|\mathbf{x}, \boldsymbol{\theta}_0)$. These pairs are then added into the training set to get $\mathcal{L}_{n_1}$, which in turn, is used to update the parameter estimate to $\hat{\boldsymbol{\theta}}_{n_1}$. Size of the new training data is $n_1 = n_0 + |X_q|$. This procedure can be done repeatedly for a desirable number of iterations i_{max}. All different techniques that we discuss in the following sections, differ only in line 3 and the rest of the steps are common between them. Each active learning algorithm $\mathcal{A}$ takes the current estimate of the parameter $\hat{\boldsymbol{\theta}}_{n_{i-1}}$ possibly together with the unlabeled set of samples X_p, and generate a set of queries X_q to be labeled for the next iteration.

In our analysis in the subsequent sections, we focus on a specific *querying iteration* indexed by i (as a positive integer). For simplicity, we replace n_{i-1} and n_i (size of the training data set before and after iteration i) by n' and n, respectively. Hence, iteration i consists of using the available parameter estimate, $\hat{\boldsymbol{\theta}}_{n'}$, obtained through the current training data set $\mathcal{L}_{n'}$, to generate queries using a given querying algorithm $\mathcal{A}(\hat{\boldsymbol{\theta}}_{n'}, \mathcal{L}_{n'})$, and then update the classifier's parameter estimate accordingly to $\hat{\boldsymbol{\theta}}_n$.

In what follows, we first discuss practical issues in using FIR in query selection (Section 5.1) and then review existing algorithms based on this objective (Section 5.2).

Algorithm 0: Classification with Active Learning

Inputs: The initial training set $\mathcal{L}_{n_0}$; number of querying iterations i_{max}
Outputs: The trained classifier with MLE $\hat{\boldsymbol{\theta}}_{n_{i_{max}}}$

/* Initializations */
1 $\hat{\boldsymbol{\theta}}_{n_0} \leftarrow \arg\max_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathcal{L}_{n_0})$
/* Starting the Iterations */
2 **for** $i = 1 \rightarrow i_{max}$ **do**
/* Generating the query set by optimizing a querying objective */
3 $X_q \leftarrow \mathcal{A}(\mathcal{L}_{n_{i-1}}, \hat{\boldsymbol{\theta}}_{n_{i-1}})$
/* Request the queries' labels from the oracle */
4 $y(\mathbf{x}) \sim p(y|\mathbf{x}, \boldsymbol{\theta}_0) \quad \forall \mathbf{x} \in X_q$
/* Taking care of indexing */
5 $n_i \leftarrow n_{i-1} + |X_q|$
/* Update the training set and update MLE */
6 $\mathcal{L}_{n_i} \leftarrow \mathcal{L}_{n_{i-1}} \cup \left\{ \bigcup_{\mathbf{x} \in X_q} (\mathbf{x}, y(\mathbf{x})) \right\}$
7 $\hat{\boldsymbol{\theta}}_{n_i} \leftarrow \arg\max_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathcal{L}_{n_i})$
8 **return** $\hat{\boldsymbol{\theta}}_{n_{i_{max}}}$

5.1 Practical Issues

The main difficulties consist of (1) having unknown variables in the objective, such as the test marginal, $p(\mathbf{x})$, and the true parameter, $\boldsymbol{\theta}_0$, and (2) lack of closed form for Fisher information matrices for most cases. In the next two sections, we review different hacks and solutions that have been proposed to resolve these issues.

5.1.1 Replacing $\boldsymbol{\theta}_0$ by $\hat{\boldsymbol{\theta}}_{n'}$

Since $\boldsymbol{\theta}_0$ is not known, the simplest idea is to replace it by the current parameter estimate, that is $\hat{\boldsymbol{\theta}}_{n'}$ (Fukumizu, 2000; Settles and Craven, 2008; Hoi et al., 2006, 2009; Chaudhuri et al., 2015b). Clearly, as the algorithm keeps running the iterations (n' increases), the approximate objective (which contains $\boldsymbol{\theta}_{n'}$ instead of $\boldsymbol{\theta}_0$) gets closer to the original objective. This is due to the regularity and invertibility conditions assumed for the log-likelihood function and Fisher information matrices, respectively. Moreover, Chaudhuri et al. (2015b) analyzed how this approximation effects the querying performance in finite-sample case.

Their analysis is done only for pool-based active learning, and when the test marginal $p(\mathbf{x})$ is a *uniform distribution* $U(\mathbf{x})$ over the pool X_p. It is also assumed that the Hessian $\frac{\partial^2 \ell(\boldsymbol{\theta}; \mathbf{x}, y)}{\partial \boldsymbol{\theta}^2}$ is independent of the class labels y, and therefore can be viewed as the conditional Fisher information $\mathbf{I}(\boldsymbol{\theta}, \mathbf{x})$, or equivalently $\mathbf{I}_p(\boldsymbol{\theta}) = \mathbb{E}_{p(\mathbf{x})}[\mathbf{I}(\boldsymbol{\theta}, \mathbf{x})]$. Furthermore, there assumed to exist four positive constants $L_1, L_2, L_3, L_4 \geq 0$ such that the following four

inequalities hold for all $\mathbf{x} \in X_p$, $y \in \{1, ..., c\}$ and $\boldsymbol{\theta} \in \Theta$:

$$\begin{aligned} \nabla \ell(\boldsymbol{\theta}_0; \mathbf{x}, y)^\top \mathbf{I}_p(\boldsymbol{\theta}_0)^{-1} \nabla \ell(\boldsymbol{\theta}_0; \mathbf{x}, y) &\leq L_1 \\ \left\| \mathbf{I}_p(\boldsymbol{\theta}_0)^{-1/2} \mathbf{I}(\boldsymbol{\theta}_0, \mathbf{x}) \mathbf{I}_p(\boldsymbol{\theta}_0)^{-1/2} \right\| &\leq L_2 \\ \left\| \mathbf{I}_p(\boldsymbol{\theta}_0)^{-1/2} (\mathbf{I}(\boldsymbol{\theta}', \mathbf{x}) - \mathbf{I}(\boldsymbol{\theta}'', \mathbf{x})) \mathbf{I}_p(\boldsymbol{\theta}_0)^{-1/2} \right\| &\leq L_3 (\boldsymbol{\theta}' - \boldsymbol{\theta}'')^\top \mathbf{I}_p(\boldsymbol{\theta}_0) (\boldsymbol{\theta}' - \boldsymbol{\theta}'') \\ -L_4 \| \boldsymbol{\theta} - \boldsymbol{\theta}_0 \|_2 \mathbf{I}(\boldsymbol{\theta}_0, \mathbf{x}) \preceq \mathbf{I}(\boldsymbol{\theta}, \mathbf{x}) - \mathbf{I}(\boldsymbol{\theta}_0, \mathbf{x}) &\preceq L_4 \| \boldsymbol{\theta} - \boldsymbol{\theta}_0 \|_2 \mathbf{I}(\boldsymbol{\theta}_0, \mathbf{x}), \end{aligned} \tag{25}$$

where $\boldsymbol{\theta}'$ and $\boldsymbol{\theta}''$ are any two parameters in a fixed neighborhood of $\boldsymbol{\theta}_0$. Then, provided that n' is large enough, the following remark can be shown regarding the relationship between FIRs computed at $\boldsymbol{\theta}_0$ and an estimate $\hat{\boldsymbol{\theta}}_{n'}$:

Remark 6 *Let the assumptions (A0) to (A8) and those in (25) hold. Moreover, assume that the Hessian is independent of the class labels. If n' is large enough, then the following inequality holds for any $\beta \geq 10$ with high probability:*

$$\operatorname{tr}\left[\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1} \mathbf{I}_p(\boldsymbol{\theta}_0)\right] \leq \frac{\beta+1}{\beta-1} \cdot \operatorname{tr}\left[\mathbf{I}_q(\hat{\boldsymbol{\theta}}_{n'})^{-1} \mathbf{I}_p(\hat{\boldsymbol{\theta}}_{n'})\right]. \tag{26}$$

The minimum value for n' that is necessary for having this inequality with probability $1-\delta$, increases quadratically with β and reciprocally with δ (Chaudhuri et al., 2015b, Lemma 2).

Proof It is shown in the proof of Lemma 2 in Chaudhuri et al. (2015a) that under assumptions mentioned in the statement, the following inequalities hold with probability $1-\delta$:

$$\frac{\beta-1}{\beta} \mathbf{I}(\mathbf{x}, \boldsymbol{\theta}_0) \preceq \mathbf{I}(\mathbf{x}, \hat{\boldsymbol{\theta}}_{n'}) \preceq \frac{\beta+1}{\beta} \mathbf{I}(\mathbf{x}, \boldsymbol{\theta}_0). \tag{27}$$

Taking expectation with respect to $p(\mathbf{x})$ and $q(\mathbf{x})$ results

$$\frac{\beta-1}{\beta} \mathbf{I}_p(\boldsymbol{\theta}_0) \preceq \mathbf{I}_p(\hat{\boldsymbol{\theta}}_{n'}) \preceq \frac{\beta+1}{\beta} \mathbf{I}_p(\boldsymbol{\theta}_0), \tag{28a}$$

$$\frac{\beta-1}{\beta} \mathbf{I}_q(\boldsymbol{\theta}_0) \preceq \mathbf{I}_q(\hat{\boldsymbol{\theta}}_{n'}) \preceq \frac{\beta+1}{\beta} \mathbf{I}_q(\boldsymbol{\theta}_0). \tag{28b}$$

Since $\mathbf{I}_q(\boldsymbol{\theta}_0)$ and $\mathbf{I}_q(\hat{\boldsymbol{\theta}}_{n'})$ are assumed to be positive definite, we can write (28b) in terms of inverted matrices:[5]

$$\frac{\beta}{\beta+1} \mathbf{I}_q^{-1}(\boldsymbol{\theta}_0) \preceq \mathbf{I}_q^{-1}(\hat{\boldsymbol{\theta}}_{n'}) \preceq \frac{\beta}{\beta-1} \mathbf{I}_q^{-1}(\boldsymbol{\theta}_0). \tag{29}$$

Now considering the first inequalities of (28a) and (29), multiplying both sides and taking the trace result (26). ■

Inequality (26) implies that minimizing $\operatorname{tr}\left[\mathbf{I}_q(\hat{\boldsymbol{\theta}}_{n'})^{-1} \mathbf{I}_p(\hat{\boldsymbol{\theta}}_{n'})\right]$ (or an approximation of it) with respect to q in each iteration of FIR-based querying algorithms, namely through the operation $\mathcal{A}(\mathcal{L}_{n'}, \hat{\boldsymbol{\theta}}_{n'})$ (line 3 of Algorithm 0), is equivalent to upper bound minimization of the original cost function, i.e. left-hand-side of (21).

5. For any two positive definite matrices $\mathbf{A}$ and $\mathbf{B}$, we have that $\mathbf{A} \succeq \mathbf{B} \Rightarrow \mathbf{A}^{-1} \preceq \mathbf{B}^{-1}$.

5.1.2 Monte-Carlo Approximation

Computation of Fisher information matrices is intractable unless when the marginal distributions are very simple or when they are restricted to be PMFs over finite number of samples. The latter is widely used in pool-based active learning, when the samples in the pool are assumed to be generated from $p(\mathbf{x})$. In such cases, one can simply utilize a Monte-Carlo approximation to compute $\mathbf{I}_p$. More specifically, denote the set of observed instances in the pool by X_p. Then the test Fisher information at any $\boldsymbol{\theta} \in \Omega$ can be approximated by

$$\mathbf{I}_p(\boldsymbol{\theta}) \approx \hat{\mathbf{I}}(\boldsymbol{\theta}; X_p) := \frac{1}{|X_p|} \sum_{\mathbf{x} \in X_p} \sum_{y=1}^{c} p(y|\mathbf{x}, \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}; \mathbf{x}, y) \nabla_{\boldsymbol{\theta}}^\top \ell(\boldsymbol{\theta}; \mathbf{x}, y) + \delta \cdot \mathbb{I}_d, \tag{30}$$

where δ is a small positive number and the weighted identity matrix is added to ensure positive definiteness. It is important to give the practical remark that when using equation (30), we are actually using some of the test samples in the training process, hence we cannot use those in X_p in order to evaluate the performance of the trained classifier.

Similarly, $\mathbf{I}_q$ can be estimated based on a candidate query set X_q. Let X_q be the set of samples drawn independently from $q(\mathbf{x})$. Then, for any $\boldsymbol{\theta} \in \Theta$, we can have the approximation $\mathbf{I}_q(\boldsymbol{\theta}) \approx \hat{\mathbf{I}}(\boldsymbol{\theta}; X_q)$. Putting everything together, the best query set $X_q \subseteq X_p$ is chosen to be the one that minimizes the approximate FIR querying objective,

$$\operatorname{tr}\left[\hat{\mathbf{I}}(\hat{\boldsymbol{\theta}}_{n'}; X_q)^{-1} \hat{\mathbf{I}}(\hat{\boldsymbol{\theta}}_{n'}; X_p)\right]. \tag{31}$$

Note that this objective is directly written in terms of X_q, and therefore the queries can be deterministically determined by fixing all the rest (including the current parameter estimate $\hat{\boldsymbol{\theta}}_{n'}$) and optimizing with respect to X_q. Therefore, such settings are usually called *deterministic* active learning, as opposed to the probabilistic nature of (24).

5.1.3 Bound Optimization

There are other types of approximation methods occurring in the optimization side. These methods are able to remove part of the unknown variables by doing upper-bound minimization or lower-bound maximization. Recall that in active learning, the querying objective is to be optimized with respect to q (or X_q in pool-based scenario). In a very simple example, when $d = 1$, note that the $\mathbf{I}_p(\boldsymbol{\theta}_0)$ is a constant scalar in (24a) and hence can be ignored. Hence, in the scalar case, we can simply focus on maximizing the training Fisher information. In the multivariate case, though it is not clear what measure of $\mathbf{I}_q(\hat{\boldsymbol{\theta}}_n)$ to optimize, one may choose the objective to be $|\mathbf{I}_q(\hat{\boldsymbol{\theta}}_n)|$ (where $|\cdot|$ is the determinant function),[6] or $\operatorname{tr}[\mathbf{I}_q(\hat{\boldsymbol{\theta}}_n)]$.[7] The latter is worth paying more attention due to the following inequality (Yang, 2000):

$$\operatorname{tr}[\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1} \mathbf{I}_p(\boldsymbol{\theta}_0)] \leq \operatorname{tr}[\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}] \cdot \operatorname{tr}[\mathbf{I}_p(\boldsymbol{\theta}_0)]. \tag{32}$$

Since $\operatorname{tr}[\mathbf{I}_p(\boldsymbol{\theta}_0)]$ is a constant with respect to q, minimizing the right-hand-side of inequality (21) can itself be approximated by another upper-bound minimization:

$$\arg\min_q \ \operatorname{tr}[\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}]. \tag{33}$$

6. Similar to *D-optimality* in Optimal Experiment Design (Fedorov, 1972).
7. Similar to *A-optimality* in Optimal Experiment Design (Fedorov, 1972).

	Algorithm	Obj.	Prob.	Det.	Pool	Syn.	Seq.	Batch
1	Fukumizu (2000)	(34)	✓			✓	✓	✓
2	Zhang and Oles (2000)	(34)		✓	✓	✓	✓	
3	Settles and Craven (2008)	(31)		✓	✓		✓	✓
4	Hoi et al. (2006, 2009)	(31)		✓	✓		✓	✓
5	Chaudhuri et al. (2015b)	(26)	✓		✓		✓	✓

Table 1: Reviewed FIR-based active learning algorithms for discriminative classifiers

This helps removing the dependence of the objective to the test distribution p. A lower bound can also be established for the FIR. Using the inequality between arithmetic and geometric means of the eigenvalues of $\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}\mathbf{I}_p(\boldsymbol{\theta}_0)$, one can see that $d \cdot |\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}| \cdot |\mathbf{I}_p(\boldsymbol{\theta}_0)| \leq \mathrm{tr}[\mathbf{I}_q(\boldsymbol{\theta}_0)^{-1}\mathbf{I}_p(\boldsymbol{\theta}_0)]$. Hence, when minimizing the upper-bound in (33), one should be careful about the determinant of this matrix as a term influencing the lower-bound of the objective.

In practice, of course, the minimization in (33) can be difficult due to matrix inversion. Thus, sometimes it is further approximated by

$$\arg\max_q \ \mathrm{tr}[\mathbf{I}_q(\boldsymbol{\theta}_0)]. \tag{34}$$

Hence, algorithms that aim to maximize $\mathrm{tr}[\mathbf{I}_q(\boldsymbol{\theta}_0)]$, indeed introduce two layers of objective approximations through equations (32) to (34). As discussed before, the dependence of the objectives in all the layers (in 33 or 34) on $\boldsymbol{\theta}_0$ can be removed by replacing it with the current estimate $\boldsymbol{\theta}_{n'}$.

5.2 Some Existing Algorithms

In this section, we discuss several existing algorithms for implementing the query selection task based on minimization of FIR. We will analyze these algorithms, sorted according to date of their publication, in the context of our unifying framework.

Besides the categorizations that have already been described in previous sections, it is also useful to divide the querying algorithms into two categories based on the size of X_q: *sequential active learning*, where a single sample is queried at each iteration, i.e., $|X_q| = 1$; and *batch active learning*, where the size of the query set is larger than one. The non-singleton query batches are usually generated greedily, with the batch size $|X_q|$ fixed to a constant value.

Table 1 lists the algorithms that we reviewed in the following sections together with a summary of their properties and the approximate objective that they optimize for querying. Note that among these algorithms, the one by Chaudhuri et al. (2015b) makes extra assumptions as is described in Section 5.1.1.

Algorithm 1 (Fukumizu, 2000)

This algorithm is the classification version of the *probabilistic active learning* proposed by Fukumizu (2000) for regression problem. The assumption is that the proposal belongs to a parametric family and is of the form $q(\mathbf{x};\boldsymbol{\alpha})$, where $\boldsymbol{\alpha}$ is the parameter vector of the

Algorithm 1: Fukumizu (2000)

Inputs: Current estimation of the parameter $\hat{\boldsymbol{\theta}}_{n'}$, size of the query set $|X_q|$
Outputs: The query set X_q

/* Parameter Optimization */
$\hat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha}} \mathbb{E}_{q(\mathbf{x};\boldsymbol{\alpha})}\left[\sum_{y=1}^{c} p(y|\,\mathbf{x},\hat{\boldsymbol{\theta}}_{n'})\nabla_{\boldsymbol{\theta}}^{\top}\ell(\hat{\boldsymbol{\theta}}_{n'};\mathbf{x},y)\nabla_{\boldsymbol{\theta}}\ell(\hat{\boldsymbol{\theta}}_{n'};\mathbf{x},y)\right]$
/* Sampling from the parametric proposal */
$\mathbf{x}_i \sim q(\mathbf{x};\hat{\boldsymbol{\alpha}}) \quad , i = 1,...,|X_q|$
return $X_q = \{\mathbf{x}_1,...,\mathbf{x}_{|X_q|}\}$

family. In this *parametric* active learning, the best set of parameters $\hat{\boldsymbol{\alpha}}$ is selected using the current parameter estimate and the query set is sampled from the resulting proposal distribution $X_q \sim q(\mathbf{x};\hat{\boldsymbol{\alpha}})$.

This algorithm makes no use of the test samples and optimizes the simplified objective in (34) to obtain the query distribution $q(\mathbf{x})$. Denote the covariates of the current training data set $\mathcal{L}_{n'}$ by $X_{\mathcal{L}}$. As is described in Section (5.1), the new trace objective can be approximated by Monte-Carlo formulation using the old queried samples $X_{\mathcal{L}}$ as well as the candidate queries X_q to be selected in this iteration:

$$\mathrm{tr}\left[\hat{\mathbf{I}}(\hat{\boldsymbol{\theta}}_{n'};X_{\mathcal{L}}\cup X_q)\right]. \tag{35}$$

More specifically, the new parameter vector is obtained by maximizing the expected contribution of the queries X_q generated from $q(\mathbf{x};\boldsymbol{\alpha})$ to this objective. Taking expectation of (35) with respect to $q(\mathbf{x};\boldsymbol{\alpha})$ yields

$$\mathbb{E}_{q(\mathbf{x};\boldsymbol{\alpha})}\left[\mathrm{tr}\left[\hat{\mathbf{I}}(\hat{\boldsymbol{\theta}}_{n'};X_{\mathcal{L}}\cup X_q)\right]\right] = \mathrm{tr}\left[\frac{n'}{n'+|X_q|}\hat{\mathbf{I}}(\hat{\boldsymbol{\theta}}_{n'};X_{\mathcal{L}}) + \frac{1}{n'+|X_q|}\mathbb{E}_{q(\mathbf{x};\boldsymbol{\alpha})}\left[\hat{\mathbf{I}}(\hat{\boldsymbol{\theta}}_{n'};X_q)\right]\right]. \tag{36}$$

Recall that n' is the size of the current training data set $\mathcal{L}_{n'}$. The first term in (36) is independent of the query set X_q (assuming that the size $|X_q|$ is fixed to a constant), hence we focus only on the second term in our optimization. Noting that the queries are generated independently, we can rewrite this term (excluding the constant coefficient) as

$$\mathbb{E}_{q(\mathbf{x};\boldsymbol{\alpha})}\left[\mathrm{tr}\left[\hat{\mathbf{I}}(\hat{\boldsymbol{\theta}}_{n'};X_q)\right]\right] = \frac{1}{|X_q|}\mathbb{E}_{q(\mathbf{x};\boldsymbol{\alpha})}\left[\sum_{\mathbf{x}\in X_q}\mathrm{tr}\left[\hat{\mathbf{I}}(\hat{\boldsymbol{\theta}}_{n'};\mathbf{x})\right]\right] \tag{37}$$

From equation (37), the parameter vector $\boldsymbol{\alpha}$ can be obtained by maximizing the expected contribution of that single query to the trace objective. That is, having fixed $|X_q|$ to a constant, the optimization for determining the parameter vector would be

$$\begin{aligned}\hat{\boldsymbol{\alpha}} &= \arg\max_{\boldsymbol{\alpha}} \ \mathbb{E}_{q(\mathbf{x};\boldsymbol{\alpha})}\left[\mathrm{tr}\left[\hat{\mathbf{I}}(\hat{\boldsymbol{\theta}}_{n'};\mathbf{x})\right]\right] \\ &= \arg\max_{\boldsymbol{\alpha}} \ \mathbb{E}_{q(\mathbf{x};\boldsymbol{\alpha})}\left[\sum_{y=1}^{c} p(y|\,\mathbf{x},\hat{\boldsymbol{\theta}}_{n'})\nabla_{\boldsymbol{\theta}}^{\top}\ell(\hat{\boldsymbol{\theta}}_{n'};\mathbf{x},y)\nabla_{\boldsymbol{\theta}}\ell(\hat{\boldsymbol{\theta}}_{n'};\mathbf{x},y)\right].\end{aligned} \tag{38}$$

Algorithm 2: Zhang and Oles (2000)

Inputs: Current estimation of the parameter $\hat{\boldsymbol{\theta}}_{n'}$
Outputs: The query singleton set $X_q = \{\mathbf{x}_q\}$

$\mathbf{x}_q \leftarrow \arg\max_{\mathbf{x}} \sum_{y=1}^{c} p(y|\mathbf{x}, \hat{\boldsymbol{\theta}}_{n'}) \nabla_{\boldsymbol{\theta}}^{\top} \ell(\hat{\boldsymbol{\theta}}_{n'}; \mathbf{x}, y) \nabla_{\boldsymbol{\theta}} \ell(\hat{\boldsymbol{\theta}}_{n'}; \mathbf{x}, y)$

return $X_q = \{\mathbf{x}_q\}$

The optimization (38) does not depend on $X_{\mathcal{L}}$, and therefore we do not need to explicitly feed this algorithm with $\mathcal{L}$. All it needs is an estimation of the parameter $\hat{\boldsymbol{\theta}}_{n'}$. The two-step procedure of generating queries from parametric query distribution is shown in Algorithm 1. This algorithm can be used in both sequential and batch modes by changing the number of samples drawn from $q(\mathbf{x}; \alpha)$.

We emphasize that Algorithm 1 is probabilistic, meaning that with any fixed parameter estimate $\hat{\boldsymbol{\theta}}_{n'}$, the next set of queries are *not* deterministically selected. The optimization is performed with respect to the parameters of the proposal distribution, which are then used to sample X_q. Fukumizu (2000) claims that introducing such randomness into active learning, which increases exploration against exploitation, may prevent the algorithm from falling into local optima. Also note that this algorithm is not pool-based, meaning that it does not select the queries from a pool of observed instances, although could be constrained to do so.

Algorithm 2 (Zhang and Oles, 2000)

Zhang and Oles (2000) started from optimization problem (34), and introduced even additional simplifications to it. They specifically considered using a binary logistic regression classifier. Here, we discuss their formulation using a general discriminate framework.

In their algorithm, a single query is selected at each iteration. Denote it by $X_q = \{\mathbf{x}_q\}$. Similar to the previous section, the Fisher information matrix $\mathbf{I}_q$ can be approximated by Monte-Carlo approximation. Zhang and Oles (2000) discarded the expectation with respect to the proposal distribution in (38) or equivalently consider q to be a uniform distribution. Therefore, the optimization with respect to parameters turned into a direct optimization with respect to the single query $\mathbf{x}_q$:

$$\mathbf{x}_q = \arg\max_{\mathbf{x} \in X} \sum_{y=1}^{c} p(y|\mathbf{x}, \hat{\boldsymbol{\theta}}_{n'}) \nabla_{\boldsymbol{\theta}}^{\top} \ell(\hat{\boldsymbol{\theta}}_{n'}; \mathbf{x}, y) \nabla_{\boldsymbol{\theta}} \ell(\hat{\boldsymbol{\theta}}_{n'}; \mathbf{x}, y). \tag{39}$$

This single-step deterministic approach, shown in Algorithm 2, is very similar to the probabilistic approach described above, except that there is no intermediate parameter optimization step.

It is important to note that Algorithm 2 can be used in pool-based active learning as well. This can be done by constraining $\mathbf{x}_q$ to be a member of a pool of samples, in which case it can even be extended to batch querying by sorting the unlabeled samples based on their objective values and taking the highest ones. However, such iterative optimization is not efficient, because the resulting queries will most probably be close to each other and therefore contain redundant information.

Algorithm 3: Settles and Craven (2008)

Inputs: Current estimation of the parameter $\hat{\boldsymbol{\theta}}_{n'}$, the set of unlabeled samples X_p, , size of the query set $|X_q|$
Outputs: The query set X_q

/* Initializing the query set for this iteration */
$X_q \leftarrow \varnothing$
/* The loop for greedy batch querying */
for $j = 1 \rightarrow |X_q|$ **do**
  /* Query optimization and adding the result into the query set */
  $X_q \leftarrow X_q \cup \arg\min_{\mathbf{x} \in X_p} \operatorname{tr}\left[\hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; \mathbf{x}\right)^{-1} \hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; X_p\right)\right]$
  /* Removing the selected queries from the pool */
  $X_p \leftarrow X_p - X_q$
return X_q

Algorithm 3 (Settles and Craven, 2008)

Inspired by Zhang and Oles (2000), Settles and Craven (2008) employed Fisher information ratio to develop a pool-based active learning, which can be used in either sequential or batch querying. The pool that is used here is the set of unlabeled samples, X_p, which are assumed to be drawn from the test marginal $p(\mathbf{x})$. Queries are chosen from X_p, that is $X_q \subseteq X_p$. The test Fisher information matrix can be approximated by Monte-Carlo simulation over the samples in X_p, meaning $\hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; X_p\right)$. Similar to Algorithm 1, the updated training Fisher information matrix after querying a set X_q can be approximated by $\hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; X_{\mathcal{L}} \cup X_q\right)$. Thus, since we do have an approximation of both Fisher information matrices, the objective to minimize is chosen to be in the form of (31).

Similar to the Zhang and Oles (2000) algorithm, the proposal distribution q is ignored in the objective (or equivalently considered as being uniform). An additional assumption Settles and Craven (2008) made to simplify the optimization task is

$$\arg\min_{X_q \subset X_p} \operatorname{tr}\left[\hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; X_{\mathcal{L}} \cup X_q\right)^{-1} \hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; X_p\right)\right] \approx \arg\min_{X_q \subset X_p} \operatorname{tr}\left[\hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; X_q\right)^{-1} \hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; X_p\right)\right]. \tag{40}$$

This simplified optimization is easy to implement for sequential active learning. However, the combinatorial optimization required for batch active learning can easily become intractable. As shown in Algorithm 3, Settles and Craven (2008) used a greedy approach to do this optimization (the inner loop).

Algorithm 4 (Hoi et al., 2006, 2009)

The algorithms proposed by Hoi et al. (2006) and Hoi et al. (2009) are very similar to the one developed by Settles and Craven (2008), described above, except that they use a more sophisticated optimization method. Their method shown in Algorithm 4, is different

Algorithm 4: Hoi et al. (2006, 2009)

Inputs: Current estimation of the parameter $\hat{\boldsymbol{\theta}}_{n'}$, the set of unlabeled samples X_p, size of the query set $|X_q|$

Outputs: The query set X_q

/* Initializing the query set */

1 $X_q \leftarrow \varnothing$

/* The loop for greedy batch querying */

2 **for** $j = 1 \rightarrow |X_q|$ **do**

/* Query optimization */

3 $\tilde{\mathbf{x}} = \arg\min_{\mathbf{x} \in X_p} \operatorname{tr}\left[\hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; X_q \cup \{\mathbf{x}\}\right)^{-1} \hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; X_p\right)\right]$

/* Add the selected query into the query set */

4 $X_q \leftarrow X_q \cup \{\tilde{\mathbf{x}}\}$

/* Remove the selected instance from the pool */

5 $X_p \leftarrow X_p - \{\tilde{\mathbf{x}}\}$

6 **return** X_q

from Algorithm 3 mainly in the way that it greedily chooses the query at each inner loop iteration of the algorithm. While Algorithm 3 exclusively considers the contribution of each $\mathbf{x} \in X_q$, ignoring the samples selected in the previous iterations and considering only $\hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; \mathbf{x}\right)$ in each iteration (line 3 of Algorithm 3), Algorithm 4 takes into account all the queries chosen so far and instead considers $\hat{\mathbf{I}}\left(\hat{\boldsymbol{\theta}}_{n'}; X_q \cup \{\mathbf{x}\}\right)$ in each querying optimization (line 3 in Algorithm 4).

Hoi et al. (2006) and Hoi et al. (2009) showed that when using *binary logistic regression* classifier, their optimization (40) can be done by maximizing a *submodular* set function with respect to the query set X_q. This allowed them to use the well-known iterative algorithm proposed by Nemhauser et al. (1978), which guarantees a tight lower-bound for maximization of submodular and monotone set functions.

In the rest of this section, we show that minimizing this objective obtained from the above-mentioned assumptions, can be efficiently approximated by a monotonically submodular maximizing under *any* discriminative classifier. This is a generalization of the result derived by Hoi et al. (2006) that is obtained in case of using logistic regression classifier. As a consequence, FIR can be efficiently optimized with guaranteed tight bounds (Nemhauser et al., 1978; Nemhauser and Wolsey, 1978). First, we show in the following lemma that (40) is approximately equivalent to maximizing a simplified set function, for any unlabeled sample pool X_p:

Lemma 7 *Let $X_p, X_q \subseteq X$ be two non-empty and finite sets of samples randomly generated from $p(\mathbf{x})$ and its resample distribution $q(\mathbf{x})$, respectively, such that $X_q \subset X_p$, and the parameter $\delta \geq 0$ in (30) is a small constant. If assumptions (A0), (A4), (A6) and (A8) hold, then the following optimization problems are approximately equivalent for some function $g_{\boldsymbol{\theta}} : X \times \{1, ..., c\} \times X \rightarrow \mathbb{R}^+$, d-dimensional non-zero vector $\mathbf{v}_{\boldsymbol{\theta}}$ depending on $\mathbf{x}$ and y, and*

for all $\boldsymbol{\theta} \in \Omega$:

$$(i) \quad \arg\min_{X_q \subset X_p} \operatorname{tr}\left[\hat{\mathbf{I}}(\boldsymbol{\theta}; X_q)^{-1} \hat{\mathbf{I}}(\boldsymbol{\theta}; X_p)\right], \tag{41a}$$

$$(ii) \quad \arg\max_{X_q \subset X_p} \sum_{\mathbf{x} \in X_p - X_q} \sum_{y=1}^{c} \frac{-1}{\delta \cdot \| \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y)\|^{-2} + \sum_{\mathbf{x}' \in X_q} g_{\boldsymbol{\theta}}(\mathbf{x}, y, \mathbf{x}')}. \tag{41b}$$

The approximation is more accurate for smaller δ and well-conditioned Monte-Carlo approximation of proposal Fisher information matrix.

The proof can be found in Appendix C. Note that Lemma 7, as stated above, does not depend on the size of X_q. However, just as before, in practice it is usually assumed that $|X_q| > 0$ is fixed and therefore the optimizations in (41) should be considered with cardinality constraint. In general, combinatorial maximization problems can turn out to be intractable. Next, it is shown that the objective at hand is a monotonically submodular set function in terms of X_q and therefore can be maximized efficiently with a greedy approach such as that shown in Algorithm 4.

Theorem 8 *Suppose $f_{\boldsymbol{\theta}} : 2^{X_p} \rightarrow \mathbb{R}$ is defined as*

$$f_{\boldsymbol{\theta}}(X_q) = \sum_{\mathbf{x} \in X_p - X_q} \sum_{y=1}^{c} \frac{-1}{\delta \cdot \| \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y)\|^{-2} + \sum_{\mathbf{x}' \in X_q} g_{\boldsymbol{\theta}}(\mathbf{x}, y, \mathbf{x}')}, \quad \forall X_q \subseteq X_p, \tag{42}$$

with $\mathbf{v}_{\boldsymbol{\theta}}$ a d-dimensional vector depending on $\mathbf{x}$ and y, and $g_{\boldsymbol{\theta}}$ defined in (91). Then $f_{\boldsymbol{\theta}}$ is a submodular and monotone (non-decreasing) set function for all $\boldsymbol{\theta} \in \Omega$.

The proof is in Appendix D. The result above, together with Lemma 7, imply that the objective of (41b) is a monotonically increasing set function with respect to X_q. Below we present the main result that guarantees tight bounds for greedy maximization of monotonic submodular set functions. Details of this result, which is also shown to be the optimally efficient solution to submodular maximization, can be found in the seminal papers by Nemhauser et al. (1978) and Nemhauser and Wolsey (1978).

Theorem 9 (Nemhauser et al. (1978)) *Let $f_{\boldsymbol{\theta}} : 2^{X_p} \rightarrow \mathbb{R}$ be any submodular and non-decreasing set function with $f(\varnothing) = 0$ satisfied.*[8] *If X_q is the output of a greedy maximization algorithm, and X_q^* is the optimal maximizer of $f_{\boldsymbol{\theta}}$ with a cardinality constraint (fixed $|X_q|$), then we have*

$$f_{\boldsymbol{\theta}}(X_q) \geq \left[1 - \left(\frac{|X_q| - 1}{|X_q|}\right)^{|X_q|}\right] f_{\boldsymbol{\theta}}(X_q^*) \geq \left(1 - \frac{1}{e}\right) f_{\boldsymbol{\theta}}(X_q^*). \tag{43}$$

In Algorithm 4, the inner loop (lines 2 to 5) implements the minimization in (40) greedily. We have seen above that this set minimization is approximately equivalent to maximizing a submodular and monotone set maximization, which, in turn, is shown to be efficient.

8. This can always be assumed since maximizing a general set function $f(X_q)$ is equivalent to maximizing its adjusted version $g(X_q) := f(X_q) - f(\varnothing)$, which satisfies $g(\varnothing) = 0$.

Algorithm 5 (Chaudhuri et al., 2015b)

This algorithm uses FIR for doing a probabilistic pool-based active learning. It has extra assumptions in comparison to our general framework, which are briefly explained in Section 5.1.1. Note that these assumptions are to be made as well as those listed in Section 2. In such settings, Chaudhuri et al. (2015b) gave a finite-sample theoretical analysis for FIR when applied to pool-based active learning.

More specifically, suppose $p(\mathbf{x})$ is a uniform PMF and $q(\mathbf{x})$ is a general PMF, both defined over the pool X_p. Using the notations in (25), the training Fisher information can be written as $\mathbf{I}_q(\hat{\boldsymbol{\theta}}_{n'}) = \sum_{\mathbf{x}\in X_p} q(\mathbf{x})\,\mathbf{I}(\hat{\boldsymbol{\theta}}_{n'}, \mathbf{x})$. Now, assuming that $\mathbf{I}_p(\hat{\boldsymbol{\theta}}_{n'})$ has a singular decomposition of the form $\sum_{j=1}^{d} \sigma_j \mathbf{u}_j \mathbf{u}_j^\top$, FIR can be written as

$$\begin{aligned} \operatorname{tr}\Big[\mathbf{I}_q(\hat{\boldsymbol{\theta}}_{n'})^{-1}\,\mathbf{I}_p(\hat{\boldsymbol{\theta}}_{n'})\Big] &= \sum_{j=1}^{d} \sigma_j \operatorname{tr}\Big[\mathbf{I}_q(\hat{\boldsymbol{\theta}}_{n'})^{-1}\mathbf{u}_j\mathbf{u}_j^\top\Big] \\ &= \sum_{j=1}^{d} \sigma_j \mathbf{u}_j^\top\,\mathbf{I}_q(\hat{\boldsymbol{\theta}}_{n'})^{-1}\mathbf{u}_j. \end{aligned} \tag{44}$$

Minimizing the last term in (44) with respect to PMF $\{q(\mathbf{x})|\,\mathbf{x}\in X_p\}$ is equivalent to a semidefinite programming after introducing a set of auxiliary variables $t_j, j =, 1..., d$ and applying Schur complements (Vandenberghe and Boyd, 1996):

$$\begin{aligned} \underset{q(\mathbf{x}),\mathbf{x}\in X_p}{\arg\min} \quad & \sum_{j=1}^{d} \sigma_j t_j \\ \text{such that} \quad & \begin{bmatrix} t_j & \mathbf{u}_j^\top \\ \mathbf{u}_j & \sum_{\mathbf{x}\in X_p} q(\mathbf{x})\,\mathbf{I}(\hat{\boldsymbol{\theta}}_{n'}, \mathbf{x}) \end{bmatrix} \succeq 0, \\ & \sum_{\mathbf{x}\in X_p} q(\mathbf{x}) = 1. \end{aligned} \tag{45}$$

The steps for this querying method is shown in Algorithm 5. Note that the solution to (45) is slightly modified by mixing it with the uniform distribution over the pool. Such modification is mainly to establish their theoretical derivations. The mixing coefficient, $0 \le \lambda \le 1$, reciprocally depends on the number of queries. More specifically, Chaudhuri et al. (2015b) made it equal to $1 - \frac{1}{|X_q|^{1/6}}$. That is, as the number of queries increases, λ shrinks and so does the modification. Furthermore, in their analysis, they assumed that sampling from $\tilde{q}(\mathbf{x})$ (line 3 of Algorithm 5) is done *with replacement*. That is, label of a given sample might be queried multiple times.

5.2.1 Comparison with Other Information-theoretic Objectives

In the last part of this section, we compare FIR and two other common querying objectives from the field of information theory. Entropy of class labels and mutual information between labeled and unlabeled samples are two other common active learning objectives. Their goal is mainly to get the largest possible amount of information about *class labels of unlabeled samples* from each querying iteration, hence naturally pool-based.

Algorithm 5: Chaudhuri et al. (2015b)

Inputs: Current estimation of the parameter $\hat{\boldsymbol{\theta}}_{n'}$, the set of unlabeled samples X_p, size of the query set $|X_q|$

Outputs: The query set X_q

/* Solving the semidefinite programming */

1 $q(\mathbf{x}) \leftarrow$ solution to (45)

/* Modification of the solution */

2 $\tilde{q}(\mathbf{x}) \leftarrow \lambda q(\mathbf{x}) + (1-\lambda)U(\mathbf{x})$

/* Sampling with replacement from the modified proposal */

3 $\mathbf{x}_i \sim \tilde{q}(\mathbf{x}) \quad , i = 1, ..., |X_q|$

4 **return** $X_q = \{\mathbf{x}_1, ...\,\mathbf{x}_{|X_q|}\}$

Algorithm	Complexity
Entropy	$O(\lvert X_p\rvert cd)$
Mutual Information	$O\left(\lvert X_p\rvert \cdot \lvert X_q\rvert \cdot c^{\lvert X_q\rvert+1} d\right)$
Zhang and Oles (2000)	$O(\lvert X_p\rvert cd)$
Settles and Craven (2008)	$O\left(\lvert X_q\rvert \cdot \lvert X_p\rvert \cdot (cd + d^3)\right)$
Hoi et al. (2006, 2009)	$O\left(\lvert X_q\rvert \cdot \lvert X_p\rvert \cdot (cd + cd\lvert X_q\rvert + d^3)\right)$
Chaudhuri et al. (2015b)	$O\left(d^3\lvert X_p\rvert^2 + d^4\lvert X_p\rvert + d^5\right)$

Table 2: Computational complexity of different querying algorithms

Entropy-based querying, also known as uncertainty sampling, directly measures the uncertainty with respect to class label of each unlabeled sample and query those with highest uncertainty. It has been widely popular due to its simplicity and effectiveness especially in sequential active learning. However, it does not consider interaction between samples when selecting multiple queries, which can cause querying very similar samples (redundancy). Therefore, uncertainty sampling shows relatively poor performance in batch active learning. Mutual information, on the other hand, does not suffer from redundancy, however, it requires a much higher computational complexity.

These two objectives directly measure the amount of information each batch can have with respect to the class labels (hence prediction-based), as opposed to Fisher information as a measure of information regarding the distribution parameters (hence inference-based). However, there is no guarantee that by minimizing uncertainty of the class labels (or equivalently, choosing queries with highest amount of information about class labels), the prediction accuracy also increases. Whereas, as we showed earlier, FIR upper-bounds the expected asymptotic variance of a parameter inference loss function. From this point of view, FIR has a closer relationship with the performance of a classifier.

Table 2 shows computational complexity of the querying objectives. The algorithm by Fukumizu (2000) is excluded from this table since it cannot be used in pool-based sampling. Also note that the complexity reported for mutual information is for the case when it is optimized greedily. Nevertheless, it still contains an exponential term in its complexity. Entropy-based and Zhang and Oles (2000) have the lowest complexity, but in the expense of introducing redundancy into the batch of queries. Algorithms by Settles and Craven (2008), Hoi et al. (2006, 2009) and Chaudhuri et al. (2015b) become very expensive when d is large, whereas mutual information can easily get intractable for selecting batches of higher size (large $|X_q|$). Observe that algorithm by Hoi et al. (2006, 2009) is more expensive than Settles and Craven (2008). Recall that despite similarities in appearance, the former guarantees tight bound for its greedy optimization, whereas the latter does not.

The complexity for the algorithm by Chaudhuri et al. (2015b) is computed assuming that a barrier method (following path) is used as its numerical optimization (Boyd and Vandenberghe, 2004). From Table 2, this algorithm is the only one whose complexity increases quadratically with size of the pool $|X_p|$, and therefore can get significantly slow for huge pools. Furthermore, it does not depend on $|X_q|$ since the optimization in (45) as its main source of computations, only depends on $|X_p|$ and d. Furthermore, for this algorithm, computing $\mathbf{I}(\hat{\boldsymbol{\theta}}_{n'}, \mathbf{x})$ is assumed to cost $O(1)$ for each $\mathbf{x} \in X_p$ as it is taken to be independent of y.

6. Conclusion

In this paper, we focused on active learning algorithms in classification problems whose objectives are based on Fisher information criterion. As the primary result, we showed the dependency of the variance of the asymptotic distribution of log-likelihood ratio on the Fisher information of the training distribution. Then, we used this dependency to derive our novel theoretical contribution by establishing the Fisher information ratio (FIR) as the upper bound of expectation of such asymptotic variance. We also showed that replacing the true parameter by the current estimate does not remove such upper-boundedness provided that the current parameter estimate has been obtained using sufficient number of samples. Moreover, we discussed that several layers of approximations can be employed in practice to simplify FIR; simplifications, that can usually be avoided in pool-based active learning. Additionally, Monte-Carlo simulations and greedy algorithms can be used to evaluate and optimize the (simplified) FIR objective, respectively. Using this framework, we can distinguish the main differences between some of the existing FIR-based querying methods in the classification context. Such comparative analysis, not only shed light on the assumptions and simplifications of the existing algorithms, it can also be helpful for finding suitable directions in developing novel active learning algorithms based on the Fisher information criterion.

Finally, we remark that the log-likelihood ratio that is used here as the loss function is an inference-based performance metric. It naturally shows up based on the set of assumptions that are usually being made in FIR-based querying frameworks. The final goal of a classifier in machine learning is to predict labels of test samples as accurately as possible and therefore, arguably, prediction-based metrics such as 0/1 loss function better evaluate the performance of a classifier. While analyzing such metrics was out of scope of this paper,

analysis and development of querying algorithms using prediction-based metrics is definitely an exciting future research direction.

Acknowledgments

This work is primarily supported by NSF IIS-1118061. In addition, Todd Leen was supported under NSF 1258633, 1355603, and 1454956; Deniz Erdogmus was supported by NSF IIS-1149570, CNS-1136027 and SMA-0835976; and Jennifery Dy was also supported by NSF IIS 0915910 and NIH R01HL089856.

Appendix A. Statistical Background

Asymptotic analysis plays an important role in statistics. It considers extreme cases where the number of observations is increased with no bounds. In such scenarios, discussions on different notions of convergence of the sequence of random variables naturally arise. Generally speaking, there are three major types of stochastic convergence: *convergence in probability, convergence in law (distribution)* and *convergence with high probability (almost surely)*. Here, we focus on the two former modes of convergence, discuss two fundamental results based on them and formalize our notations regarding parameter estimators. Further details of the following definitions and results can be found in any standard statistical textbook such as Lehmann and Casella (1998).

A.1 Convergence of Sequence of Random Variables

Throughout this section, $\{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, ..., \boldsymbol{\theta}_n, ...\}$, denoted simply by $\{\boldsymbol{\theta}_n\}$, is a sequence of multivariate random variables lying in $\Omega \subseteq \mathbb{R}^d$. Also suppose that $\boldsymbol{\theta}_0$ is a constant vector and $\tilde{\boldsymbol{\theta}}$ is another random variable in the same space Ω.

Definition 10 *We say that the sequence* $\{\boldsymbol{\theta}_n\}$ converges in probability *to* $\boldsymbol{\theta}_0$ *and write* $\boldsymbol{\theta}_n \xrightarrow{P} \boldsymbol{\theta}_0$, *iff for every* $\varepsilon > 0$ *we have*

$$P(|\theta_{ni} - \theta_{0i}| > \varepsilon) \;\rightarrow\; 0, \quad \textit{for all } i = 1, ..., d, \tag{46}$$

where θ_{ni} *is the* i*'th component of* $\boldsymbol{\theta}_n$.

Convergence in probability is invariant with respect to any continuous mapping:

Proposition 11 (Brockwell and Davis 1991, Proposition 6.1.4) *If* $\boldsymbol{\theta}_n \xrightarrow{P} \boldsymbol{\theta}_0$ *and* $g : \Omega \rightarrow \mathbb{R}$ *is a continuous function, then* $g(\boldsymbol{\theta}_n) \xrightarrow{P} g(\boldsymbol{\theta}_0)$.

Definition 12 *We say that a sequence* $\{\boldsymbol{\theta}_n\}$ converges in law (in distribution) *to the random variable* $\tilde{\boldsymbol{\theta}}$ *and write* $\boldsymbol{\theta}_n \xrightarrow{L} \tilde{\boldsymbol{\theta}}$, *iff the sequence of their joint CDFs,* F_n, *point-wise converges to the joint CDF of* $\tilde{\boldsymbol{\theta}}$*:*

$$F_n(\mathbf{a}) = P(\theta_{n1} \leq a_1, ..., \theta_{nd} \leq a_d) \;\rightarrow\; F(\mathbf{a}) = P(\tilde{\theta}_1 \leq a_1, ..., \tilde{\theta}_d \leq a_d) \quad \forall \mathbf{a} \in \mathcal{C}_F \subseteq \mathbb{R}^d, \tag{47}$$

where $\mathcal{C}_F$ *is the set of continuity points of the CDF* F.

Equation (47) means that for large values of n, the distribution of $\boldsymbol{\theta}_n$ can be well approximated by the distribution of $\tilde{\boldsymbol{\theta}}$. Note that throughout this paper, for simplicity, we say that a random sequence $\{\boldsymbol{\theta}_n\}$ converges to a distribution with density function $p(\boldsymbol{\theta})$, or write $\boldsymbol{\theta}_n \xrightarrow{L} p(\boldsymbol{\theta})$, instead of the full statement that $\{\boldsymbol{\theta}_n\}$ converges in law to a random variable with that distribution.

Note that $\boldsymbol{\theta}_n \xrightarrow{P} \boldsymbol{\theta}_0$ suggests that $\boldsymbol{\theta}_n - \boldsymbol{\theta}_0 \xrightarrow{L} \delta(\boldsymbol{\theta})$ where δ is the Kronecker delta function, which can be viewed as the density function of a degenerate distribution at $\boldsymbol{\theta} = \mathbf{0}$. This, however, does not give any information about the speed with which $\boldsymbol{\theta}_n$ converges to $\boldsymbol{\theta}_0$. In order to take the rate into account, we consider the convergent distribution of the sequence $\{a_n \cdot (\boldsymbol{\theta}_n - \boldsymbol{\theta}_0)\}$, where a_n is any sequence of positive integers and $a_n \to \infty$(as $n \to \infty$). In practice a_n is usually considered to have the form n^r with $r > 0$.

Definition 13 *Assume $\boldsymbol{\theta}_n \xrightarrow{P} \boldsymbol{\theta}_0$. We say that the sequence $\{\boldsymbol{\theta}_n\}$ converges to $\boldsymbol{\theta}_0$ with* rate of convergence *$r > 0$, iff $n^r(\boldsymbol{\theta}_n - \boldsymbol{\theta}_0)$ converges in law to a random variable with non-degenerate distribution. Furthermore, the non-degenerate distribution is called the* asymptotic distribution *of $\boldsymbol{\theta}_n$.*

Next, we discuss some of the classic results in asymptotic statistics:

Theorem 14 (Law of Large Numbers, Brockwell and Davis 1991) *Let $\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_n$ be a set of independent and identically distributed (i.i.d) samples. If $\mathbb{E}[\boldsymbol{\theta}_i] = \boldsymbol{\mu}$, then*

$$\bar{\boldsymbol{\theta}}_n \;=\; \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\theta}_i \;\xrightarrow{P}\; \boldsymbol{\mu}\,. \tag{48}$$

Theorem 15 (Central Limit Theorem, Lehmann and Casella 1998) *Let $\boldsymbol{\theta}_1$,..., $\boldsymbol{\theta}_n$ be a set of i.i.d samples with mean $\mathbb{E}[\boldsymbol{\theta}_i] = \boldsymbol{\mu}$ and covariance $\mathrm{Cov}[\boldsymbol{\theta}_i] = \boldsymbol{\Sigma}$ (with a symmetric and positive semi-definite matrix $\boldsymbol{\Sigma}$), then the sequence of sample averages $\{\bar{\boldsymbol{\theta}}_n\}$ with $\bar{\boldsymbol{\theta}}_n = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{\theta}_i$ converges to the true mean with convergence rate $1/2$. Moreover, its asymptotic distribution is a zero-mean Gaussian distribution with covariance matrix $\boldsymbol{\Sigma}$, that is,*

$$\sqrt{n} \cdot (\bar{\boldsymbol{\theta}}_n - \boldsymbol{\mu}) \;\xrightarrow{L}\; \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}). \tag{49}$$

The following results are very useful when deriving the asymptotic distribution of a random sequence under a continuous mapping:

Theorem 16 (Multivariate Delta Method, first order, Lehmann and Casella 1998) *Let $\{\boldsymbol{\theta}_n\}$ be a sequence of random variables such that it converges to $\boldsymbol{\theta}_0$ with rate of convergence $1/2$ and a normal asymptotic distribution, that is, $\sqrt{n} \cdot (\boldsymbol{\theta}_n - \boldsymbol{\theta}_0) \xrightarrow{L} \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. If $g : \mathbb{R}^d \to \mathbb{R}$ is a continuously differentiable mapping and $\nabla_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0) \neq \mathbf{0}$, then*

$$\sqrt{n} \cdot \Big[g(\boldsymbol{\theta}_n) - g(\boldsymbol{\theta}_0) \Big] \;\xrightarrow{L}\; \mathcal{N}\Big(0, \nabla_{\boldsymbol{\theta}}^{\top} g(\boldsymbol{\theta}_0)\, \boldsymbol{\Sigma}\, \nabla_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0)\Big)\,. \tag{50}$$

Theorem 17 (Multivariate Delta Method, second order) *Let $\{\boldsymbol{\theta}_n\}$ be a sequence of random variables such that it converges to $\boldsymbol{\theta}_0$ with rate of convergence $1/2$ and a normal asymptotic distribution, that is, $\sqrt{n} \cdot (\boldsymbol{\theta}_n - \boldsymbol{\theta}_0) \xrightarrow{L} \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. If $g : \mathbb{R}^d \to \mathbb{R}$ is a continuously differentiable mapping where $\nabla_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0) = \mathbf{0}$ and $\nabla^2_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0)$ is non-singular in a neighborhood of $\boldsymbol{\theta}_0$, then the sequence $\{g(\boldsymbol{\theta}_n) - g(\boldsymbol{\theta}_0)\}$ converges in law to a mixture of random variables with degree-one Chi-square distributions, and the rate of convergence is one. More specifically,*

$$n \cdot \Big[g(\boldsymbol{\theta}_n) - g(\boldsymbol{\theta}_0) \Big] \;\xrightarrow{L}\; \sum_{i=1}^{d} \lambda_i \chi^2_1, \tag{51}$$

where λ_i's are eigenvalues of $\boldsymbol{\Sigma}^{1/2}\, \nabla_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0)\, \boldsymbol{\Sigma}^{1/2}$. Moreover, variance of this asymptotic distribution can be written as

$$\frac{1}{2} \left\| \boldsymbol{\Sigma}^{1/2}\, \nabla^2_{\mathbf{x}} g(\mathbf{x}_0)\, \boldsymbol{\Sigma}^{1/2} \right\|_F^2, \tag{52}$$

where $\| \cdot \|_F$ is the Frobenius norm.

Proof For proof see Appendix B. ■

A.2 Parameter Estimation

Now suppose that the set of independent and identically distributed (i.i.d) set of samples $\mathbf{x}_1, ... \mathbf{x}_n$ are generated from an underlying distribution that belongs to a parametric family, for which the density function $p(\mathbf{x} \,|\, \boldsymbol{\theta})$ can be represented by a multivariate parameter vector $\boldsymbol{\theta}$. Assume the true parameter is $\boldsymbol{\theta}_0$, that is $\{\mathbf{x}_i\} \sim p(\mathbf{x} \,|\, \boldsymbol{\theta}_0), i = 1, ..., n$. An *estimator* $\boldsymbol{\theta}_n = \boldsymbol{\theta}(\mathbf{x}_1, ..., \mathbf{x}_n)$ is a function that maps the observed random variables to a point in the parameter space Ω. The subscript n in $\boldsymbol{\theta}_n$ indicates its dependence on the sample size. Since the observations are generated randomly, the estimators are also random and thus $\{\boldsymbol{\theta}_n\}$ can be viewed as a sequence of random variables. There are some reserved terms for such a sequence, which we introduce in the remaining of this section:

Definition 18 (Consistency) *We say that an estimator $\boldsymbol{\theta}_n$ is* consistent *iff $\boldsymbol{\theta}_n \xrightarrow{P} \boldsymbol{\theta}_0$.*

Based on Theorem 14, sample average of the observation set is a consistent estimator of the true mean of the samples. Another important characteristic of estimators is based on the following bound over their covariance matrices:

Theorem 19 (Cramér-Rao, Lehmann and Casella 1998) *Let $\mathbf{x}_1, ..., \mathbf{x}_n \sim p(\mathbf{x} \,|\, \boldsymbol{\theta}_0)$ and $\boldsymbol{\theta}_n = \boldsymbol{\theta}(\mathbf{x}_1, ..., \mathbf{x}_n)$ be an estimator. If the first moment of $\boldsymbol{\theta}_n$ is differentiable with respect to the parameter vector and its second moment is finite, then the following inequality holds for every $\boldsymbol{\theta} \in \Omega$:*

$$\mathrm{Cov}[\boldsymbol{\theta}_n] \;\succeq\; -\left(\nabla_{\boldsymbol{\theta}} \mathbb{E}[\boldsymbol{\theta}_n]\right)^{\top} \mathbf{I}(\boldsymbol{\theta})^{-1} \nabla_{\boldsymbol{\theta}} \mathbb{E}[\boldsymbol{\theta}_n]. \tag{53}$$

The right-hand-side of (53) is called the *Cramér-Rao bound* of the estimator, where the middle term is the inverse of the *Fisher information matrix* of the parametric distribution $p(\mathbf{x} \,|\, \boldsymbol{\theta})$, defined as

$$\mathbf{I}(\boldsymbol{\theta}) \;:=\; \mathbb{E}\left[\nabla_{\boldsymbol{\theta}} \log p(\mathbf{x} \,|\, \boldsymbol{\theta}) \cdot \nabla_{\boldsymbol{\theta}}^{\top} \log p(\mathbf{x} \,|\, \boldsymbol{\theta}) \right].$$

Theorem 19 suggests that for an unbiased estimator $\boldsymbol{\theta}_n$, the inequality over the covariance matrix becomes: $\text{Cov}[\boldsymbol{\theta}_n] \succeq \mathbf{I}(\boldsymbol{\theta})^{-1}, \forall\, \boldsymbol{\theta} \in \Omega$.

Definition 20 (Efficiency) *We say that an estimator $\boldsymbol{\theta}_n$ is* efficient*, iff it attains the Cramér-Rao bound, meaning that* $\text{Cov}[\boldsymbol{\theta}_n]$ *achieves the lower-bound in (53) for every* $n = 1, 2, \ldots$ *. Furthermore, we say that $\boldsymbol{\theta}_n$ is* asymptotically efficient*, iff the lower bound is attained asymptotically (when $n \to \infty$).*

Appendix B. Proof of Second-order Multivariate Delta Method

In order to prove this theorem, we have to formulate the statistical Taylor expansion. This, in turn, needs a brief introduction of stochastic order notations.

B.1 Stochastic Order Notations

The stochastic order notations are denoted by o_p and O_p, where the former is equivalent to convergence in probability (Definition 10) and the latter implies *boundedness in probability*. In what follows, if otherwise stated, $\{\boldsymbol{\theta}_n\}$ is a sequence of multivariate random variables lying in $\Omega \subseteq \mathbb{R}^d$ and $\{a_n\}$ is a sequence of strictly positive real numbers. The skipped proofs can be found in many textbooks on asymptotic theory, such as Brockwell and Davis (1991, Chapter 6).

Definition 21 *We write $\boldsymbol{\theta}_n = o_p(a_n)$ iff*

$$\frac{\theta_{in}}{a_n} \;=\; o_p(1), \quad \textit{for all } i = 1, ..., d \tag{54}$$

Definition 22 *We write $\boldsymbol{\theta}_n = O_p(a_n)$ iff the sequence $\left\{\frac{\theta_{in}}{a_n}\right\}$ is* bounded in probability *for every $i = 1, ..., d$, that is, for every $\epsilon > 0$ there exists δ_ϵ such that*

$$P\left(\left|\frac{\theta_{in}}{a_n}\right| > \delta_\epsilon\right) \;<\; \epsilon, \quad n = 1, 2, ... \tag{55}$$

We also need the following propositions:

Proposition 23 (Brockwell and Davis, 1991) *Let $\{\theta_n\}$ and $\{\eta_n\}$ be two sequences of scalar random variables, and $\{a_n\}$ and $\{b_n\}$ be two sequences of positive real numbers. If $\theta_n = O_p(a_n)$ and $\eta_n = o_p(b_n)$, then*

(i) $\theta_n^2 = O_p(a_n^2)$

(ii) $\theta_n \eta_n = o_p(a_n b_n)$

Proposition 24 *The followings are true:*[9]

(i) $\boldsymbol{\theta}_n = o_p(a_n) \;\Leftrightarrow\; \| \boldsymbol{\theta}_n \| = o_p(a_n)$.

(ii) $\boldsymbol{\theta}_n = O_p(a_n) \;\Leftrightarrow\; \| \boldsymbol{\theta}_n \| = O_p(a_n)$.

9. Unless subscripted otherwise, $\| \cdot \|$ denotes the L_2 norm in all the equations.

Proof The proof of part (i) can be found in Brockwell and Davis (1991, Proposition 6.1.2). Here, we only prove part (ii):

$(ii, \Rightarrow)$: Since $\boldsymbol{\theta}_n = O_p(a_n)$, for every $\varepsilon > 0$ and for every $i = 1, ..., d$, there exists a coefficient $\delta_i > 0$ such that

$$P\left(|\theta_{ni}| > a_n \cdot \delta_i\right) \;<\; \frac{\varepsilon}{d} \quad , n = 1, 2, ... \tag{56}$$

Define $\delta_{\max} = \max\{\delta_1, ..., \delta_d\}$ and note that we can write

$$\begin{aligned}\left\{\boldsymbol{\theta}_n : \; \sum_{i=1}^{d} |\theta_{ni}|^2 > (d \cdot a_n \cdot \delta_{\max})^2\right\} \;&\subseteq\; \left[\bigcap_{i=1}^{d} \{\boldsymbol{\theta}_n : \; |\theta_{ni}| \le a_n \cdot \delta_{\max}\}\right]^c \\ &= \bigcup_{i=1}^{d} \{\boldsymbol{\theta}_n : \; |\theta_{ni}| > a_n \cdot \delta_{\max}\}, \end{aligned} \tag{57}$$

implying that

$$\begin{aligned} P\left(\| \boldsymbol{\theta}_n \|^2 > (d \cdot a_n \cdot \delta_{\max})^2\right) \;&\le\; P\left(\bigcup_{i=1}^{d} \{\boldsymbol{\theta}_n : \; |\theta_{ni}| > a_n \cdot \delta_{\max}\}\right) \\ &\le \sum_{i=1}^{d} P(|\theta_{ni}| > a_n \cdot \delta_{\max}). \end{aligned} \tag{58}$$

Furthermore, for every $i \in \{1, ..., d\}$, we have $\delta_{\max} \ge \delta_i$. Consequently, the interval $(a_n \delta_{\max}, \infty)$ is a subset of $(a_n \delta_i, \infty)$ and $P(|\theta_{ni}| > a_n \delta_{\max}) \le P(|\theta_{ni}| > a_n \delta_i)$. This implies that

$$P\left(\| \boldsymbol{\theta}_n \|^2 > (d \cdot a_n \cdot \delta_{\max})^2\right) \;\le\; \sum_{i=1}^{d} P(|\theta_{ni}| > a_n \cdot \delta_i) \;<\; \varepsilon. \tag{59}$$

Therefore, for every $\varepsilon > 0$, we can choose $\delta_\varepsilon = d \cdot \delta_{\max}$ such that $P\left(\frac{\| \boldsymbol{\theta}_n \|}{a_n} > \delta_\varepsilon\right) < \varepsilon$ for every $n = 1, 2, ...$ Therefore, by definition, $\| \boldsymbol{\theta}_n \| = O_p(a_n)$.
$(ii, \Leftarrow)$: Suppose $\| \boldsymbol{\theta}_n \| = O_p(a_n)$, that is for every $\varepsilon > 0$ we can find $\delta_\varepsilon > 0$ such that

$$P(\| \boldsymbol{\theta}_n \| > a_n \cdot \delta_\varepsilon) \;< \varepsilon \quad , n = 1, 2, ... \tag{60}$$

It is clear that for any given $i \in \{1, ..., d\}$ we have

$$\{\boldsymbol{\theta}_n : |\theta_{ni}| > a_n \cdot \delta_\varepsilon\} \;\subseteq\; \{\boldsymbol{\theta}_n : \| \boldsymbol{\theta}_n \| > a_n \cdot \delta_\varepsilon\}, \tag{61}$$

hence

$$P(|\theta_{ni}| > a_n \cdot \delta_\varepsilon) \;\le\; P(\| \boldsymbol{\theta}_n \| > a_n \cdot \delta_\varepsilon) \;< \epsilon \quad , n = 1, 2, ... \tag{62}$$

meaning that $\theta_{ni} = O_p(a_n), \forall i \in \{1, ..., d\}$, or equivalently $\boldsymbol{\theta}_n = O_p(a_n)$. ■

Proposition 25 *If $\boldsymbol{\theta}_n = O_p(a_n)$ and $a_n \to 0 (as\ n \to \infty)$, then $\boldsymbol{\theta}_n = o_p(1)$.*

Proof The goal is to show $\boldsymbol{\theta}_n = o_p(1)$ or equivalently $\| \boldsymbol{\theta}_n \| = o_p(1)$ by proving that $P(\| \boldsymbol{\theta}_n \| > \varepsilon) \to 0 (\text{as } n \to \infty)$ for every $\varepsilon > 0$. Fix ε to a positive real number. In order to have the sequence of probability numbers $\{P(\| \boldsymbol{\theta}_n \| > \varepsilon)\}$ converging to zero, for every $\varepsilon_0 > 0$ there should exist a positive integer $N > 0$ such that

$$P(\| \boldsymbol{\theta}_n \| > \varepsilon) < \varepsilon_0 \quad \forall n > N. \tag{63}$$

Because of the assumption of being bounded by a_n, that is $\boldsymbol{\theta}_n = O_p(a_n)$ or equivalently $\| \boldsymbol{\theta}_n \| = O_p(a_n)$, we can choose a real number $\delta_0 > 0$ such that

$$P(\| \boldsymbol{\theta}_n \| > a_n \delta_0) \;\; < \;\; \varepsilon_0 \quad n = 1, 2, ... \tag{64}$$

On the other hand, since $a_n \to 0 (\text{as } n \to \infty)$, there exists a large enough number $N_0 > 0$ such that $0 < a_n < \frac{\varepsilon}{\delta_0}$ for all $n > N_0$. Therefore, we get

$$[0, a_n \delta_0] \;\; \subseteq \;\; [0, \varepsilon] \quad \forall n > N_0, \tag{65}$$

implying that

$$P(\| \boldsymbol{\theta}_n \| \leq a_n \delta_0) \;\; \leq \;\; P(\| \boldsymbol{\theta}_n \| \leq \varepsilon) \quad \forall n > N_0. \tag{66}$$

From inequalities (64) and (66), and noticing that the latter holds for all n whereas the former is satisfied when $n > N_0$, one can write:

$$P(\| \boldsymbol{\theta}_n \| > \varepsilon) \;\; \leq \;\; P(\| \boldsymbol{\theta}_n \| > a_n \delta_0) \;\; < \;\; \varepsilon_0 \quad \forall n > N_0. \tag{67}$$

Therefore, for every $\varepsilon_0 > 0$, equation (63) is guaranteed if N is chosen to be equal to N_0 so that inequality (66) is satisfied. Similarly, this can be written for every $\varepsilon > 0$, thus the proof is complete. ■

Proposition 26 (Serfling 2002, Chapter 1) *Let $\{\boldsymbol{\theta}_n\}$ be a sequence of random variables. If there exists a random variable $\tilde{\boldsymbol{\theta}}$ such that $\boldsymbol{\theta}_n \xrightarrow{L} \tilde{\boldsymbol{\theta}}$, then $\boldsymbol{\theta}_n = O_p(1)$.*

B.2 Second-order Statistical Taylor Expansion

Now we are ready to establish the second-order statistical Taylor expansion.

Theorem 27 *Let $\{\boldsymbol{\theta}_n\}$ be a sequence of random vectors in a convex and compact set $\Omega \subseteq \mathbb{R}^d$ and $\boldsymbol{\theta}_0 \in \Omega$ be a constant vector such that $\boldsymbol{\theta}_n - \boldsymbol{\theta}_0 = O_p(a_n)$ where $a_n \to 0 (\text{as } n \to \infty)$. If $g : \Omega \to \mathbb{R}$ is a $\mathcal{C}^3$ function , then*

$$g(\boldsymbol{\theta}_n) \;\; = \;\; g(\boldsymbol{\theta}_0) \;\; + \;\; \nabla_{\boldsymbol{\theta}}^{\top} g(\boldsymbol{\theta}_0)(\boldsymbol{\theta}_n - \boldsymbol{\theta}_0) \;\; + \;\; \frac{1}{2}(\boldsymbol{\theta}_n - \boldsymbol{\theta}_0)^{\top} \nabla_{\boldsymbol{\theta}}^2 g(\boldsymbol{\theta}_0)(\boldsymbol{\theta}_n - \boldsymbol{\theta}_0) \;\; + \;\; o_p(a_n^2). \tag{68}$$

Proof Since g is twice continuously differentiable in a neighborhood of $\boldsymbol{\theta}_0$, it can be written in terms of the Taylor expansion as

$$g(\boldsymbol{\theta}) \;\; = \;\; g(\boldsymbol{\theta}_0) \;\; + \;\; (\boldsymbol{\theta} - \boldsymbol{\theta}_0)^{\top} \nabla_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0) \;\; + \;\; \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_0)^{\top} \nabla_{\boldsymbol{\theta}}^2 g(\boldsymbol{\theta}_0)(\boldsymbol{\theta} - \boldsymbol{\theta}_0) \;\; + \;\; r_2(\boldsymbol{\theta}, \boldsymbol{\theta}_0), \tag{69}$$

where $r_2(\boldsymbol{\theta}, \boldsymbol{\theta}_0)$ is the Lagrange remainder of second order. Based on Taylor's polynomial theorem for multivariate functions, there exists a number $t \in [0, 1]$ such that $\boldsymbol{\theta}^* = t\,\boldsymbol{\theta} + (1 - t)\,\boldsymbol{\theta}_0 \in \Omega$ (due to convexity of Ω) and

$$r_2(\boldsymbol{\theta}, \boldsymbol{\theta}_0) \;\; = \;\; \frac{1}{6} \sum_{1 \leq i,j,k \leq d} \frac{\partial^3 g(\boldsymbol{\theta}^*)}{\partial \theta_i \partial \theta_j \partial \theta_k} (\theta_i - \theta_{0i})(\theta_j - \theta_{0j})(\theta_k - \theta_{0k}). \tag{70}$$

But since Ω is compact and $g \in \mathcal{C}^3$, the third derivative of g is bounded[10] and therefore there exists $M > 0$ such that

$$\left| \frac{\partial^3 g(\boldsymbol{\theta})}{\partial \theta_i \partial \theta_j \partial \theta_k} \right| \;\; \leq \;\; M \quad , \forall \boldsymbol{\theta} \in \Omega \;\; , \forall i, j, k \in \{1, ..., d\}. \tag{71}$$

Hence, the Lagrange remainder can be bounded by

$$\begin{aligned} |r_2(\boldsymbol{\theta}, \boldsymbol{\theta}_0)| \;\; &\leq \;\; \frac{M}{6} \sum_{1 \leq i,j,k \leq 3} |\theta_i - \theta_{0i}| \cdot |\theta_j - \theta_{0j}| \cdot |\theta_k - \theta_{0k}| \\ &= \;\; \frac{M}{6} \| \boldsymbol{\theta} - \boldsymbol{\theta}_0 \|_1^3 \\ &\leq \;\; \frac{M'}{6} \| \boldsymbol{\theta} - \boldsymbol{\theta}_0 \|^3, \end{aligned} \tag{72}$$

where $M' = c_u M$ and c_u is obtained from the equivalence of norms in $\mathbb{R}^d$ vector space.[11] Now define the function $h : \Omega \to \mathbb{R}$ as below

$$h(\boldsymbol{\theta}) \;\; := \;\; \begin{cases} \dfrac{r_2(\boldsymbol{\theta}, \boldsymbol{\theta}_0)}{\| \boldsymbol{\theta} - \boldsymbol{\theta}_0 \|^2 / 2} & , \boldsymbol{\theta} \neq \boldsymbol{\theta}_0 \\ 0 & , \boldsymbol{\theta} = \boldsymbol{\theta}_0 \end{cases} \tag{74}$$

Note that $h(\boldsymbol{\theta})$ is continuous at $\boldsymbol{\theta} = \boldsymbol{\theta}_0$: due to boundedness of $r_2(\boldsymbol{\theta}, \boldsymbol{\theta}_0)$, $h(\boldsymbol{\theta})$ is also bounded by

$$|h(\boldsymbol{\theta})| \;\; \leq \;\; \frac{M'}{3} \| \boldsymbol{\theta} - \boldsymbol{\theta}_0 \|. \tag{75}$$

Hence, for every $\varepsilon > 0$, we can select $\delta_\varepsilon = \frac{3\varepsilon}{M'}$ such that the following continuity condition holds:

$$\| \boldsymbol{\theta} - \boldsymbol{\theta}_0 \| < \delta_\varepsilon \;\; \Rightarrow \;\; |h(\boldsymbol{\theta})| \;\; \leq \;\; \varepsilon. \tag{76}$$

10. This is because of the following Theorem in real analysis:

    Theorem 28 *Let X and Y be two vector spaces. If $g : X \to Y$ is continuous and X is compact, then $f(X)$ is compact in Y.*

    In special case of this theorem, where $Y = \mathbb{R}$, compactness of $f(X)$ is equivalent to boundedness and closedness.

11. Two norm functions $\| \cdot \|_{(1)}$ and $\| \cdot \|_{(2)}$ in a vector space Ω, are called *equivalent* iff there exist constants $c_u \geq c_d > 0$ such that

$$c_d \| \boldsymbol{\theta} \|_{(2)} \;\; \leq \;\; \| \boldsymbol{\theta} \|_{(1)} \;\; \leq \;\; c_u \| \boldsymbol{\theta} \|_{(2)} \quad , \forall \boldsymbol{\theta} \in \Omega. \tag{73}$$

Continuity of $h(\boldsymbol{\theta})$ at $\boldsymbol{\theta} = \boldsymbol{\theta}_0$ implies $\lim_{\boldsymbol{\theta}\to\boldsymbol{\theta}_0} h(\boldsymbol{\theta}) = h(\boldsymbol{\theta}_0) = 0$. Furthermore, since $\boldsymbol{\theta}_n - \boldsymbol{\theta}_0 = O_p(a_n)$ and $a_n \to 0$(as $n \to \infty$), Proposition 25 suggests that $\boldsymbol{\theta}_n - \boldsymbol{\theta}_0 = o_p(1)$. These two enable us to use Proposition 11 and write

$$h(\boldsymbol{\theta}_n) - h(\boldsymbol{\theta}_0) \;=\; h(\boldsymbol{\theta}_n) \;=\; o_p(1). \tag{77}$$

Finally, from equation (74) and Propositions 23, 24 and 25, we can write that

$$r_2(\boldsymbol{\theta}_n, \boldsymbol{\theta}_0) \;=\; h(\boldsymbol{\theta}_n) \cdot \frac{\| \boldsymbol{\theta}_n - \boldsymbol{\theta}_0 \|^2}{2} \;=\; o_p(1) \cdot O_p(a_n^2) \;=\; o_p(a_n^2) \tag{78}$$

■

B.3 Second-order Multivariate Delta Method

Finally, here is the proof of second-order multivariate Delta method (Theorem 17):

Proof From the assumption for convergence in law, i.e., $\sqrt{n}(\boldsymbol{\theta}_n - \boldsymbol{\theta}_0) \xrightarrow{L} \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$, and Proposition 26, one concludes that $\sqrt{n}(\boldsymbol{\theta}_n - \boldsymbol{\theta}_0) = O_p(1)$ and therefore $\boldsymbol{\theta}_n - \boldsymbol{\theta}_0 = O_p\left(\frac{1}{\sqrt{n}}\right)$. Thus, we can use Theorem 27 with $a_n = \frac{1}{\sqrt{n}}$ to write

$$g(\boldsymbol{\theta}) \;=\; g(\boldsymbol{\theta}_0) \;+\; (\boldsymbol{\theta} - \boldsymbol{\theta}_0)^\top \nabla_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0) \;+\; \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_0)^\top \nabla^2_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0)(\boldsymbol{\theta} - \boldsymbol{\theta}_0) \;+\; o_p\left(\frac{1}{n}\right), \tag{79}$$

hence

$$\begin{aligned} n\Big[g(\boldsymbol{\theta}) - g(\boldsymbol{\theta}_0)\Big] \;&=\; \frac{1}{2}\left[\sqrt{n}\cdot(\boldsymbol{\theta} - \boldsymbol{\theta}_0)\right]^\top \nabla^2_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0) \left[\sqrt{n}\cdot(\boldsymbol{\theta} - \boldsymbol{\theta}_0)\right] \;+\; o_p(1) \\ &\xrightarrow{L}\; \frac{1}{2}\mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})^\top \nabla^2_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0) \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}) \\ &=\; \frac{1}{2}\mathcal{N}(\mathbf{0}, \mathbb{I}_d)^\top \left[\boldsymbol{\Sigma}^{1/2}\, \nabla^2_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0)\, \boldsymbol{\Sigma}^{1/2}\right] \mathcal{N}(\mathbf{0}, \mathbb{I}_d). \end{aligned} \tag{80}$$

Define $\boldsymbol{\Gamma} \coloneqq \boldsymbol{\Sigma}^{1/2}\, \nabla^2_{\boldsymbol{\theta}} g(\boldsymbol{\theta}_0)\, \boldsymbol{\Sigma}^{1/2}$ and rewrite the right-hand-side element-wise as

$$\frac{1}{2}\mathcal{N}(\mathbf{0}, \mathbb{I}_d)^\top \boldsymbol{\Gamma} \mathcal{N}(\mathbf{0}, \mathbb{I}_d) \;=\; \frac{1}{2}\sum_{i=1}^{d} \lambda_i \mathcal{N}(0,1)^2 \;=\; \frac{1}{2}\sum_{i=1}^{d} \lambda_i \chi^2_1, \tag{81}$$

where λ_i's are eigenvalues of $\boldsymbol{\Gamma}$. Finally, noting that the terms in the Chi-square mixture are independent, variance of the convergent random variable can be easily computed as

$$\begin{aligned} \mathrm{Var}\left[\frac{1}{2}\sum_{i=1}^{d} \lambda_i \chi^2_1\right] \;&=\; \frac{1}{4}\sum_{i=1}^{d} \lambda_i^2 \cdot \mathrm{Var}\left[\chi^2_1\right] \\ &=\; \frac{1}{2}\sum_{i=1}^{d} \lambda_i^2 \\ &=\; \frac{1}{2}\left\| \boldsymbol{\Sigma}^{1/2}\, \nabla^2_{\mathbf{x}} g(\mathbf{x}_0)\, \boldsymbol{\Sigma}^{1/2} \right\|_F^2. \end{aligned} \tag{82}$$

■

Appendix C. Proof of Lemma 7

We first substitute the score function of the classifier, i.e.,

$$\nabla_{\boldsymbol{\theta}} \log p(y|\,\mathbf{x}, \boldsymbol{\theta}) \;=\; \frac{\nabla_{\boldsymbol{\theta}} p(y|\,\mathbf{x}, \boldsymbol{\theta})}{p(y|\,\mathbf{x}, \boldsymbol{\theta})},$$

into Monte-Carlo approximation of $\mathbf{I}_q$ to get

$$\begin{aligned} \hat{\mathbf{I}}(\boldsymbol{\theta}; X_q) \;&=\; \frac{1}{|X_q|}\sum_{\mathbf{x}\in X_q}\sum_{y=1}^{c} p(y|\,\mathbf{x}, \boldsymbol{\theta}) \cdot \frac{\nabla_{\boldsymbol{\theta}} p(y|\,\mathbf{x}, \boldsymbol{\theta}) \nabla^\top_{\boldsymbol{\theta}} p(y|\,\mathbf{x}, \boldsymbol{\theta})}{p(y|\,\mathbf{x}, \boldsymbol{\theta})^2} \;+\; \delta\mathbb{I}_d \\ &=\; \frac{1}{|X_q|}\sum_{\mathbf{x}\in X_q}\sum_{y=1}^{c} \frac{\nabla_{\boldsymbol{\theta}} p(y|\,\mathbf{x}, \boldsymbol{\theta}).\nabla^\top_{\boldsymbol{\theta}} p(y|\,\mathbf{x}, \boldsymbol{\theta})}{p(y|\,\mathbf{x}, \boldsymbol{\theta})} \;+\; \delta\mathbb{I}_d. \end{aligned} \tag{83}$$

Define the vector $\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y) \coloneqq \nabla_{\boldsymbol{\theta}} p(y|\,\mathbf{x}, \boldsymbol{\theta}) \Big/ \sqrt{p(y|\,\mathbf{x}, \boldsymbol{\theta})}$ and rewrite $\hat{\mathbf{I}}(\boldsymbol{\theta}; X_q)$ as

$$\hat{\mathbf{I}}(\boldsymbol{\theta}; X_q) \;=\; \frac{1}{|X_q|}\sum_{\mathbf{x}\in X_q}\sum_{y=1}^{c} \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y).\, \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y)^\top + \delta \cdot \mathbb{I}_d. \tag{84}$$

On the other hand, since $X_q \subset X_p$ we can write $\hat{\mathbf{I}}(\boldsymbol{\theta}; X_p)$ in terms of $\hat{\mathbf{I}}(\boldsymbol{\theta}; X_q)$ by breaking the summation over X_p into summations over X_q and $X_p - X_q$ as follows:

$$\begin{aligned} \hat{\mathbf{I}}(\boldsymbol{\theta}; X_p) \;&=\; \frac{|X_q|}{|X_p|}\left[\frac{1}{|X_q|}\sum_{\mathbf{x}\in X_q}\sum_{y=1}^{c} \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y).\, \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y)^\top + \delta \cdot \mathbb{I}_d\right] \\ &\quad\; + \frac{1}{|X_p|}\sum_{\mathbf{x}\in X_p - X_q}\sum_{y=1}^{c} \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y).\, \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y)^\top \;+\; \delta\left(\frac{|X_p| - |X_q|}{|X_p|}\right)\cdot \mathbb{I}_d \\ &=\; \left(\frac{|X_q|}{|X_p|}\right)\cdot \hat{\mathbf{I}}(\boldsymbol{\theta}; X_q) \;+\; \frac{1}{|X_p|}\sum_{\mathbf{x}\in X_p - X_q}\sum_{y=1}^{c} \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y).\, \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y)^\top \\ &\quad\; + \delta\left(\frac{|X_p| - |X_q|}{|X_p|}\right)\cdot \mathbb{I}_d \end{aligned} \tag{85}$$

Now that we related the Fisher information matrices to each other, we can compute the product of $\hat{\mathbf{I}}(\boldsymbol{\theta}; X_p)$ and $\hat{\mathbf{I}}(\boldsymbol{\theta}; X_q)^{-1}$ as

$$\begin{aligned} \hat{\mathbf{I}}(\boldsymbol{\theta}; X_q)^{-1}\hat{\mathbf{I}}(\boldsymbol{\theta}; X_p) \;&=\; \left(\frac{|X_q|}{|X_p|}\right)\cdot \mathbb{I}_d \;+\; \frac{\hat{\mathbf{I}}(\boldsymbol{\theta}; X_q)^{-1}}{|X_p|}\left[\sum_{\mathbf{x}\in X_p - X_q}\sum_{y=1}^{c} \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y)\cdot \mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}, y)^\top\right] \\ &\quad\; + \delta\left(\frac{|X_p| - |X_q|}{|X_p|}\right)\cdot \hat{\mathbf{I}}(\boldsymbol{\theta}; X_q)^{-1}. \end{aligned} \tag{86}$$

Applying the trace function to both sides of the equation will result

$$\begin{aligned}\operatorname{tr}\left[\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)^{-1}\hat{\mathbf{I}}(\boldsymbol{\theta};X_p)\right] &= \frac{|X_q|\cdot d}{|X_p|}+\frac{1}{|X_p|}\sum_{\mathbf{x}\in X_p-X_q}\sum_{y=1}^{c}\operatorname{tr}\left[\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)^{-1}\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\cdot\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)^{\top}\right]\\ &\quad+\ \delta\left(\frac{|X_p|-|X_q|}{|X_p|}\right)\cdot\operatorname{tr}\left[\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)^{-1}\right]\\ &\approx \frac{|X_q|\cdot d}{|X_p|}\ +\ \frac{1}{|X_p|}\sum_{\mathbf{x}\in X_p-X_q}\sum_{y=1}^{c}\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)^{\top}\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)^{-1}\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y),\end{aligned}\tag{87}$$

where the last term is dropped since the overloading constant, δ, is assumed to be small. Furthermore, the term including $\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)^{-1}$ can be approximated by replacing the weighted harmonic mean of the eigenvalues of $\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)$ by their weighted arithmetic mean (Hoi et al., 2006)

$$\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)^{\top}\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)^{-1}\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\ \approx\ \frac{\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^4}{\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)^{\top}\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)}.\tag{88}$$

Note that this approximation becomes exact when the condition number of $\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)$ is one. Substituting $\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)$ from equation (84) into the denominator of the approximation above yields

$$\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)^{\top}\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\ =\ \frac{1}{|X_q|}\sum_{\mathbf{x}'\in X_q}\sum_{y'=1}^{c}\left[\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)^{\top}\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}',y')\right]^2\ +\ \delta\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2\tag{89}$$

Assume that the value of $\boldsymbol{\theta}$ is not located at the stationary point of the conditional density $p(y|\,\mathbf{x},\boldsymbol{\theta})$, hence $\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)$ is not the zero vector. Integrating approximation (88) with equation (87) results

$$\begin{aligned}\operatorname{tr}\left[\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)^{-1}\hat{\mathbf{I}}(\boldsymbol{\theta};X_p)\right] &\approx \frac{|X_q|\cdot d}{|X_p|}\\ &+\ \frac{1}{|X_p|}\sum_{\mathbf{x}\in X_p-X_q}\sum_{y=1}^{c}\frac{1}{\delta\cdot\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^{-2}+\sum_{\mathbf{x}'\in X_q}g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')},\end{aligned}\tag{90}$$

where

$$g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\ :=\ \frac{1}{|X_q|}\sum_{y'=1}^{c}\left[\frac{\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)^{\top}\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x}',y')}{\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}\right]^2.\tag{91}$$

Finally by removing the constants in (90), we get the optimization

$$\begin{aligned}&\operatorname*{arg\,min}_{X_q\subset X_p}\ \operatorname{tr}\left[\hat{\mathbf{I}}(\boldsymbol{\theta};X_q)^{-1}\hat{\mathbf{I}}(\boldsymbol{\theta};X_p)\right]\\ &\qquad\approx\ \operatorname*{arg\,max}_{X_q\subset X_p}\sum_{\mathbf{x}\in X_p-X_q}\sum_{y=1}^{c}\frac{-1}{\delta\cdot\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^{-2}+\sum_{\mathbf{x}'\in X_q}g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')}.\end{aligned}\tag{92}$$

Appendix D. Proof of Theorem 8

Proof of this Theorem is a generalization of the discussion by Hoi et al. (2006), with clarification of all the assumptions and approximations made.

First, note that the function $f_{\boldsymbol{\theta}}$ can be broken into simpler terms $f_{\boldsymbol{\theta}}(X_q)=\sum_{y=1}^{c}f_{\boldsymbol{\theta}}(X_q;y)$, where

$$f_{\boldsymbol{\theta}}(X_q;y)\ =\ \sum_{\mathbf{x}\in X_p-X_q}\frac{-1}{\delta\cdot\|v_{\boldsymbol{\theta}}(\mathbf{x},y)\|^{-2}+\sum_{\mathbf{x}'\in X_q}g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')},\quad\forall X_q\subseteq X_p.\tag{93}$$

Therefore, in order to prove submodularity and monotonicity of $f_{\boldsymbol{\theta}}$, it suffices to prove these properties for $f_{\boldsymbol{\theta}}(\cdot;y)$ for all $y\in\{1,...,c\}$. Fix y and take any subset $X_q\subseteq X_p$ and $\boldsymbol{\xi}\in X_p-X_q$. Then, we can write

$$\begin{aligned}f_{\boldsymbol{\theta}}(X_q\cup\{\boldsymbol{\xi}\};y) &= \sum_{\mathbf{x}\in X_p-(X_q\cup\{\boldsymbol{\xi}\})}\frac{-1}{\delta\cdot\|v_{\boldsymbol{\theta}}(\mathbf{x},y)\|^{-2}+\sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}}g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')}\\ &= \sum_{\mathbf{x}\in X_p-X_q}\frac{-1}{\delta\cdot\|v_{\boldsymbol{\theta}}(\mathbf{x},y)\|^{-2}+\sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}}g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')}\\ &\quad+\ \frac{1}{\delta\cdot\|v_{\boldsymbol{\theta}}(\boldsymbol{\xi},y)\|^{-2}+\sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}}g_{\boldsymbol{\theta}}(\boldsymbol{\xi},y,\mathbf{x}')}.\end{aligned}$$

We then form the discrete derivative of $f_{\boldsymbol{\theta}}(\cdot;y)$ at X_q to get

$$\begin{aligned}\rho_{f_{\boldsymbol{\theta}}(\cdot;y)}(X_q;\boldsymbol{\xi}) &= f_{\boldsymbol{\theta}}(X_q\cup\{\boldsymbol{\xi}\};y)\ -\ f_{\boldsymbol{\theta}}(X_q;y)\\ &= \sum_{\mathbf{x}\in X_p-X_q}\left[\frac{-1}{\frac{\delta}{\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}+\sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}}g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')}+\frac{1}{\frac{\delta}{\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}+\sum_{\mathbf{x}'\in X_q}g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')}\right]\\ &\quad+\ \frac{1}{\frac{\delta}{\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}+\sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}}g_{\boldsymbol{\theta}}(\boldsymbol{\xi},y,\mathbf{x}')}.\end{aligned}$$

The right-hand-side can be rewritten as

$$\begin{aligned}&\sum_{\mathbf{x}\in X_p-X_q}\left[\frac{g_{\boldsymbol{\theta}}(\mathbf{x},y,\boldsymbol{\xi})}{\left(\frac{\delta}{\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}+\sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}}g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)\left(\frac{\delta}{\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}+\sum_{\mathbf{x}'\in X_q}g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)}\right]\\ &\quad+\ \frac{1}{\frac{\delta}{\|\,\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}+\sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}}g_{\boldsymbol{\theta}}(\boldsymbol{\xi},y,\mathbf{x}')}.\end{aligned}\tag{94}$$

Since by definition $g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\geq 0,\forall\,\mathbf{x},y,\mathbf{x}'$, all of the terms in (94) are non-negative and therefore $\rho_{f_{\boldsymbol{\theta}}(\cdot;y)}(X_q;\boldsymbol{\xi})\geq 0$. This is true for any $X_q\subseteq X_p$, hence monotonicity of $f_{\boldsymbol{\theta}}(\cdot;y)$ is obtained. Now let us take any superset $X_{q'}$ such that $X_q\subseteq X_{q'}\subseteq X_p$ and $\boldsymbol{\xi}\in X_p-X_{q'}$, and form the difference between their corresponding discrete derivatives. From (94), we will

have

$$\begin{aligned}
&\rho_{f_{\boldsymbol{\theta}}(\cdot;y)}(X_q;\boldsymbol{\xi}) - \rho_{f_{\boldsymbol{\theta}}(\cdot;y)}(X_{q'};\boldsymbol{\xi}) \\
&= \sum_{\mathbf{x}\in X_p - X_q} \left[\frac{g_{\boldsymbol{\theta}}(\mathbf{x},y,\boldsymbol{\xi})}{\left(\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2} + \sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)\left(\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2} + \sum_{\mathbf{x}'\in X_q} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)} \right] \\
&+ \frac{1}{\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\boldsymbol{\xi},y)\|^2} + \sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\boldsymbol{\xi},y,\mathbf{x}')} \\
&- \sum_{\mathbf{x}\in X_p - X_{q'}} \left[\frac{g_{\boldsymbol{\theta}}(\mathbf{x},y,\boldsymbol{\xi})}{\left(\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2} + \sum_{\mathbf{x}'\in X_{q'}\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)\left(\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2} + \sum_{\mathbf{x}'\in X_{q'}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)} \right] \\
&- \frac{1}{\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\boldsymbol{\xi},y)\|^2} + \sum_{\mathbf{x}'\in X_{q'}\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\boldsymbol{\xi},y,\mathbf{x}')}. \qquad (95)
\end{aligned}$$

From non-negativity of $g_{\boldsymbol{\theta}}$ and that $X_q \subseteq X_{q'}$, we can conclude that for any $\mathbf{x} \in X$ and $y \in \{1,...,c\}$,

$$\begin{aligned}
&\sum_{\mathbf{x}'\in X_{q'}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}') \;\geq\; \sum_{\mathbf{x}'\in X_q} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}') \\
\Leftrightarrow\quad &\left[\sum_{\mathbf{x}'\in X_{q'}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}') + \frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}\right]^{-1} \;\leq\; \left[\sum_{\mathbf{x}'\in X_q} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}') + \frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}\right]^{-1} \\
\Leftrightarrow\quad &-\left[\sum_{\mathbf{x}'\in X_{q'}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}') + \frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}\right]^{-1} \;\geq\; -\left[\sum_{\mathbf{x}'\in X_q} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}') + \frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}\right]^{-1} \qquad (96)
\end{aligned}$$

Similarly, since $X_q \cup \{\boldsymbol{\xi}\} \subseteq X_{q'} \cup \{\boldsymbol{\xi}\}$ we will get

$$-\left[\sum_{\mathbf{x}'\in X_{q'}\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}') + \frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}\right]^{-1} \geq -\left[\sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}') + \frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2}\right]^{-1} \qquad (97)$$

Applying the inequalities (96) and (97) into euqation (95) results

$$\begin{aligned}
&\rho_{f_{\boldsymbol{\theta}}(\cdot;y)}(X_q;\boldsymbol{\xi}) - \rho_{f_{\boldsymbol{\theta}}(\cdot;y)}(X_{q'};\boldsymbol{\xi}) \\
&\geq \sum_{\mathbf{x}\in X_p - X_q} \left[\frac{g_{\boldsymbol{\theta}}(\mathbf{x},y,\boldsymbol{\xi})}{\left(\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2} + \sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)\left(\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2} + \sum_{\mathbf{x}'\in X_q} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)} \right] \\
&- \sum_{\mathbf{x}\in X_p - X_{q'}} \left[\frac{g_{\boldsymbol{\theta}}(\mathbf{x},y,\boldsymbol{\xi})}{\left(\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2} + \sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)\left(\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2} + \sum_{\mathbf{x}'\in X_q} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)} \right] \\
&+ \frac{1}{\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\boldsymbol{\xi},y)\|^2} + \sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\boldsymbol{\xi},y,\mathbf{x}')} - \frac{1}{\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\boldsymbol{\xi},y)\|^2} + \sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\boldsymbol{\xi},y,\mathbf{x}')}, \qquad (98)
\end{aligned}$$

which yields[12]

$$\sum_{\mathbf{x}\in X_{q'} - X_q} \frac{g_{\boldsymbol{\theta}}(\mathbf{x},y,\boldsymbol{\xi})}{\left(\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2} + \sum_{\mathbf{x}'\in X_q\cup\{\boldsymbol{\xi}\}} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)\left(\frac{\delta}{\|\mathbf{v}_{\boldsymbol{\theta}}(\mathbf{x},y)\|^2} + \sum_{\mathbf{x}'\in X_q} g_{\boldsymbol{\theta}}(\mathbf{x},y,\mathbf{x}')\right)} \;\geq\; 0. \qquad (99)$$

Inequality (99) holds for any $X_q \subseteq X_p$; hence submodularity of $f_{\boldsymbol{\theta}}(\cdot;y)$ stands for all $y \in \{1,...,c\}$ and $\boldsymbol{\theta} \in \Omega$.

References

Pranjal Awasthi, Maria Florina Balcan, and Philip M Long. The power of localization for efficiently learning linear separators with noise. *arXiv preprint arXiv:1307.8371*, 2013.

Javad Azimi, Alan Fern, Xiaoli Zhang-Fern, Glencora Borradaile, and Brent Heeringa. Batch active learning via coordinated matching. *arXiv preprint arXiv:1206.6458*, 2012.

Maria-Florina Balcan, Alina Beygelzimer, and John Langford. Agnostic active learning. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 65–72. ACM, 2006.

Alina Beygelzimer, John Langford, Zhang Tong, and Daniel J Hsu. Agnostic active learning without constraints. In *Advances in Neural Information Processing Systems*, pages 199–207, 2010.

Christopher M Bishop. *Pattern Recognition and Machine Learning*, volume 1. Springer New York, 2006.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Peter J Brockwell and Richard A Davis. *Time Series: Theory and Methods*. Springer, 1991.

Kamalika Chaudhuri, Sham M Kakade, Praneeth Netrapalli, and Sujay Sanghavi. Convergence rates of active learning for maximum likelihood estimation. *arXiv preprint arXiv:1506.02348*, 2015a.

Kamalika Chaudhuri, Sham M Kakade, Praneeth Netrapalli, and Sujay Sanghavi. Convergence rates of active learning for maximum likelihood estimation. In *Advances in Neural Information Processing Systems*, pages 1090–1098, 2015b.

Yuxin Chen and Andreas Krause. Near-optimal batch mode active learning and adaptive submodular optimization. In *Proceedings of The 30th International Conference on Machine Learning*, pages 160–168, 2013.

12. The inequality in (98) is obtained by the fact that, for every four positive real numbers a, a_0, b and b_0, if we have $-a \geq -a_0$ and $-b \geq -b_0$ (similar to equations 96 and 97), then

$$-a \cdot b \;=\; (-a)\cdot b \;\geq\; (-a_0)\cdot b \;=\; a_0 \cdot (-b) \;\geq\; a_0 \cdot (-b_0) \;=\; -a_0 \cdot b_0.$$

David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine learning*, 15(2):201–221, 1994.

David A Cohn. Neural network exploration using optimal experiment design. *Neural Networks*, 9(6):1071–1083, 1996.

David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *arXiv preprint cs/9603104*, 1996.

Sanjoy Dasgupta. Coarse sample complexity bounds for active learning. In *Advances in Neural Information Processing Systems*, pages 235–242, 2005.

Sanjoy Dasgupta, Adam Tauman Kalai, and Claire Monteleoni. Analysis of perceptron-based active learning. In *Learning Theory*, pages 249–263. Springer, 2005.

Sanjoy Dasgupta, Claire Monteleoni, and Daniel J Hsu. A general agnostic active learning algorithm. In *Advances in Neural Information Processing Systems*, pages 353–360, 2007.

Richard O Duda, Peter E Hart, and David G Stork. *Pattern Classification*. John Wiley & Sons,, 1999.

Valerii Vadimovich Fedorov. *Theory of Optimal Experiments*. Elsevier, 1972.

Yoav Freund, H Sebastian Seung, Eli Shamir, and Naftali Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168, 1997.

Yifan Fu, Xingquan Zhu, and Bin Li. A survey on instance selection for active learning. *Knowledge and information systems*, 35(2):249–283, 2013.

Kenji Fukumizu. Statistical active learning in multilayer perceptrons. *IEEE Transactions on Neural Networks*, 11(1):17–26, 2000.

Yuhong Guo. Active instance sampling via matrix partition. In *Advances in Neural Information Processing Systems*, pages 802–810, 2010.

Yuhong Guo and Russell Greiner. Optimistic active-learning using mutual information. In *National Joint Conferences on Artificial Intelligence*, volume 7, pages 823–829, 2007.

Yuhong Guo and Dale Schuurmans. Discriminative batch mode active learning. In *Advances in Neural Information Processing Systems*, pages 593–600, 2008.

Steve Hanneke. Activized learning: Transforming passive to active with improved label complexity. *Journal of Machine Learning Research*, 13(May):1469–1587, 2012.

Steve Hanneke et al. Rates of convergence in active learning. *The Annals of Statistics*, 39 (1):333–361, 2011.

Steven CH Hoi, Rong Jin, Jianke Zhu, and Michael R Lyu. Batch mode active learning and its application to medical image classification. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 417–424. ACM, 2006.

Steven CH Hoi, Rong Jin, and Michael R Lyu. Batch mode active learning with applications to text categorization and image retrieval. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1233–1248, 2009.

Alex Holub, Pietro Perona, and Michael C Burl. Entropy-based active learning for object recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Workshop on Online Learning for Classification*, pages 1–8, 2008.

Ming Ji and Jiawei Han. A variance minimization criterion to active learning on graphs. In *AISTATS*, pages 556–564, 2012.

Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *The Journal of Machine Learning Research*, 9:235–284, 2008.

Erich Leo Lehmann and George Casella. *Theory of Point Estimation*. Springer, 1998.

David MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992.

Noboru Murata, Shuji Yoshizawa, and S-I Amari. Network information criterion-determining the number of hidden units for an artificial neural network model. *IEEE Transactions on Neural Networks*, 5(6):865–872, 1994.

George L Nemhauser and Leonard A Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of Operations Research*, 3(3):177–188, 1978.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1): 265–294, 1978.

Hieu T Nguyen and Arnold Smeulders. Active learning using pre-clustering. In *Proceedings of the 21st International Conference on Machine Learning*, page 79. ACM, 2004.

Andrew I Schein and Lyle H Ungar. Active learning for logistic regression: an evaluation. *Machine Learning*, 68(3):235–265, 2007.

Greg Schohn and David Cohn. Less is more: Active learning with support vector machines. In *International Conference on Machine Learning*, pages 839–846, 2000.

Robert J Serfling. *Approximation Theorems of Mathematical Statistics*. John Wiley & Sons, 2002.

Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012. doi: 10.2200/S00429ED1V01Y201207AIM018.

Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079, 2008.

Burr Settles, Mark Craven, and Soumya Ray. Multiple-instance active learning. In *Advances in Neural Information Processing Systems*, pages 1289–1296, 2008.

Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244, 2000.

Jamshid Sourati, Murat Akcakaya, Jennifer G Dy, Todd K Leen, and Deniz Erdogmus. Classification active learning based on mutual information. *Entropy*, 18(2):51, 2016.

Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2:45–66, 2002.

Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM review*, 38(1): 49–95, 1996.

Larry Wasserman. *All of Statistics: A Concise Course in Statistical Inference*. Springer, 2004.

Xiaojing Yang. A matrix trace inequality. *Journal of Mathematical Analysis and Applications*, 250(1):372–374, 2000.

Chicheng Zhang and Kamalika Chaudhuri. Beyond disagreement-based agnostic active learning. In *Advances in Neural Information Processing Systems*, pages 442–450, 2014.

Tong Zhang and F Oles. The value of unlabeled data for classification problems. In *Proceedings of the 17th International Conference on Machine Learning*, pages 1191–1198, 2000.

Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *International Conference on Machine Learning, Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, pages 58–65, 2003.

Journal of Machine Learning Research 18 (2017) 1-39 Submitted 9/15; Revised 3/17; Published 4/17

A Spectral Algorithm for Inference in Hidden semi-Markov Models

Igor Melnyk IGOR.MELNYK@IBM.COM
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA

Arindam Banerjee BANERJEE@CS.UMN.EDU
Department of Computer Science and Engineering
University of Minnesota
Minneapolis, MN 55414, USA

Editor: Le Song

Abstract

Hidden semi-Markov models (HSMMs) are latent variable models which allow latent state persistence and can be viewed as a generalization of the popular hidden Markov models (HMMs). In this paper, we introduce a novel spectral algorithm to perform inference in HSMMs. Unlike expectation maximization (EM), our approach correctly estimates the probability of given observation sequence based on a set of training sequences. Our approach is based on estimating moments from the sample, whose number of dimensions depends only logarithmically on the maximum length of the hidden state persistence. Moreover, the algorithm requires only a few matrix inversions and is therefore computationally efficient. Empirical evaluations on synthetic and real data demonstrate the advantage of the algorithm over EM in terms of speed and accuracy, especially for large data sets.

Keywords: Graphical models, hidden semi-Markov model, spectral algorithm, tensor analysis, aviation safety

1. Introduction

Hidden semi-Markov models (HSMMs) are discrete latent variable models which allow temporal persistence of latent states, and can be viewed as a generalization of the popular hidden Markov models (HMMs) (Chiappa, 2014; Murphy, 2002; Yu, 2010). In HSMMs, the stochastic model for the unobservable process is defined by a semi-Markov chain: latent state at the next time step is determined by the current latent state as well as time elapsed since the entry into the current state. The ability to flexibly model such latent state persistence turns out to be useful in many application areas, including anomaly detection (Tan and Xi, 2008; Xie and Yu, 2009), activity recognition (van Kasteren et al., 2010), and speech synthesis (Zen et al., 2007). Such state persistence is in contrast to HMMs, which use a Markov chain over latent state transitions and hence have an implicit geometric distribution for the state duration (Rabiner, 1989).

Given a set of training sequences, one can formulate two distinct but related problems: *learning*, i.e., estimating model parameters and *inference*, i.e., computing the probability of an observed and/or latent variable sequence. The methods proposed for learning HSMMs usually follow the initial idea due to Rabiner (Rabiner, 1989) based on the modifications of the Baum-Welch algorithm (Baum and Petrie, 1966), which are all variants of the expectation maximization (EM) framework, presented in (Dempster et al., 1977). Once the parameters are estimated, we can then perform inference using, e.g., the forward-backward algorithm of (Yu and Kobayashi, 2003). However, since EM, in general, has no global guarantees in estimating the parameters correctly and can suffer from slow convergence, such methods can be inefficient and/or inconsistent.

Bayesian nonparametric approaches based on hierarchical Dirichlet processes have also been proposed for HMMs (Fox et al., 2008) and HSMMs (Johnson and Willsky, 2013). Such models avoid the need to specify the number of latent states and can, in principle, learn it from data. However, in practice, inference algorithms for such models are often sensitive to initialization and may suffer from slow convergence.

In recent years, there has been an increased interest in spectral algorithms, which provide computationally efficient, local-minimum-free, provably consistent inference and/or parameter estimation algorithms for latent variable models. For example, (Anandkumar et al., 2013a, 2014b, 2013c) have proposed spectral methods for learning the parameters of a wide class of tree-structured latent graphical models, including Gaussian mixture models, topic models, and latent Dirichlet allocation. The main idea is based on a tensor decomposition of certain low order moments, computable directly from data, in order to extract the model parameters.

In many problems, however, the end goal is not the recovery of model parameters but statistical inference, i.e., computing the probability of a given test sequence, which may be doable without estimating the canonical model parameters. In this regard, (Hsu et al., 2012) have proposed an efficient spectral algorithm for inference in HMMs. It is based on the idea of expressing the probability of the observed sequence in a representation which does not depend on the model parameters and uses easily computable second and third order sample moments to perform inference. Although their work has been used in models on sequences and trees used in Natural Language Processing (NLP) and Reinforcement Learning (RL) (Boots and Gordon, 2010; Dhillon et al., 2011; Balle et al., 2011; Cohen et al., 2014), their approach is not easily extendable to general latent variable models. The work of (Parikh et al., 2011), on the other hand, introduced a spectral algorithm to perform inference in latent tree graphical models with arbitrary topology, and later in (Parikh et al., 2012) a general spectral inference framework for latent junction trees.

In this paper, we utilize the framework of (Parikh et al., 2012) and introduce a novel spectral algorithm for inference in HSMMs. Since we address a more specific problem than (Parikh et al., 2012), our results shed more light into the details of the spectral framework for HSMMs, allow for a sharper analysis, and yield a significantly more efficient algorithm than the general framework in (Parikh et al., 2012). There are two main technical contributions in this work:

- By exploiting the *homogeneity* of HSMMs we make our proposed algorithm more efficient and accurate than the algorithm which directly follows from the recipe in (Parikh et al., 2012) for general graphs. In particular, our approach ensures that during the training phase the number of matrix multiplications and inverses is fixed and independent of the sequence length of the observations.

- Through careful analysis we show that the number of dimensions in the sample moments (represented as a multidimensional matrix or a tensor) in estimated observable representation depends only *logarithmically* on the maximum length of latent state persistence (this is in contrast to a standard implementation, which would have a linear dependence).

©2017 Igor Melnyk and Arindam Banerjee.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-468.html.

In experiments, comparing our method with EM on both synthetic and real data sets, two observations stand out: (i) the spectral method gets similar or better performance than EM as the number of samples increases, and (ii) the spectral method is orders of magnitude faster than EM for the datasets we consider.

Few remarks are in order about the proposed algorithm. Note that our method does not estimate model parameters explicitly but rather learns alternative representation to perform inference on observable variables. The idea of the observable representations was first introduced with the name 'observable operators models' by (Jaeger, 2000) in the context of constructing learning algorithm for the identification of linearly dependent processes. Our formulation cannot be directly used to infer hidden states, although methods such as in (Mossel and Roch, 2005) can be potentially utilized to recover original HSMM parameters from the learned representation. Finally, we note that the similar ideas of using homogeneity of HMMs to improve algorithm's efficiency has also been utilized in other related works, e.g., (Siddiqi et al., 2010; Hsu et al., 2012).

The rest of the paper is organized as follows: We introduce notation in Section 2. In Section 3, we present HSMM inference from a tensor product perspective and in Section 4 introduce the spectral algorithm for inference. In Section 5, we present a careful technical analysis to establish logarithmic dependence of the number of modes in the tensor on maximum latent state persistence. We present experimental results in Section 6 and conclude in Section 7.

2. Notation and Preliminaries

In this section, we cover basic facts about tensor algebra. Detailed tutorials on tensors can be found in (Kiers, 2000) or (Kolda and Bader, 2009). A tensor is defined as a multidimensional array of data, which will be denoted by boldface Euler script letters, e.g., $\underset{m_1,\ldots,m_N}{\mathcal{X}} \in \mathbb{R}^{I_{m_1}\times\cdots\times I_{m_N}}$, which is N-mode tensor of dimensions $I_{m_1}\times\cdots\times I_{m_N}$. A specific mode is denoted by the subscript variable m_i, whose dimension is I_{m_i}.

Any tensor can be matrisized (or flattened) into a matrix. This mapping can be done in multiple ways, the only requirement is that the number of elements is preserved and the mapping is one-to-one. If we split the modes into two disjoint sets, one corresponding to rows and the other to columns, e.g., $\{m_1,\ldots,m_N\}=\{p_1,\ldots,p_K\}\cup\{q_1,\ldots,q_L\}$, then a matrisization of $\mathcal{X}$ is denoted by a corresponding capital boldface letter, e.g., $\underset{p_1,\ldots,p_K q_1,\ldots,q_L}{\mathbf{X}} \in \mathbb{R}^{I_{p_1}\cdots I_{p_K}\times I_{q_1}\cdots I_{q_L}}$.

Tensor Multiplication. Multiplication of two tensors is performed along specific modes. For this, we flatten each tensor to a matrix, perform the usual matrix multiplication and transform the result back to a tensor. The multiplication is denoted by a symbol $\times$ with an optional subscript representing the modes along which the operation is performed, e.g.,:

$$\underset{p_1,\ldots,p_K,r_1,\ldots,r_M}{\mathcal{Z}} = \underset{p_1,\ldots,p_K,q_1,\ldots,q_L}{\mathcal{X}} \times_{q_1,\ldots,q_L} \underset{q_1,\ldots,q_L,r_1,\ldots,r_M}{\mathcal{Y}},$$

where $\underset{q_1,\ldots,q_L,r_1,\ldots,r_M}{\mathcal{Y}} \in \mathbb{R}^{I_{q_1}\times\cdots\times I_{q_L}\times I_{r_1}\times\cdots\times I_{r_M}}$ and the resulting tensor on the left hand side is of the form $\underset{p_1,\ldots,p_K,r_1,\ldots,r_M}{\mathcal{Z}} \in \mathbb{R}^{I_{p_1}\times\cdots\times I_{p_K}\times I_{r_1}\times\cdots\times I_{r_M}}$. Observe that in the above, we can flatten the tensors $\mathcal{X}$ and $\mathcal{Y}$ in multiple different ways as long as the matrix multiplication remains valid. For example, we could assign the multiplication modes in both tensors to columns, in this case the matrix product becomes $\mathbf{Z}=\mathbf{X}\mathbf{Y}^T$. Alternatively, the tensor $\mathcal{Y}$ could be matrisized with the multiplication modes corresponding to rows, resulting in the product $\mathbf{Z}=\mathbf{X}\mathbf{Y}$.

$\underset{p,q,r}{\mathcal{X}} \in \mathbb{R}^{I_p\times I_q\times I_r}$	N-mode tensor of dimensions $I_p\times I_q\times I_r$
$\underset{p,q,r}{\mathbf{X}} \in \mathbb{R}^{I_pI_q\times I_r}$	Matricization of tensor $\underset{p,q,r}{\mathcal{X}}$ with I_pI_q rows and I_r columns
$\underset{p,q,r}{\mathcal{X}} \times_r \underset{r,s,t}{\mathcal{Y}}$	Multiplication of tensor $\underset{p,q,r}{\mathcal{X}}$ and tensor $\underset{r,s,t}{\mathcal{Y}}$ along mode r
$\underset{p,q,r}{\mathcal{X}} \times_{q,r} \underset{p,q,r}{\mathcal{X}}^{-1} = \underset{p,p}{\mathcal{I}} = \underset{p}{\mathcal{I}}$	Inversion of tensor $\underset{p,q,r}{\mathcal{X}}$ with respect to modes q and r
$\mathbb{X}_t$	Representation of a clique in a Junction tree
$o_t \in \{1,\ldots,n_o\}$	Observation variable in HSMM
$x_t \in \{1,\ldots,n_x\}$	Latent state variable in HSMM
$d_t \in \{1,\ldots,n_d\}$	Latent duration variable in HSMM
$\mathbf{O}_{R_t} := \{o_{t+1}, o_{t+2},\ldots\}$	Set of observations to the right of time step t
$\mathbf{O}_{L_t} := \{\ldots, o_{t-2}, o_{t-1}\}$	Set of observations to the left of time step t

Table 1: Summary of some of the key notations used throughout the paper.

An important fact about tensor multiplication is that in a series of tensor multiplications the order is irrelevant (i.e., it is an associative operation) as long as the multiplication is performed along the matching modes, e.g,

$$\underset{sp}{\mathcal{X}} \times_s \left(\underset{tr}{\mathcal{Y}} \times_r \underset{rs}{\mathcal{Z}}\right) = \left(\underset{sp}{\mathcal{X}} \times_s \underset{rs}{\mathcal{Z}}\right) \times_r \underset{tr}{\mathcal{Y}}.$$

If we let the matrisized tensors to be $\mathbf{X}\in\mathbb{R}^{I_p\times I_s}$, $\mathbf{Y}\in\mathbb{R}^{I_t\times I_r}$ and $\mathbf{Z}\in\mathbb{R}^{I_r\times I_s}$, then the above can be verified to be true since

$$\mathbf{X}\left(\mathbf{Y}\mathbf{Z}\right)^T = \left(\mathbf{X}\mathbf{Z}^T\right)\mathbf{Y}^T.$$

To reduce clutter, in many places we will drop the multiplication subscripts. The implied modes of multiplication can then be inferred from the subscripts of the tensors. Specifically, when two tensors are multiplied, we first check their modes and then multiply along the modes which are common to both of them. For example, in the product $\underset{pqr}{\mathcal{X}} \times \underset{qsr}{\mathcal{Y}}$, the implied multiplication is performed along the common modes, i.e., q and r.

Tensor Inversion. We also discuss the operation of tensor inversion. Tensor inverse $\mathcal{X}^{-1}$ is always defined with respect to a certain subset of modes and can be written as follows:

$$\underset{p_1,\ldots,p_K,q_1,\ldots,q_L}{\mathcal{X}} \times_{q_1,\ldots,q_L} \underset{p_1,\ldots,p_K,q_1,\ldots,q_L}{\mathcal{X}^{-1}} = \underset{p_1,\ldots,p_K,p_1,\ldots,p_K}{\mathcal{I}},$$

where the inversion is performed along the modes $q_1,\ldots,q_L$, and $\underset{p_1,\ldots,p_K,p_1,\ldots,p_K}{\mathcal{I}}$ denotes an identity tensor, whose elements are everywhere zero, except $\mathcal{I}(i_1,\ldots,i_K,i_1,\ldots,i_K)=1$. To perform inversion, we first convert tensor to a matrix, i.e., matrisized tensor. If the modes to be inverted along are associated with columns of the matrix, we compute the right matrix inverse, so that these modes get eliminated after the product. Otherwise, if those modes associated with rows, we compute left matrix inverse. Obviously, for the full rank square matrices both choices would produce the same result. For example, in the above equation the matrisized tensor might be of the form $\underset{p_1,\ldots,p_K q_1,\ldots,q_L}{\mathbf{X}} \in \mathbb{R}^{I_{p_1}\cdots I_{p_K}\times I_{q_1}\cdots I_{q_L}}$, therefore, we would compute the right matrix inverse so that

the modes $q_1, \ldots, q_L$ are eliminated. If the matrisized $\mathbf{X}$ has full row rank, then the inverse can be computed, otherwise we could only compute its pseudo-inverse. Tensorizing the matrix $\mathbf{X}^{-1}$ gives us the desired tensor inverse.

Mode Duplication. Observe that in the above, the tensor $\underset{p_1,\ldots,p_K,p_1,\ldots,p_K}{\mathcal{I}}$ has duplicate modes. In general, if a tensor has duplicate modes, the corresponding sub-tensor can be interpreted as a hyper-diagonal. For example, if for a tensor $\underset{pq}{\mathcal{X}}$ we construct a tensor $\underset{pppq}{\overline{\mathcal{X}}}$, which has its mode p duplicated three times, then for a fixed index i, the sub-tensor $\overline{\mathcal{X}}(:,:,:,i)$ is a hypercube with elements $\mathcal{X}(:,i)$ on the diagonal.

Mode duplication enables us to multiply several tensors along the same mode. For example, if we need to multiply tensors $\underset{sp}{\mathcal{X}}$, $\underset{pr}{\mathcal{Y}}$ and $\underset{tp}{\mathcal{Z}}$ along the mode p, then a simple product of the form

$$\underset{sp}{\mathcal{X}} \times_p \underset{pr}{\mathcal{Y}} \times_p \underset{tp}{\mathcal{Z}}$$

cannot be done since any product of two tensors along the mode p would eliminate it, preventing any further multiplications. In general, if there are N multiplications along the specific mode, then there are must be cumulatively $2N$ number of times such a mode is encountered in the participating tensors. In our example, we might duplicate the mode p in, say, tensor $\mathcal{Z}$ to have

$$\underset{sp}{\mathcal{X}} \times_p \left(\underset{pr}{\mathcal{Y}} \times_p \underset{tpp}{\mathcal{Z}} \right) = \underset{sp}{\mathcal{X}} \times_p \underset{prt}{\mathcal{W}} = \underset{srt}{\mathcal{V}},$$

so that there are two multiplications over mode p and cumulatively there are four times such a mode is encountered in the participating tensors. To reduce clutter, we sometimes do not explicitly show the duplicated variables in the subscripts; the implied mode repetition will be evident from the context or explicitly stated in cases when there is a confusion. For example, the identity tensor will often be written as $\underset{p_1,\ldots,p_K}{\mathcal{I}}$.

Tensor rank Finally, we discuss the meaning of a tensor rank. A tensor can have multiple ranks and each of them is defined based on the rank of a particular matricization. For example, consider a tensor $\underset{pqs}{\mathcal{X}}$. If we flatten it to a matrix $\mathbf{X} \in \mathbb{R}^{I_p \times I_q I_s}$ then it can have a rank r_1. On the other hand, a matricization of the form $\mathbf{X} \in \mathbb{R}^{I_p I_q \times I_s}$ can have a rank r_2, and so on. In our derivations, the particular rank we are referring to will be evident from the context.
In Table 2 we summarized some of the key notations used throughout the paper.

3. Problem Formulation

In this paper, we consider the problem of inference in HSMM[1] (see Figure 1). Unlike the popular HMM, which has a geometric probability for state persistence, i.e., the probability of persisting in the same state over t time steps decreases as π^t, where π is the probability of persistence for one time step, HSMM explicitly models state persistence. From a graphical model perspective, HSMM has three sets of variables: the observations $o_t \in \{1, \ldots, n_o\}$, the latent states $x_t \in \{1, \ldots, n_x\}$, and another latent variable $d_t \in \{1, \ldots, n_d\}$ which determines the length of state persistence. HSMM

1. To reduce clutter, in the main paper we only consider the model for a general time stamp t and ignore the initial ($t = 0$) and final ($t = T$) steps of the model, whose representation differs slightly from what is shown in Figure 1. The details for these parts are presented separately in Appendix B.

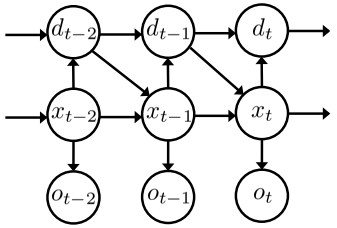


Figure 1: Hidden Semi-Markov Model (HSMM) depicted as a dynamic Bayesian network. Here $o_t \in \{1, \ldots, n_o\}$ denotes an observation at time step t, $x_t \in \{1, \ldots, n_x\}$ is a latent state and $d_t \in \{1, \ldots, n_d\}$ is the length of state persistence at time step t.

is specified by three conditional probability tables (CPTs): the observation/emission probability $p(o_t|x_t)$ and the state transition and the duration probabilities given by

$$p(d_t|x_t, d_{t-1}) = \begin{cases} p(d_t|x_t) & \text{if } d_{t-1} = 1 \\ \delta(d_t, d_{t-1} - 1) & \text{if } d_{t-1} > 1 \,, \end{cases} \tag{1}$$

$$p(x_t|x_{t-1}, d_{t-1}) = \begin{cases} p(x_t|x_{t-1}) & \text{if } d_{t-1} = 1 \\ \delta(x_t, x_{t-1}) & \text{if } d_{t-1} > 1 \,, \end{cases} \tag{2}$$

where $\delta(a, b)$ denotes the Dirac delta function: $\delta(a, b) = 1$ if $a = b$ and 0 otherwise. In addition, one can consider suitable prior probabilities $p(x_0)$ and $p(d_0)$. In essence, d_t works as a down counter for state persistence. When $d_{t-1} > 1$, the model remains in the same state $x_t = x_{t-1}$, while when $d_{t-1} = 1$, one samples a new state x_t and the new duration in that state $d_t|x_t$. For our analysis, we assume $p(d_t|x_t, d_{t-1} = 1)$ to be a discrete distribution over $\{1, \ldots, n_d\}$ where n_d denotes the largest duration of state persistence.

The considered inference problem can be posed as follows: given a set of discrete sequences $\{\mathbf{S}^1, \ldots, \mathbf{S}^N\}$ drawn independently from the HSMM model, where each sequence is defined as $\mathbf{S}^i = \{o_1^i, \ldots, o_{T_i}^i\}, i = 1, \ldots, N$, our goal is to compute the probability $p(\mathbf{S}^{test})$ of any given test sequence $\mathbf{S}^{test} = (o_1^{test}, \ldots, o_T^{test})$. A traditional approach would be to estimate the CPTs using the EM algorithm, and use the estimates to compute $p(\mathbf{S}^{test})$. However, the EM algorithm is not guaranteed to estimate the parameters optimally, and hence the computation of $p(\mathbf{S}^{test})$ may be incorrect. The focus of our work is to develop a provably correct spectral algorithm for computing the probability $p(\mathbf{S}^{test})$.

3.1 HSMM in Tensor Notations

We start by considering the matrix forms of the HSMM parameters and writing the computations in tensor notation, as introduced in Section 2. Specifically, $p(d_t|x_t, d_{t-1} = 1)$ is denoted as $D \in \mathbb{R}^{n_d \times n_x}$, $p(x_t|x_{t-1}, d_{t-1} = 1)$ is denoted as $X \in \mathbb{R}^{n_x \times n_x}$, and $p(o_t|x_t)$ as $O \in \mathbb{R}^{n_o \times n_x}$. We make the following assumptions on the HSMM parameters:

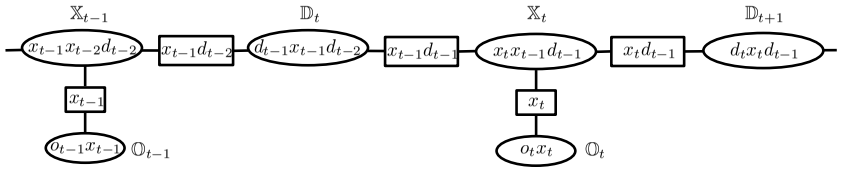


Figure 2: Junction Tree for Hidden Semi-Markov Model. The ovals represent cliques, which are denoted by capital blackboard bold variables; the rectangles denote separators. Symbols within the shapes represent the variables on which the corresponding potentials depend.

Assumptions

$A1$. X is full rank and has non-zero probability of visiting any state from any other state.

$A2$. D has a non-zero probability of any duration in any state.

$A3$. O is full column rank and, as a consequence, $n_x \leq n_o$.

We provide some comments on the above assumptions. We note that the assumption $A1$ can be relaxed to allow zero entries (while still ensuring full rank structure) and thus prevent certain states to be directly reachable from other states; however, this would require more involved analysis based on the mixing time of the corresponding Markov chain (Levin et al., 2009), and is not pursued in this work. Also, observe that the assumption of $n_x \leq n_o$ is needed in order to ensure that hidden states are identifiable, although recent work is showing that such an assumption can be relaxed in some cases (Bailly et al., 2009; Anandkumar et al., 2013b). Intuitively, it means that the number of different observations coming from each state is large enough, so that one hidden state can be differentiated from the other.

To express the joint probability $p(o_1, \ldots, o_T)$ for any possible observation sequence in tensor form, we utilize the junction tree algorithm (Barber, 2012). The resulting tree is shown in Figure 2 and it corresponds to the graphical model of HSMM in Figure 1. Recall, that the junction tree is a tree-structured representation of an arbitrary graph enabling efficient inference. It can be constructed by forming a maximal spanning tree from the cliques of the graph. The cliques then represent vertices in the junction tree and the edges connecting the vertices are labeled with variables common to two cliques it connects. The set of variables on the edges are referred to as separators. For example, in Figure 2 the cliques $\mathbb{X}_t$ and $\mathbb{D}_t$ have two variables in common, x_{t-1} and d_{t-1}, and which define the sepatator between $\mathbb{X}_t$ and $\mathbb{D}_t$.

We proceed by representing the clique CPTs of the junction tree as tensors. For example, the clique $\mathbb{X}_t$, containing the CPT of $p(x_t|x_{t-1}, d_{t-1})$ is represented as tensor $\underset{x_t|x_{t-1}d_{t-1}}{\mathcal{X}}$. For ease of exposition, the tensor's modes are named based on the variables on which the tensor depends. We also keep the conditioning symbol $|$ for clarity. Similarly, we represent the clique $\mathbb{D}_t$ with its CPT $p(d_t|x_t, d_{t-1})$ as tensor $\underset{d_t|x_td_{t-1}}{\mathcal{D}}$, and $\mathbb{O}_t$ containing $p(o_t|x_t)$ as tensor $\underset{o_t|x_t}{\mathcal{O}}$.

If we denote the joint probability of the observed sequence $p(o_1, \ldots, o_T)$ as $\underset{o_1,..,o_T}{\mathcal{P}}$ then the message passing for the junction tree algorithm in Figure 2 can be represented as tensor multiplications:

$$\underset{o_1,\ldots,o_T}{\mathcal{P}} = \prod_t \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_tx_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{o_t|x_t}{\mathcal{O}} \right), \tag{3}$$

where, for simplicity, we denoted by $\prod_t$ the tensor product over multiple time steps.

Note that in (3) the neighboring tensors are multiplied along the modes which are the separator variables between two corresponding neighboring cliques in Figure 2. Therefore, as we discussed in Section 2, if a certain mode of a tensor is to participate multiple times in products with other tensor, the mode must be duplicated for the expression to remain correct. It can easily be seen from the junction tree that the number of times the mode is duplicated depends on the number of times such a variable appears in separators adjacent to the clique. For example, the tensor $\underset{x_tx_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}}$ has a mode x_{t-1} appearing once in the separator connecting $\mathbb{X}_t$ and $\mathbb{D}_t$ in Figure 2, while x_t appears a total of two times, once in the separator connecting $\mathbb{X}_t$ and $\mathbb{O}_t$, and once in the separator connecting $\mathbb{X}_t$ and $\mathbb{D}_{t+1}$. Finally, d_{t-1} appears in the separator between $\mathbb{D}_t$ and $\mathbb{X}_t$, and between $\mathbb{D}_{t+1}$ and $\mathbb{X}_t$. Applying the same reasoning to tensors $\mathcal{D}$ and $\mathcal{O}$ results in the expression (3).

3.2 Summary of Technical Results

In this work, we represent expression (3), which is defined in terms of unknown model parameters, in a different *observable form*, where all the factors can be estimated directly from data using certain sample moments without knowledge of model parameters. Such an observable form is derived in Sections 4.1 and 4.2. Based on the observable form, in Section 4.3 we propose a simple spectral algorithm which requires estimating $\mathcal{X}$, $\mathcal{D}$ and $\mathcal{O}$ for all the time stamps t. This estimation process is expensive as it involves costly tensor operations to be performed at each time index t. Moreover, the accurate estimation of these tensors requires large number of training sequences which might not be available, leading to inaccurate and unstable computations. However, exploiting the homogeneity property of HSMMs, i.e., the probability distributions represented by the above tensors are the same across all time t, we derive a computationally efficient and accurate spectral algorithm in Section 4.4 which requires the estimation of only three tensors for all the time stamps t. Although the computational complexity of the inference, i.e., the evaluation of expression (3), is not affected by the introduced modifications, the overall algorithm becomes faster and more accurate.

In Section 5 we return to the results of Sections 4.1 and establish the conditions under which the derived observable form can be computed from data. In particular, our analysis shows that the number of dimensions of the required sample moments (in the form of tensors, estimated from data and representing the co-occurrence frequency of certain observable variables), has logarithmic dependence on the longest state persistence n_d. Such conclusion is in contrast to the analysis, which would follow from the work of (Parikh et al., 2012), in which case the required number of dimensions in the estimated sample moments would have had linear dependence on n_d. The exponential reduction in the size of the estimated tensors represents significant improvement in algorithm's efficiency and accuracy since the multidimensional matrices are of smaller size and consequently more data is available to estimate each of its entry.

4. Spectral Algorithm for Inference in HSMM

In this Section we present the details of the spectral inference approach. In particular, in Sections 4.1 and 4.2 we derive observable tensor representation and show how to estimate each of its factors directly from data. Practical algorithms implementing these ideas are then derived in Sections 4.3 and 4.4.

4.1 Observable Tensor Representation

Observe that the computation of the joint probability in (3) requires knowledge of the unknown model parameters. Our goal is to change the tensor representation such that $\underset{o_1,\ldots,o_T}{\mathcal{P}}$ can be written in terms of the quantities directly computable from data. To that end, we follow (Parikh et al., 2012) and between every two factors in (3) introduce an identity tensor with the modes corresponding to the modes along which the multiplication is performed. For example, consider a part of (3) after introducing identity tensors:

$$\times \underset{x_{t-1}d_{t-2}}{\mathcal{I}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{x_{t-1}d_{t-1}}{\mathcal{I}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_t x_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{x_t}{\mathcal{I}} \times_{x_t} \underset{o_t x_t}{\mathcal{O}} \right) \times_{x_t d_{t-1}} \underset{x_t d_{t-1}}{\mathcal{I}} \times, \quad (4)$$

where all the identity tensors have duplicated modes which are not shown.

Now rewrite each of the identity tensors in (4) as a multiplication of some factor times its inverse. For example,

$$\underset{x_t}{\mathcal{I}} = \underset{\omega_{x_t}x_t}{\mathcal{F}} \times_{\omega_{x_t}} \underset{\omega_{x_t}x_t}{\mathcal{F}^{-1}},$$

for some invertible factor $\underset{\omega_{x_t}x_t}{\mathcal{F}}$, whose modes are x_t and ω_{x_t}. Note that the choice of mode x_t is fixed and is determined by the modes of the identity tensor $\underset{x_t}{\mathcal{I}}$, while the mode ω_{x_t} is not fixed and we have freedom in selecting it as convenient. Moreover, observe that since the tensor inversion is done along the mode ω_{x_t} and the matrix $\mathbf{F}$ has its rows associated with mode ω_{x_t}, we need to ensure such a matrix has full column rank for the inverse to exist and for the product $\mathbf{F}^{-1}\mathbf{F}$ to be the identity matrix (see Section 2 for more details on tensor inversion). Based on the above discussion, we choose tensor $\mathcal{F}$ such that (i) ω_{x_t} are the observed variables, (ii) $\underset{\omega_{x_t}x_t}{\mathcal{F}}$ is invertible, i.e., matrix $\mathbf{F}$, whose columns correspond to x_t, has full column rank, and (iii) we interpret the factor $\underset{\omega_{x_t}x_t}{\mathcal{F}}$ as corresponding to a conditional probability distribution, i.e., $p(\omega_{x_t}|x_t)$ and therefore write $\underset{\omega_{x_t}|x_t}{\mathcal{F}}$.

After expanding each of the identity tensors, regrouping the factors and recalling that in a series of tensor multiplication the order is irrelevant, we can identify three modified tensors:

$$\begin{aligned}
\underset{\omega_{x_{t-1}d_{t-2}}\omega_{x_{t-1}d_{t-1}}}{\tilde{\mathcal{D}}} &= \underset{\omega_{x_{t-1}d_{t-2}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{\omega_{x_{t-1}d_{t-1}}|x_{t-1}d_{t-1}}{\mathcal{F}} \\
\underset{\omega_{x_{t-1}d_{t-1}}\omega_{x_t}\omega_{x_t d_{t-1}}}{\tilde{\mathcal{X}}} &= \underset{\omega_{x_{t-1}d_{t-1}}|x_{t-1}d_{t-1}}{\mathcal{F}^{-1}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_t x_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{\omega_{x_t}|x_t}{\mathcal{F}} \right) \times_{x_t d_{t-1}} \underset{\omega_{x_t d_{t-1}}|x_t d_{t-1}}{\mathcal{F}} \\
\underset{\omega_{x_t}o_t}{\tilde{\mathcal{O}}} &= \underset{\omega_{x_t}|x_t}{\mathcal{F}^{-1}} \times_{x_t} \underset{o_t|x_t}{\mathcal{O}}.
\end{aligned}$$

Note that although each of the above tensors depends only on certain observed variables ω, for a concrete algorithm one has to decide what these ω are, and also how to estimate the associated

tensors from data. The right hand side in the above expressions depend on the unknown model parameters, whereas the tensors on the left do not correspond to valid probability distributions (due to the presence of inverses $\mathcal{F}^{-1}$), and so cannot be estimated from data using sample moments. For example, $\underset{\omega_{x_{t-1}d_{t-2}}\omega_{x_{t-1}d_{t-1}}}{\tilde{\mathcal{D}}}$ is not a tensor form of $p(\omega_{x_{t-1}d_{t-2}}, \omega_{x_{t-1}d_{t-1}})$.

Next, we discuss the choice of the observable set ω in the factors $\mathcal{F}$. From Figure 2 we can see that there are three types of separators which depend on $x_{t-1}d_{t-1}$, $x_t d_{t-1}$ and x_t, consequently, there are three types of identity tensors which we introduced in (4), i.e., $\underset{x_{t-1}d_{t-1}}{\mathcal{I}}$, $\underset{x_t d_{t-1}}{\mathcal{I}}$ and $\underset{x_t}{\mathcal{I}}$. Therefore, we need to define three types of observable sets $\omega_{x_{t-1}d_{t-1}}$, $\omega_{x_t d_{t-1}}$ and ω_{x_t}. There are multiple choices for these sets, one of them is $\omega_{x_{t-1}d_{t-1}} = \omega_{x_t d_{t-1}} = \{o_{t+1}, o_{t+2}, \ldots\}$ for all t (see Figure 3 for an illustration). Ideally, we want these sets to be of minimal size, since they need to be estimated from observations. The detailed description of how many and which of these observations to select to get a minimal set is deferred until Section 5, where we also show that we can set $\omega_{x_t} = o_t$.

In what follows, we define $\mathbf{O}_{R_t} := \{o_{t+1}, o_{t+2}, \ldots, o_{t+\tau}\}$ (see Figure 3) to emphasize that this is a fixed set of observations whose length τ is yet to be determined, starting after time stamp t and going to the right (or forward in time) in the graphical model in Figure 1. With these definitions, setting $\omega_{x_{t-1}d_{t-1}} = \mathbf{O}_{R_t}$, $\omega_{x_t d_{t-1}} = \mathbf{O}_{R_t}$, $\omega_{x_{t-1}d_{t-2}} = \mathbf{O}_{R_{t-1}}$ and $\omega_{x_t} = o_t$, we can now rewrite (3) in the form:

$$\underset{o_1,\ldots,o_T}{\mathcal{P}} = \prod_t \underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}} \times_{\mathbf{O}_{R_t}} \left(\underset{\mathbf{O}_{R_t}o_t\mathbf{O}_{R_t}}{\tilde{\mathcal{X}}} \times_{o_t} \underset{o_t o_t}{\tilde{\mathcal{O}}} \right). \quad (5)$$

Comparing (3) and (5) we see that the above equation expresses the joint probability distribution in the observable form. As noted above, we cannot yet use this formula in practice since we do not know how to compute the transformed tensors. In what follows, we show how to estimate such tensors directly from data, without the need for the model parameters.

4.2 Estimation of Observable Tensors

In this Section we express each of the tensors in (5) in forms which can be directly estimated from the observed sequences.

4.2.1 COMPUTATION OF TENSOR $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}}$

Consider the tensor from Section 4.1

$$\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}} = \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}, \quad (6)$$

whose modes are the observable variables $\mathbf{O}_{R_{t-1}}$ and $\mathbf{O}_{R_t}$. To estimate this tensor from data, consider $\mathbf{O}_{L_{t-1}}$, a set of the observed variables such that $\mathbf{O}_{L_{t-1}}$ and $\mathbf{O}_{R_{t-1}}$ are independent, conditioned on $x_{t-1}d_{t-2}$ (see Figure 3):

$$p(\mathbf{O}_{L_{t-1}}, \mathbf{O}_{R_{t-1}}) = \sum_{x_{t-1}d_{t-2}} p(\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}) p(\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}) p(x_{t-1}d_{t-2}). \quad (7)$$

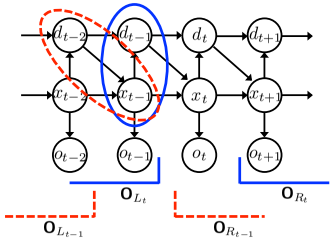


Figure 3: Conditional independence in HSMM. The figure depicts two sets of relationships: $\mathbf{O}_{L_t}$ and $\mathbf{O}_{R_t}$ are independent conditioned on $x_{t-1}d_{t-1}$, similarly, $\mathbf{O}_{L_{t-1}}$ and $\mathbf{O}_{R_{t-1}}$ are conditionally independent given $x_{t-1}d_{t-2}$. We defined $\mathbf{O}_{L_t} = \{\ldots, o_{t-2}, o_{t-1}\}$ and $\mathbf{O}_{R_t} = \{o_{t+1}, o_{t+2}, \ldots\}$.

The above conditional independence relationship can be written in tensor form:

$$\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}} = \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}}, \tag{8}$$

where tensor $\mathcal{K}$ represents the marginal $p(x_{t-1}, d_{t-2})$. Note that, though not shown, the modes x_{t-1} and d_{t-2} need to appear twice in $\mathcal{K}$, since it interacts with both other terms (see the discussion on mode duplication in Section 2). The set $\mathbf{O}_{L_{t-1}}$ is defined in a way similar to $\mathbf{O}_{R_t}$ but with the set of observations starting at time stamp $t-2$ and going to the left (or backward in time), i.e., $\mathbf{O}_{L_{t-1}} := \{\ldots, o_{t-3}, o_{t-2}\}$ (see Figure 3).

Next, we express the inverse of the tensor $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$ from (8) and substitute back to (6). For this, we observe that in (6) the tensor $\mathcal{F}^{-1}$ is inverted with respect to mode $\mathbf{O}_{R_{t-1}}$, therefore, we do the following:

$$\begin{aligned}
\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}} \times_{\mathbf{O}_{R_{t-1}}} \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} &= \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{I}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}} \\
\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} &= \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}^{-1}} \times_{\mathbf{O}_{L_{t-1}}} \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}},
\end{aligned} \tag{9}$$

where $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}^{-1}}$ is inverted with respect to mode $\mathbf{O}_{L_{t-1}}$. Next, substituting (9) back to (6), we get

$$\begin{aligned}
\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}} &= \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}^{-1}} \times_{\mathbf{O}_{L_{t-1}}} \overline{\underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}} \\
&= \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}^{-1}} \times_{\mathbf{O}_{L_{t-1}}} \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathcal{M}},
\end{aligned} \tag{10}$$

where we have eliminated all the latent variables by multiplying the last four terms on the first line.

Observe that the tensors $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}}$ and $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathcal{M}}$ represent valid joint probability distributions over a subset of observations $p(\mathbf{O}_{L_{t-1}}, \mathbf{O}_{R_{t-1}})$ and $p(\mathbf{O}_{L_{t-1}}, \mathbf{O}_{R_t})$, respectively, and though they are

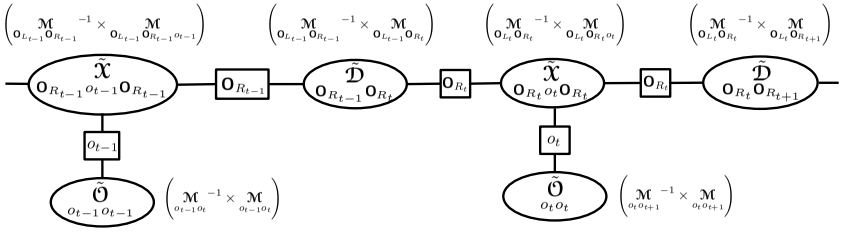


Figure 4: Graphical representation of the HSMM spectral algorithm for inference in Algorithm 1. As compared to junction tree in Figure 2, the cliques and separators are now defined in terms of the tensors, which are defined with respect to the observed data. The expressions in the parenthesis show the observable representation of the corresponding tensors.

defined with respect to unknown model parameters (as, for example, in (7)), we can readily estimate them from data. For example, $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathcal{M}}$ is a tensor, where each entry is computed from the frequency of co-occurrence of tuples of the observations $\{\ldots, o_{t-3}, o_{t-2}, o_{t+1}, o_{t+2}, \ldots\}$. Ideally, we want a small number of observations since we need to estimate their co-occurrence frequency from the training data. A precise characterization of how many and which of these observations suffices for the analysis will be done in Section 5.

4.2.2 COMPUTATION OF TENSOR $\underset{\mathbf{O}_{R_t}o_t\mathbf{O}_{R_t}}{\tilde{\mathcal{X}}}$

The form of this tensor was established at the beginning of Section 4.2 to be:

$$\underset{\mathbf{O}_{R_t}o_t\mathbf{O}_{R_t}}{\tilde{\mathcal{X}}} = \underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}^{-1}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_tx_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{o_t|x_t}{\mathcal{F}}\right) \times_{x_td_{t-1}} \underset{\mathbf{O}_{R_t}|x_td_{t-1}}{\mathcal{F}}. \tag{11}$$

Consider the following conditional independence relationship (see Figure 3):

$$\underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathcal{M}} = \underset{\mathbf{O}_{L_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \times_{x_{t-1}d_{t-1}} \underset{x_{t-1}d_{t-1}}{\mathcal{K}}, \tag{12}$$

where $\underset{x_{t-1}d_{t-1}}{\mathcal{K}} = \underset{x_{t-1}d_{t-1}x_{t-1}d_{t-1}}{\mathcal{K}}$ and we omitted the duplicated modes.

We express the inverse of tensor $\underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}$ from the above equation

$$\underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}^{-1}} = \underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathcal{M}^{-1}} \times_{\mathbf{O}_{L_t}} \underset{\mathbf{O}_{L_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \times_{x_{t-1}d_{t-1}} \underset{x_{t-1}d_{t-1}}{\mathcal{K}},$$

where tensor $\underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}$ is inverted with respect to mode $\mathbf{O}_{R_t}$, while $\underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathcal{M}}$ is inverted with respect to mode $\mathbf{O}_{L_t}$. Substituting back to (11), we get

$$\underset{\mathbf{O}_{R_t}o_t\mathbf{O}_{R_t}}{\tilde{\mathcal{X}}} = \underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathcal{M}^{-1}} \times_{\mathbf{O}_{L_t}} \underset{\mathbf{O}_{L_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \times_{x_{t-1}d_{t-1}} \underset{x_{t-1}d_{t-1}}{\mathcal{K}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_tx_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{o_t|x_t}{\mathcal{F}}\right) \times_{x_td_{t-1}} \underset{\mathbf{O}_{R_t}|x_td_{t-1}}{\mathcal{F}}.$$

Considering the last five factors and multiplying them together, we obtain

$$\underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}o_t}{\mathcal{M}} = \underset{\mathbf{O}_{L_t}|x_{t-1}d_{t-1}}{\mathcal{F}} \times_{x_{t-1}d_{t-1}} \underset{x_{t-1}d_{t-1}}{\mathcal{K}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_tx_t|x_{t-1}d_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{o_t|x_t}{\mathcal{F}} \right) \times_{x_td_{t-1}} \underset{\mathbf{O}_{R_t}|x_td_{t-1}}{\mathcal{F}}.$$

Finally, (11) can now be written as

$$\underset{\mathbf{O}_{R_t}o_t\mathbf{O}_{R_t}}{\tilde{\mathcal{X}}} = \underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathcal{M}}^{-1} \times_{\mathbf{O}_{L_t}} \underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}o_t}{\mathcal{M}}, \tag{13}$$

where the right hand side can now be estimated directly from data, without the need for the model parameters.

4.2.3 COMPUTATION OF TENSOR $\tilde{\mathcal{O}}_{o_to_t}$

Finally, we consider the tensor

$$\underset{o_to_t}{\tilde{\mathcal{O}}} = \underset{o_t|x_t}{\mathcal{F}}^{-1} \times_{x_t} \underset{o_t|x_t}{\mathcal{O}}. \tag{14}$$

The conditional independence relationship can take the form

$$\underset{o_to_{t+1}}{\mathcal{M}} = \underset{o_t|x_t}{\mathcal{F}} \times_{x_t} \underset{o_{t+1}|x_t}{\mathcal{F}} \times_{x_t} \underset{x_t}{\mathcal{K}}.$$

Expressing the inverse of $\underset{o_t|x_t}{\mathcal{F}}$

$$\underset{o_t|x_t}{\mathcal{F}}^{-1} = \underset{o_to_{t+1}}{\mathcal{M}}^{-1} \times_{o_{t+1}} \underset{o_{t+1}|x_t}{\mathcal{F}} \times_{x_t} \underset{x_t}{\mathcal{K}},$$

and substituting in (14), we get

$$\begin{aligned} \underset{o_to_t}{\tilde{\mathcal{O}}} &= \underset{o_to_{t+1}}{\mathcal{M}}^{-1} \times_{o_{t+1}} \underset{o_{t+1}|x_t}{\mathcal{F}} \times_{x_t} \underset{x_t}{\mathcal{K}} \times_{x_t} \underset{o_t|x_t}{\mathcal{O}} \\ &= \underset{o_to_{t+1}}{\mathcal{M}}^{-1} \times_{o_{t+1}} \underset{o_to_{t+1}}{\mathcal{M}}. \end{aligned} \tag{15}$$

4.3 Basic Version of Spectral Algorithm

The basic version of the spectral HSMM algorithm to compute $\underset{o_1,\ldots,o_T}{\mathcal{P}}$ entirely using the observed variables can be described as a two step process: in the learning step, compute tensors $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}}$, $\underset{\mathbf{O}_{R_{t-1}}o_t\mathbf{O}_{R_t}}{\tilde{\mathcal{X}}}$, and $\underset{o_to_t}{\tilde{\mathcal{O}}}$ for each t using (10), (13) and (15) from the training data. In the inference step, use (5) to compute $p(\mathbf{S}^{test})$. Algorithm 1 shows its basic version and Figure 4 shows the graphical representation of this algorithm in terms of the transformed junction tree of Figure 2.

As an example, consider the learning step of the algorithm and the computation of tensor in (10), i.e.,

$$\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}} = \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}}^{-1} \times_{\mathbf{O}_{L_{t-1}}} \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathcal{M}}.$$

Algorithm 1 Basic Spectral Algorithm for HSMM inference

Input: Training sequences: $\mathbf{S}^i = \{o_1^i, \ldots, o_{T_i}^i\}, i = 1, \ldots, N$.
Testing sequence: $\mathbf{S}^{test} = \{o_1^{test}, \ldots, o_T^{test}\}$.
Output: $p(\mathbf{S}^{test})$

Learning phase:
for all t **do**
  Estimate $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}}$, $\underset{\mathbf{O}_{R_t}o_t\mathbf{O}_{R_t}}{\tilde{\mathcal{X}}}$ and $\underset{o_to_t}{\tilde{\mathcal{O}}}$ from data $\{\mathbf{S}^1, \ldots, \mathbf{S}^N\}$ using equations (10), (13) and (15).
end for

Inference phase:
$p(\mathbf{S}^{test}) = 1$
for $t = T$ **down to** $t = 1$ **do**
  $p(\mathbf{S}^{test}) = p(\mathbf{S}^{test}) \times \underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}} \times_{\mathbf{O}_{R_t}} \left(\underset{\mathbf{O}_{R_t}o_t\mathbf{O}_{R_t}}{\tilde{\mathcal{X}}} \times_{o_t} \underset{o_to_t}{\tilde{\mathcal{O}}}\Big|_{o_t=o_t^{test}} \right)$
end for

For a fixed t, we estimate each entry of $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}}$ from the frequency of co-occurrence of tuples of the observed symbols $\{\ldots, o_{t-3}, o_{t-2}, o_{t+1}, o_{t+2}, \ldots\}$ in the given data set (the sets $\mathbf{O}_{L_{t-1}}$ and $\mathbf{O}_{R_{t-1}}$ were defined at the beginning of Section 4.2). Next, following our discussion after (9), we invert $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathcal{M}}^{-1}$ along the modes $\mathbf{O}_{L_{t-1}}$. For this, we matrisize the tensor so that the modes $\mathbf{O}_{L_{t-1}}$ are associated with columns and $\mathbf{O}_{R_{t-1}}$ with rows in matrix $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{L_{t-1}}}{\mathbf{M}}$ (see Section 2 for the discussion on tensor matrisization and inversion). Finally, we compute the right inverse of the matrix to obtain $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{L_{t-1}}}{\mathbf{M}}^{-1}$, so that $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{L_{t-1}}}{\mathbf{M}} \cdot \underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{L_{t-1}}}{\mathbf{M}}^{-1} = \mathbf{I}$

Similarly, we estimate the tensor $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathcal{M}}$ using the corresponding co-occurrences of the observed symbols. Matrisizing the result, so that the rows correspond to the modes $\mathbf{O}_{L_{t-1}}$ and the columns to $\mathbf{O}_{R_t}$, we get the matrix $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathbf{M}}$. The multiplication $\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{L_{t-1}}}{\mathbf{M}}^{-1} \cdot \underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_t}}{\mathbf{M}} = \underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathbf{D}}}$ produces a matrix, which is then converted to a tensor to get the final result in (10).

In the inference step we perform tensor multiplications for each t running along the length of the testing sequence. The only nuance here is that before multiplying the tensor $\underset{o_to_t}{\tilde{\mathcal{O}}}$ with others, the second mode o_t, whose dimension is n_o is collapsed into a scalar. This operation is denoted as $\underset{o_to_t}{\tilde{\mathcal{O}}}\Big|_{o_t=o_t^{test}}$, which means that based on the value of the tth symbol in testing sequence, we select the column corresponding to the element o_t^{test}. For example, if $\underset{o_to_t}{\tilde{\mathcal{O}}} \in \mathbb{R}^{10\times10}$ and $o_t^{test} = 3$ then $\underset{o_to_t}{\tilde{\mathcal{O}}}\Big|_{o_t=o_t^{test}} \in \mathbb{R}^{10\times1}$, a third column in the original matrix.

Analyzing (10), (13) and (15), we see that the computational complexity of the learning phase of the algorithm is determined by the tensor inverses and multiplications. For example, if in (10) we denote $|\mathbf{O}_R| = |\mathbf{O}_L| = \ell$ (in Section 5 we will show that $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$), then $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathbf{M}} \in \mathbb{R}^{n_o^\ell \times n_o^\ell}$

and $\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathbf{M}} \in \mathbb{R}^{n_o^\ell \times n_o^\ell}$. The computational complexity of the multiplications and inversions would then be $\mathcal{O}(n_o^{3\ell})$. Performing this computations for all t and assuming that the length of the sequences is T, would result in $\mathcal{O}\left(n_o^{3\ell}T\right)$. Additionally, with N training examples there will be a cost of $\mathcal{O}\left(\ell NT\right)$ to estimate the sample moments $\mathfrak{M}$, which is based on counting the co-occurrences of certain observable symbols. In the inference phase of the algorithm, we perform a series of tensor multiplications with the cost of $\mathcal{O}(n_o^{3\ell}T)$.

4.4 Efficient Version of Spectral Algorithm

Note that for large ℓ the accurate estimation of tensors $\mathfrak{M}$ for each t will require large number of training sequences which might not be available, leading to inaccurate and unstable computations. Observe, however, that for example the estimated sample-based tensors $\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathbf{M}}$ in (10) for each t estimate the same population quantity due to homogeneity of HSMM. Thus, a novel aspect of our work is the improvement of the accuracy and efficiency of the basic Algorithm 1 by exploiting the homogeneity property of HSMM and estimating the tensors $\tilde{\mathfrak{X}}$, $\tilde{\mathfrak{D}}$ and $\tilde{\mathfrak{O}}$ using all time steps, i.e., by pooling samples across different t and averaging the estimates. Thus, we compute only three tensors across all t, as opposed to computing these tensors separately for each t.

We show the details for computing the tensors $\tilde{\mathfrak{D}}$ in the batch form. The derivations for other tensors $\tilde{\mathfrak{X}}$ and $\tilde{\mathfrak{O}}$ can be computed in a similar manner. Recall from (10) the form of $\underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathfrak{D}}}$, and consider the following alternative expression, based on the sum over all t:

$$\tilde{\mathfrak{D}} = \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_{t-1}}}{\mathbf{M}}\right)^{-1} \times_{\mathbf{o}_L} \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathbf{M}}\right), \tag{16}$$

where $\mathbf{o}_L$ denotes a generic mode of the averaged tensor $\mathfrak{M}$, corresponding to $\mathbf{o}_{L_{t-1}}$ for all t. Note that in practice, instead of summation, we use averaging to avoid numerical overflow problems, and the average is equivalent to the expression in (16) since the term $\frac{1}{T}$ cancels out. Since

$$\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_{t-1}}}{\mathbf{M}} = \underset{\mathbf{o}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{\mathbf{o}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}}, \tag{17}$$

the first term inside brackets can be rewritten as:

$$\sum_t \underset{\mathbf{o}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{\mathbf{o}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}} \overset{(a)}{=} \sum_t \underset{\mathbf{o}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}}$$
$$\overset{(b)}{=} \underset{\mathbf{o}_{R_2}|x_2d_1}{\mathcal{F}} \times \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}}\right), \tag{18}$$

where in (a) we combined the two factors, i.e., $\underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}} = \underset{\mathbf{o}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}x_{t-1}d_{t-2}}{\mathcal{K}}$ and in (b) we used the homogeneity property of HSMM, i.e., the fact that $\underset{\mathbf{o}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$ does not depend on time stamp t, and extracted one of the common factors, in fact, the first factor. Note that the term $\underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}}$, on the other hand, does depend on t since the factor $\underset{x_{t-1}d_{t-2}}{\mathcal{K}}$, which represents the probability $p(x_{t-1}, d_{t-2})$, changes as the time stamp t changes.

Similarly, since

$$\underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathbf{M}} = \underset{\mathbf{o}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}, \tag{19}$$

rewrite the second term in (16) as

$$\sum_t \underset{\mathbf{o}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}$$
$$= \sum_t \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}} \times_{x_{t-1}d_{t-2}} \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}$$
$$= \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}}\right) \times \underset{d_2|x_2x_2d_1}{\mathcal{D}} \times_{x_2d_2} \underset{\mathbf{o}_{R_3}|x_2d_2}{\mathcal{F}}, \tag{20}$$

where we used the transformations similar as in (18), i.e., the fact that the factors $\underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}}$ and $\underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}$ are homogeneous, independent of t. Now if we multiply the inverse of (18) with (20), we get

$$\underset{\mathbf{o}_{R_2}|x_2d_1}{\mathcal{F}}^{-1} \times \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}}\right)^{-1} \times \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}x_{t-1}d_{t-2}}{\overline{\mathcal{F}}}\right) \times \underset{d_2|x_2x_2d_1}{\mathcal{D}} \times \underset{\mathbf{o}_{R_3}|x_2d_2}{\mathcal{F}} \tag{21}$$
$$= \underset{\mathbf{o}_{R_2}|x_2d_1}{\mathcal{F}}^{-1} \times_{x_2d_1} \underset{d_2|x_2x_2d_1}{\mathcal{D}} \times_{x_2d_2} \underset{\mathbf{o}_{R_3}|x_2d_2}{\mathcal{F}}$$
$$= \underset{\mathbf{o}_{R_2}\mathbf{o}_{R_3}}{\tilde{\mathfrak{D}}} = \underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathfrak{D}}}, \tag{22}$$

where in (21) we used the fact that the order in which tensors are multiplied is irrelevant and also the fact that the terms in parenthesis are invertible. This is due to the fact that the set of observations $\mathbf{o}_{L_{t-1}}$ for all t is selected so as to make each of the summand invertible (see Section 5 for the details about the choice of $\mathbf{o}_{L_{t-1}}$). Moreover, in (22) we used the definition of $\underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathfrak{D}}}$

$$\underset{\mathbf{o}_{R_{t-1}}\mathbf{o}_{R_t}}{\tilde{\mathfrak{D}}} = \underset{\mathbf{o}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}^{-1} \times \underset{d_{t-1}|x_{t-1}d_{t-2}}{\mathcal{D}} \times \underset{\mathbf{o}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}},$$

together with the homogeneity property of HSMM. We note that although the above derivations rely on the assumption of the existence of the matrix summation inverse in equation (21), the idea of aggregating observations from multiple time steps has also been utilized by other works, e.g., (Siddiqi et al., 2010; Anandkumar et al., 2014a) and shown to be very effective in practice, significantly improving the accuracy of corresponding algorithms.

We can conclude that the batch form of the tensor takes the form:

$$\tilde{\mathfrak{D}} = \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_{t-1}}}{\mathbf{M}}\right)^{-1} \times_{\mathbf{o}_L} \left(\sum_t \underset{\mathbf{o}_{L_{t-1}}\mathbf{o}_{R_t}}{\mathbf{M}}\right). \tag{23}$$

Algorithm 2 Efficient Spectral Algorithm for HSMM inference

Input: Training sequences: $\mathbf{S}^i = \{o_1^i, \ldots, o_{T_i}^i\}, i = 1, \ldots, N$.
Testing sequence: $\mathbf{S}^{test} = \{o_1^{test}, \ldots, o_T^{test}\}$.
Output: $p(\mathbf{S}^{test})$

Learning phase:
Estimate $\tilde{\mathcal{D}}, \tilde{\mathfrak{X}}$ and $\tilde{\mathfrak{O}}$ from data $\{\mathbf{S}^1, \ldots, \mathbf{S}^N\}$ using equations (23), (24) and (25).

Inference phase:
$p(\mathbf{S}^{test}) = 1$
for $i = T$ **down to** $i = 1$ **do**
$p(\mathbf{S}^{test}) = p(\mathbf{S}^{test}) \times \tilde{\mathcal{D}} \times \left(\tilde{\mathfrak{X}} \times \tilde{\mathfrak{O}}|_{o=o_i^{test}}\right)$
end for

Similar derivations can be carried out to obtain the rest of the tensors in the batch form:

$$\tilde{\mathfrak{X}} = \left(\sum_t \underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathfrak{M}}\right)^{-1} \times_{\mathbf{O}_L} \left(\sum_t \underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t} o_t}{\mathfrak{M}}\right), \tag{24}$$

$$\tilde{\mathfrak{O}} = \left(\sum_t \underset{o_t o_{t+1}}{\mathfrak{M}}\right)^{-1} \times_o \left(\sum_t \underset{o_t o_{t+1}}{\mathfrak{M}}\right). \tag{25}$$

where in the last expression the mode o corresponds to the mode $o_{t_{t+1}}$ after averaging of tensor $\underset{o_t o_{t+1}}{\mathfrak{M}}$ for all t.

Analyzing (23), (24) and (25), we see that the computational complexity of the learning phase of the Algorithm 2 is now $\mathcal{O}\left((n_o^{2\ell} + \ell N)T\right)$, mainly determined by the tensor additions and the estimation of the sample moments $\mathfrak{M}$. The number of inverses and multiplications is now fixed and independent of sequence length T. Specifically, there will be only three tensor multiplications and inversions for a total cost of $\mathcal{O}(n_o^{3\ell})$ (as opposed to T tensor multiplications and inversions as in Algorithm 1). The computational complexity of the inference phase is $\mathcal{O}(n_o^{3\ell}T)$, which is the same as for Algorithm 1.

Note that such a batch tensor computation significantly improves the accuracy of the resulting spectral algorithm. In part, this is due to the fact that we now use more data to estimate the tensors as compared to the original form (5). The estimates obtained in this form have lower variance, which in turn ensures that the inverses we compute in (23), (24) and (25) are more stable and accurate.

5. Rank Analysis of Observable Tensors

In Section 4.2.1, when we derived the equations (10), (13) and (15), we glossed over the question of the existence of tensor inverses $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathfrak{M}^{-1}}$, $\underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathfrak{M}^{-1}}$ and $\underset{o_t o_{t+1}}{\mathfrak{M}^{-1}}$. In this section, our task is to analyze the rank structure of these tensors and impose restrictions on the sets $\mathbf{O}_L$ and $\mathbf{O}_R$ to ensure that the rank conditions are satisfied. For example, consider equation (10) and expand all its terms using (8)

to get

$$\underset{\mathbf{O}_{R_{t-1}}\mathbf{O}_{R_t}}{\tilde{\mathcal{D}}} = \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times \overbrace{\underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times \underbrace{\underset{x_{t-1}d_{t-2}}{\mathcal{K}^{-1}} \times \underset{x_{t-1}d_{t-2}}{\mathcal{K}}}} \times \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times \underset{d_{t-1}|x_{t-1}x_{t-1}d_{t-2}}{\mathcal{D}} \times \underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}},$$

where we dropped the multiplication subscripts and some of the duplicated modes, which can be inferred from the context. Observe that in order for the above equation to produce (6), the terms in the middle must multiply out into identity tensor

$$\underset{x_{t-1}d_{t-2}}{\mathcal{I}} = \underset{x_{t-1}d_{t-2}}{\mathcal{K}^{-1}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}}{\mathcal{K}} \qquad \underset{x_{t-1}d_{t-2}}{\mathcal{I}} = \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times_{\mathbf{O}_{L_{t-1}}} \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}. \tag{26}$$

Moreover, recall that $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$ was originally introduced as part of the identity tensor

$$\underset{x_{t-1}d_{t-2}}{\mathcal{I}} = \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}^{-1}} \times_{\mathbf{O}_{R_{t-1}}} \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}, \tag{27}$$

therefore, we can conclude that for (10) to exist, the identity statements in (26) and (27) must be satisfied. These statements have implications for the ranks of $\underset{x_{t-1}d_{t-2}}{\mathcal{K}}$, $\underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$ and $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$, which in turn determine the length of the observation sequences $\mathbf{O}_{L_{t-1}}$ and $\mathbf{O}_{R_{t-1}}$.

Since $\underset{x_{t-1}d_{t-2}}{\mathcal{K}}$ represents a distribution $p(x_{t-1}d_{t-2})$, its matrisized version is a diagonal matrix with $p(x_{t-1}d_{t-2})$ on the diagonal. Using assumptions $A1$ and $A2$, it can be concluded that the diagonal elements in this matrix are non-zero and it has rank $n_x n_d$, it is thus invertible and so the first equation in (26) is satisfied.

Next, consider the second equation in (26) and recall from Section 2 that if we matrisize the tensor as $\underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathbf{F}} \in \mathbb{R}^{n_o^{|\mathbf{O}_{L_{t-1}}|} \times n_x n_d}$ then $\mathbf{F}$ must have full column rank $n_x n_d$ for the proper inverse to exist, implying $n_o^{|\mathbf{O}_{L_{t-1}}|} \geq n_x n_d$. Similarly, $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$ in (27) must have rank $n_x n_d$. As a consequence of the above, the tensor

$$\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathfrak{M}} = \underset{\mathbf{O}_{L_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times \underset{x_{t-1}d_{t-2}}{\mathcal{K}} \tag{28}$$

will have rank $n_x n_d$ and, in general, is rank-deficient.

The argument above can also be used to show that $\underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathfrak{M}}$ has rank $n_x n_d$ since in (12) the tensors $\underset{x_{t-1}d_{t-1}}{\mathcal{K}}$, $\underset{\mathbf{O}_{L_t}|x_{t-1}d_{t-1}}{\mathcal{F}}$ and $\underset{\mathbf{O}_{R_t}|x_{t-1}d_{t-1}}{\mathcal{F}}$ all have rank $n_x n_d$. Similarly, $\underset{o_t o_{t+1}}{\mathfrak{M}}$ will have rank n_x because in (15) the rank of the participating tensors $\underset{x_t}{\mathcal{K}}$, $\underset{o_{t+1}|x_t}{\mathcal{F}}$ and $\underset{o_t|x_t}{\mathcal{F}}$ is n_x. In particular, note that the tensor $\underset{o_t|x_t}{\mathcal{F}}$ is the observation matrix $O \in \mathbb{R}^{n_o \times n_x}$ of the model and it has rank n_x according to assumption $A3$. This conclusion also justifies our choice for $\omega_{x_t} = o_t$ at the end of Section 4.1.

The key unknowns now are the sets of the observed variables $\mathbf{O}_R$ and $\mathbf{O}_L$ that must be appropriately selected for the corresponding tensors to have rank $n_x n_d$. Recall that we defined $\mathbf{O}_{R_{t-1}} = \{o_t, o_{t+1}, \ldots\}$. As one of the new key results of our work, we established that if we select the observations o_t non-sequentially with gaps that grow exponentially with the state size n_x then the following result holds for all t:

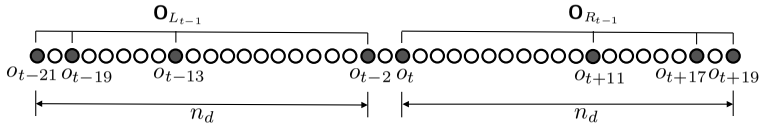


Figure 5: Observations required to estimate $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathfrak{M}}$ from data for HSMM with $n_x = 3$ and $n_d = 20$.

Theorem 1 *Let the number of observations be $|\mathbf{O}_{R_{t-1}}| = \ell$ and define the set of indices $\mathcal{S} = \{\max\{t,\ t+(n_d-1)-(n_x^i-1)\} \mid i = 0,\dots,\ell-1\}$, such that $\mathbf{O}_{R_{t-1}} = \{o_k | k \in \mathcal{S}\}$ then the rank of tensor $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$ is $\min[n_x^\ell,\ n_x n_d]$.*

As a consequence of this result, to achieve the rank $n_x n_d$ we will require $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$ observations, since we need to ensure $n_x^\ell \geq n_x n_d$ and we want the minimal ℓ which satisfies this. The span of the selected observations is n_d, while their number is only logarithmic in n_d. For example, consider the estimation of tensor $\underset{\mathbf{O}_{L_{t-1}}\mathbf{O}_{R_{t-1}}}{\mathfrak{M}}$ for an HSMM with $n_x = 3$ and $n_d = 20$. In this case $\ell = 4$ and $\mathbf{O}_{R_{t-1}} = \{o_t, o_{t+11}, o_{t+17}, o_{t+19}\}$ and $\mathbf{O}_{L_{t-1}} = \{o_{t-21}, o_{t-19}, o_{t-13}, o_{t-2}\}$, where the set $\mathbf{O}_{L_{t-1}}$ is defined similar to $\mathbf{O}_{R_{t-1}}$ in Theorem 1 but for the indices to the left of time stamp $t-1$. Figure 5 illustrates this example. We note that the requirement for the span of the selected observations to be n_d, which is a maximum state persistence, is to ensure that for a given time stamp t, we select the observations far enough to the right and left of it so that those observations are likely to be sampled from different hidden states.

In order to prove the above Theorem, we will focus our analysis on the tensor $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathcal{F}}$ instead of $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$. This specific choice was only done to ensure the compactness in our notations, however the HSMM homogeneity property enables us to transfer this result for tensors for any t. Note that

$$\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathcal{F}} = \underset{\mathbf{O}_{R_{t-1}}|x_{t-2}d_{t-2}}{\mathcal{F}} = \underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}} \times_{x_{t-1}d_{t-2}} \underset{x_{t-1}d_{t-2}|x_{t-2}d_{t-2}}{\mathcal{X}},$$

where the first equality is due to the homogeneity property of the model and in the second equality we embedded the HSMM transition matrix into tensor $\underset{x_{t-1}d_{t-2}|x_{t-2}d_{t-2}}{\mathcal{X}}$ with mode d_{t-2} duplicated. It can be shown that the matricized tensor $\underset{x_{t-1}d_{t-2}|x_{t-2}d_{t-2}}{\mathbf{X}} \in \mathbb{R}^{n_x n_d \times n_x n_d}$ has rank $n_x n_d$, i.e., it is full rank. Therefore, the rank structure of $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathcal{F}}$ determines the rank structure of $\underset{\mathbf{O}_{R_{t-1}}|x_{t-1}d_{t-2}}{\mathcal{F}}$.

The rest of Section 5 is devoted to the proof of Theorem 1. We first establish the rank structure of tensor $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathcal{F}}$ for sequential set of observations $\mathbf{O}_{R_{t+1}}$ and then analyze the rank structure for the observations which were selected non-sequentially.

5.1 Rank Structure of Tensor $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathcal{F}}$

Define by $\mathbf{X}_{R_{t+1}} = \{x_{t+2}, x_{t+3}, \dots\}$, the sequence of hidden states corresponding to observations $\mathbf{O}_{R_{t+1}} = \{o_{t+2}, o_{t+3}, \dots\}$. Then using conditional independence property of the graphical model in Figure 1, namely, that the variables $\mathbf{O}_{R_{t+1}}$ and $x_t d_t$ are independent given $\mathbf{X}_{R_{t+1}}$, we can write:

$$\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathcal{F}} = \underset{\mathbf{O}_{R_{t+1}}|\mathbf{X}_{R_{t+1}}}{\mathcal{Q}} \times \underset{\mathbf{X}_{R_{t+1}}|x_t d_t}{\mathcal{T}}, \tag{29}$$

for some tensors $\mathcal{Q}$ and $\mathcal{T}$, representing the appropriate probability distributions.

Denoting $\ell = |\mathbf{O}_{R_{t+1}}| = |\mathbf{X}_{R_{t+1}}|$, it can be verified, that the matrisized form of $\mathcal{Q}$ in (29) can be written as $\mathbf{Q} = \otimes_\ell O \in \mathbb{R}^{n_o^\ell \times n_x^\ell}$, i.e., a Kronecker product of the observation matrix O with itself ℓ times. According to the assumption $A3$, $rank(O) = n_x$ and $n_x \leq n_o$, and using the rank property of the Kronecker product, we infer that $rank(\mathbf{Q}) = n_x^\ell$.

Combining the above conclusion with the fact that the matrisized form of the other two tensors in (29) is $\mathbf{F} \in \mathbb{R}^{n_o^\ell \times n_x n_d}$ and $\mathbf{T} \in \mathbb{R}^{n_x^\ell \times n_x n_d}$, to ensure invertibility of $\mathcal{F}$, we need to select a set of variables $\mathbf{X}_{R_{t+1}}$ so that $rank\left(\underset{\mathbf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}\right) = n_x n_d$ with the condition that $n_x^\ell \geq n_x n_d$. Thus, the problem of the analysis of the rank structure of tensor $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathcal{F}}$ translates to the problem of rank structure of matrix $\underset{\mathbf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$. In what follows, we assume that $\mathbf{X}_{R_{t+1}} = \{x_{t+2}, \dots, x_{t+\ell+1}\}$ are sequential and so we would be interested in determining ℓ which makes $rank\left(\underset{\mathbf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}\right) = n_x n_d$. Later, the sequential assumption will be removed and we show how to select such variables in a more efficient way.

5.1.1 COMPUTATION OF FACTOR T

In order to study the rank structure of $\underset{\mathbf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ we will have to understand the mechanism how this matrix is constructed and how the rank changes as the size of $\mathbf{X}_{R_{t+1}}$ increases. We start by considering the following conditional independence relationships from the model in Figure 1:

$$p(x_{t+3}, x_{t+2}|x_{t+1}, d_{t+1}) = \sum_{d_{t+2}} p(x_{t+3}|x_{t+2}, d_{t+2}) \underline{p(d_{t+2}|x_{t+2}, d_{t+1}) p(x_{t+2}|x_{t+1}, d_{t+1})} \tag{30}$$

$$p(x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t) = \sum_{d_{t+1}} p(x_{t+3}, x_{t+2}|x_{t+1}, d_{t+1}) \underline{p(d_{t+1}|x_{t+1}, d_t) p(x_{t+1}|x_t, d_t)}. \tag{31}$$

Using the model's homogeneity property, we see that the quantity underlined in expression (30) is the same as the one in (31). Moreover, equation (30) can then be thought of as transforming $p(x_{t+1}|x_t, d_t)$ into $p(x_{t+2}, x_{t+1}|x_t, d_t)$, while the expression in (31), in effect, transforms probability $p(x_{t+2}, x_{t+1}|x_t, d_t)$ into $p(x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t)$. Thus (30) and (31) encode the following chain of transformations:

$$p(x_{t+1}|x_t, d_t) \to p(x_{t+2}, x_{t+1}|x_t, d_t) \to p(x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t).$$

Based on the above considerations, we can rewrite (30) and (31) in the tensor form as follows:

$$\underset{x_{t+3},x_{t+2}|x_{t+1},d_{t+1}}{\mathcal{T}} = \underset{x_{t+3},x_{t+2}|x_{t+2},d_{t+2}}{\mathcal{T}} \times_{x_{t+2}d_{t+2}} \underset{x_{t+2},d_{t+2}|x_{t+1}d_{t+1}}{\mathcal{V}} \tag{32}$$

$$\underset{x_{t+3},x_{t+2},x_{t+1}|x_t,d_t}{\mathcal{T}} = \underset{x_{t+3},x_{t+2},x_{t+1}|x_{t+1},d_{t+1}}{\mathcal{T}} \times_{x_{t+1}d_{d+1}} \underset{x_{t+1},d_{t+1}|x_td_t}{\mathcal{V}}, \tag{33}$$

where $\underset{x_{t+2},d_{t+2}|x_{t+1},d_{t+1}}{\mathcal{V}} = \underset{x_{t+1},d_{t+1}|x_t,d_t}{\mathcal{V}} = \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathcal{D}} \times_{x_{t+1}d_t} \underset{x_{t+1},d_t|x_t,d_t}{\mathcal{X}}$. The homogeneity property allows us to rewrite the above as

$$\underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathcal{T}} = \underset{x_{t+1},x_t|x_t,d_t}{\mathcal{T}} \times \mathcal{V} \tag{34}$$

$$\underset{x_{t+3},x_{t+2},x_{t+1}|x_t,d_t}{\mathcal{T}} = \underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathcal{T}} \times \mathcal{V}. \tag{35}$$

Our next step is to represent the above tensor equations in the matrix form. First, consider tensor $\mathcal{V}$, its matricized form can be written as:

$$\mathbf{V} = \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}} \ \underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}} \tag{36}$$

where $\underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}} \in \mathbb{R}^{n_x n_d \times n_x n_d}$ and $\underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}} \in \mathbb{R}^{n_x n_d \times n_x n_d}$. Next, consider the equations (34) and (35), its matrix version is of the form:

$$\underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} = \underset{x_{t+1},x_t|x_t,d_t}{\mathbf{T}} \ \mathbf{V} \tag{37}$$

$$\underset{x_{t+3},x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} = \underset{x_{t+2},x_{t+1},x_t|x_t,d_t}{\mathbf{T}} \ \mathbf{V}, \tag{38}$$

here the sizes of the matrices are $\underset{x_{t+1},x_t|x_t,d_t}{\mathbf{T}} \in \mathbb{R}^{n_x^2 \times n_x n_d}$, $\underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} \in \mathbb{R}^{n_x^2 \times n_x n_d}$, and similarly $\underset{x_{t+2},x_{t+1},x_t|x_t,d_t}{\mathbf{T}} \in \mathbb{R}^{n_x^3 \times n_x n_d}$, and matrix $\underset{x_{t+3},x_{t+2},x_t|x_t,d_t}{\mathbf{T}} \in \mathbb{R}^{n_x^3 \times n_x n_d}$.

Summarizing the above derivations, we can describe the following algorithmic approach for analyzing $\underset{\mathbf{X}_{R_{t+1}}|x_td_t}{\mathbf{T}}$ as $\mathbf{X}_{R_{t+1}}$ increases. We begin with $\underset{x_{t+1}|x_t,d_t}{\mathbf{T}} = [\mathcal{X}\ \mathbf{I} \cdots \mathbf{I}] \in \mathbb{R}^{n_x \times n_x n_d}$, where the first block $\mathcal{X} \in \mathbb{R}^{n_x \times n_x}$ corresponds to $d_t = 1$, and the subsequent $(n_d - 1)$ blocks of $\mathbf{I} \in \mathbb{R}^{n_x \times n_x}$ correspond to $d_t > 1$ for which $x_{t+1} = x_t$. We then use (37) to get $\underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}}$. However, notice that in (37) the matrix $\underset{x_{t+1},x_t|x_t,d_t}{\mathbf{T}}$ has a duplicated mode x_t, therefore, we need to restructure $\underset{x_{t+1}|x_t,d_t}{\mathbf{T}}$, which can be accomplished with:

$$\underset{x_{t+1},x_t|x_t,d_t}{\mathbf{T}'} = \underset{x_{t+1}|x_t,d_t}{\mathbf{T}} \odot \ \mathbf{E},$$

where $\mathbf{E} = [\mathbf{I} \cdots \mathbf{I}] \in \mathbb{R}^{n_x \times n_x n_d}$ and $\odot$ denotes a Khatri-Rao product (row-wise Kronecker product)[2]. Then, we use (38) to transform $\underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}}$ into $\underset{x_{t+3},x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}}$ where, again a

2. Let $\mathbf{P} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \vdots \\ \mathbf{p}_n \end{bmatrix} \in \mathbb{R}^{m \times n}$ and $\mathbf{Q} \in \mathbb{R}^{k \times n}$ then $\mathbf{P} \odot \mathbf{Q} = \begin{bmatrix} \mathbf{p}_1 \otimes \mathbf{Q} \\ \mathbf{p}_2 \otimes \mathbf{Q} \\ \vdots \\ \mathbf{p}_n \otimes \mathbf{Q} \end{bmatrix} \in \mathbb{R}^{mk \times n}$, where $\otimes$ is a Kronecker product.

Algorithm 3 Computation of $\underset{\mathbf{X}_{R_{t+1}}|x_td_t}{\mathbf{T}}$

Input: $p(d_t|x_t, d_{t-1})$ and $p(x_t|x_{t-1}, d_{t-1})$ - duration and transition distributions, ℓ - the number of sequential hidden states represented by $\mathbf{X}_{R_{t+1}}$.
Initialization:

$$p(x_{t+1}|x_t, d_t) \rightarrow \underset{x_{t+1}|x_t,d_t}{\mathbf{T}}$$
$$p(d_{t+1}|x_{t+1}, d_t) \rightarrow \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}}$$
$$p(x_{t+1}|x_t, d_t) \rightarrow \underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}}$$
$$\mathbf{V} = \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}} \ \underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}}, \quad \mathbf{E} = [\mathbf{I} \cdots \mathbf{I}]$$

for $i = 1$ **to** $\ell - 1$ **do**

$$\underset{x_{t+i},\ \ldots\ ,x_{t+1},x_t|x_t,d_t}{\mathbf{T}'} = \underset{x_{t+i},\ \ldots\ ,x_{t+1}|x_t,d_t}{\mathbf{T}} \odot \mathbf{E} \tag{39}$$

$$\underset{x_{t+i+1},\ \ldots\ ,x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} = \underset{x_{t+i},\ \ldots\ ,x_{t+1},x_t|x_t,d_t}{\mathbf{T}'} \mathbf{V} \tag{40}$$

end for

preliminary step is to restructure the matrix as follows:

$$\underset{x_{t+2},x_{t+1},x_t|x_t,d_t}{\mathbf{T}'} = \underset{x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} \odot \ \mathbf{E}.$$

Algorithm 3 summarizes the above constructions for a general case.

The following Theorem characterizes the rank structure of matrix $\underset{\mathbf{X}_{R_{t+1}}|x_td_t}{\mathbf{T}}$ in the output of the Algorithm 3. The proof can be found in Appendix A.1.

Theorem 2 *The rank of the output matrix* $\underset{\mathbf{X}_{R_{t+1}}|x_td_t}{\mathbf{T}}$ *in Algorithm 3 is* $\min(\ell n_x, n_x n_d)$.

Applying now Theorem 2 to equation (29) in matrix form

$$\underset{\mathbf{O}_{R_{t+1}}|x_td_t}{\mathbf{F}} = \underset{\mathbf{O}_{R_{t+1}}|\mathbf{X}_{R_{t+1}}}{\mathbf{Q}} \times \underset{\mathbf{X}_{R_{t+1}}|x_td_t}{\mathbf{T}},$$

where $rank(\mathbf{Q}) = n_x^\ell$ we can now conclude the following result:

Corollary 3 *To achieve the full column rank for* $\underset{\mathbf{O}_{R_{t+1}}|x_td_t}{\mathbf{F}} \in \mathbb{R}^{n_o^\ell \times n_x n_d}$, *i.e. to ensure that the rank of tensor* $\underset{\mathbf{O}_{R_{t+1}}|x_td_t}{\mathcal{F}}$ *is* $n_x n_d$, *the number of observations* ℓ *in* $\mathbf{O}_{R_{t+1}} = \{o_{t+2}, o_{t+3}, \ldots, o_{t+\ell+1}\}$ *must be equal to the maximum state persistence i.e.,* $\ell = n_d$.

Algorithm 4 Efficient computation of $\underset{\mathbf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$

Input: $p(d_t|x_t, d_{t-1})$ and $p(x_t|x_{t-1}, d_{t-1})$ - duration and transition distributions, ℓ - the number of sequential hidden states represented by $\mathbf{X}_{R_{t+1}}$
Initialization:

$$p(x_{t+1}|x_t, d_t) \rightarrow \underset{x_{t+1}|x_t,d_t}{\mathbf{T}}$$
$$p(d_{t+1}|x_{t+1}, d_t) \rightarrow \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}}$$
$$p(x_{t+1}|x_t, d_t) \rightarrow \underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}}$$
$$\mathbf{V} = \underset{x_{t+1},d_{t+1}|x_{t+1},d_t}{\mathbf{D}} \underset{x_{t+1},d_t|x_t,d_t}{\mathbf{X}}, \quad \mathbf{E} = [\mathbf{I} \cdots \mathbf{I}]$$

$c = 1$
for $i = 1$ **to** $\ell - 1$ **do**

$$\mathbf{T} = \mathbf{T}\ \mathbf{V} \tag{41}$$

if $i == (n_x)^c - 1$ **or** $i == \ell - 1$ **do**

$$\mathbf{T} = \mathbf{T} \odot \mathbf{E} \tag{42}$$

end if
$c = c + 1$
end for

5.1.2 EFFICIENT COMPUTATION OF FACTOR $\mathbf{T}$

In Corollary 3 we established that the number of observations in $\mathbf{O}_{R_{t+1}} = \{o_{t+2}, o_{t+3}, \ldots, o_{t+\ell+1}\}$ is $\ell = n_d$. Therefore, the sizes of the estimated quantities $\tilde{\mathcal{D}} \in \mathbb{R}^{n_o^{n_d} \times n_o^{n_d}}$ and $\tilde{\mathcal{X}} \in \mathbb{R}^{n_o^{n_d} \times n_o^{n_d} \times n_o}$ in Algorithm 3 will have exponential dependency on n_d. When maximum state persistence is large, the estimation of such quantity becomes impractical. Fortunately, we can modify Algorithm 3 to significantly reduce the number of observations. The idea is to apply the step (40) multiple times in-between the applications of step (39). Recall that in the previous construction we established that $\ell = n_d$ consecutive observations are sufficient, e.g., $\mathbf{O}_{R_{t+1}} = \{o_{t+2}, \ldots, o_{t+\ell+1}\}$. In contrast, in the proposed approach, every time we add an observation, say $o_{t+\tau}$, we skip certain number δ of time steps before adding another observation $o_{t+\tau+\delta}$, so that the observations are non-consecutive. As we illustrate next, the span of these non-consecutive observations is still n_d but the number of them is only logarithmic in n_d. The proposed approach still achieves the full rank structure of $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathbf{F}}$ but with smaller number of data points. Algorithm 4, which is a simple modification of Algorithm 3, summarizes the above procedure.

The following result establishes the rank structure of the matrix $\underset{\mathbf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ in the output of the Algorithm 4. The proof can be found in Appendix A.2.

Theorem 4 *The rank of the output matrix* $\underset{\mathbf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ *in Algorithm 4 is* $\min(n_x^\ell, n_x n_d)$.

Note that based on the above theorem, Algorithm 4 increases the rank at every step exponentially rather than linearly. In order for $\underset{\mathbf{X}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ to achieve the rank $n_x n_d$ we will now require $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$ observations, since we need to ensure $n_x^\ell = n_x n_d$. Observe that the span of the selected observations is still n_d, while the number of the observations is only logarithmic in n_d. The following Corollary summarizes the above conclusions.

Corollary 5 *To achieve the full column rank for* $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathbf{F}} \in \mathbb{R}^{n_o^\ell \times n_x n_d}$*, i.e. to ensure that the rank of tensor* $\underset{\mathbf{O}_{R_{t+1}}|x_t d_t}{\mathcal{F}}$ *is* $n_x n_d$*, the number of observations* ℓ *in* $\mathbf{O}_{R_{t+1}}$ *must be equal to* $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$*, since we need to ensure* $n_x^\ell = n_x n_d$.

Theorem 4 together with Corollary 5 now proves the Theorem 1 stated earlier.

6. Experiments

In this section we evaluated the performance of the proposed algorithm both on synthetic as well as real data sets and compared its performance to a standard EM algorithm.

6.1 Synthetic Data

Using synthetic data, we compared the estimation accuracy and the runtime of the proposed spectral algorithm with EM. For this, we defined two HSMMs, one with $n_o = 3, n_x = 2, n_d = 2$ and another with $n_o = 5, n_x = 4, n_d = 6$. For each model, we generated a set of $N_{train} = \{500, 1000, 5000, 10^4, 10^5\}$ training and $N_{test} == 1000$ testing sequences, each of length $T = 100$. The accuracy of estimating likelihood for each testing sequence was measured using the relative deviation from the true likelihood, i.e., $\epsilon_i = \frac{|\hat{p}(\mathbf{S}_i^{test}) - p(\mathbf{S}_i^{test})|}{p(\mathbf{S}_i^{test})}$ for $i = 1, \ldots, 1000$. Given 1000 such values, we then computed the final score, which is the root-mean-square error (RMSE) across all the testing sequences, RMSE $= \sqrt{\frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \epsilon_i^2}$.

Figure 6 shows results, where the top row of graphs corresponds to the model with $n_o = 3, n_x = 2, n_d = 2$ and the bottom row is for model with $n_o = 5, n_x = 4, n_d = 6$. The left column of graphs shows the progression of RMSE across EM iterations for both models; the middle column shows the dependence of testing error on the number of training samples and the right column shows the running times. It can be observed from plots (b) and (e) in Figure 6 that with the small training set, EM achieves smaller errors, while as the number of training samples increases, the spectral method becomes more accurate, outperforming EM. Also, comparing the plots (a), (b) with (d) and (e), we can conclude that for larger models, i.e., whose n_o, n_x and n_d are larger, the spectral method requires more data in order to achieve same or better accuracy than EM. This is expected since the sizes of estimated tensors grow with the model size. Moreover, the plots (c) and (f) in Figure 6 show that spectral method is several orders of magnitude faster than EM.

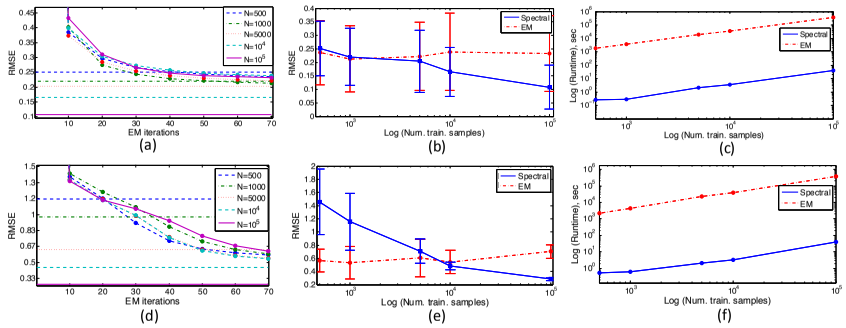


Figure 6: Performance of the spectral algorithm and EM on synthetic data generated from HSMM with $n_o = 3, n_x = 2, n_d = 2$ (top row) and $n_o = 5, n_x = 4, n_d = 6$ (bottom row). (a), (d): Error for EM across different iterations for various training data sets. The straight lines show the performance for spectral method. (b), (e): Average error and one standard deviation over 100 runs for EM after convergence and spectral algorithm across different number of training data. (c), (f): Runtime, in seconds, for both methods.

Given the above results, we can conclude that (i) for small data sets EM is a preferable algorithm, (ii) for large data, the spectral algorithm is a better choice, since it achieves higher accuracy and (iii) across all data sets the spectral algorithm requires significantly less computations as compared to EM.

6.2 Application to Aviation Safety Data

We also compared the performance of the spectral algorithm and EM on real NASA flight data set (NASA), containing over $180,000$ flights of 35 aircrafts from a defunct mid-western airline company. For each flight, the data has a record of 186 parameters, sampled at 1 Hz, including sensor readings and pilot actions. We considered a problem of anomaly detection in aviation systems (Budalakoti et al., 2009; Gorinevsky et al., 2012; Matthews et al., 2013) and used HSMM to detect abnormal flights based on pilot actions. Our idea is based on the observation that a flight can be partitioned into a number of phases, e.g., initial descent, touch down, or braking on the runway, and where within each phase the pilot performs certain actions. For example, during the initial descent, the pilot reduces throttle, lowers the flaps, and uses the ailerons and elevator to stabilize the aircraft. On the other hand, in the braking stage, the pilot uses brakes as well as rudder to keep the aircraft in the middle of the runway. Using HSMM as a model, we represented the flight phases as hidden states and the pilot actions as the observations from these states (see Melnyk et al. (2013) and Melnyk et al. (2016) for more details).

In our experiments, we focused on a part of flight related to the landing phase, which typically lasts 15-60 minutes in duration from when the flight crosses 10,000 ft while approaching an airport

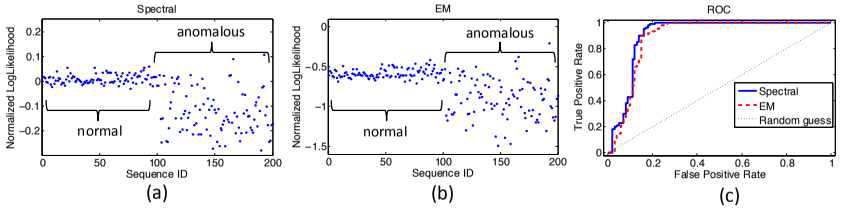


Figure 7: Evaluation of the spectral algorithm and EM on aviation safety data. (a) and (b): Normalized joint loglikelihood computed by spectral algorithm (a) and EM (b) for a set of 200 test flights, with 100 normal and 100 anomalous. HSMM parameters: $n_o = 9, n_x = 8, n_d = 40$ (c): The Receiver Operating Characteristic (ROC) curve, illustrating classification accuracy of the algorithms. Area Under Curve (AUC) for spectral algorithm is 0.91 and for EM is 0.89.

to the touch down on the runway. Our experiments are run on a subset of flights landing at the same airport. We chose 9 pilot commands, which include "selected altitude", "selected heading", "selected throttle level", etc. A simple data filter, based on the histogram of the pilot actions, was applied to select $10,020$ normal flights for training. The test set contained 200 flights, with 100 of them being similar to the training set and the rest were selected from the flights rejected by the filter. Most of anomalous flights contained low occurrence or rare events, such as fast descent, unusual usage of air brakes, etc., and few significant anomalies, e.g., aborted descent in order to delay the flight when the runway is not available. The length of the considered flight sequences varied from 500 to 4000 seconds.

For each flight in the testing set, we applied EM and spectral algorithm to compute the normalized joint log-likelihood

$$\frac{1}{T_i} \log p(o_1, o_2, \ldots, o_{T_i}),$$

where o_i are the observed pilot actions for test flight i, and T_i is the length of the test flight i, with $i = 1, \ldots, 200$. Figure 7 shows the results. The high-likelihood sequences were considered normal and low-likelihood ones classified as anomalous (see plots (a) and (b) in Figure 7). Both algorithms achieved similar detection accuracy, with the spectral algorithm having the Area Under Curve (AUC) score of 0.91 and the EM had AUC $= 0.89$ (see plot (c) on Figure 7). On the other hand, the computational time of the spectral algorithm was orders of magnitude smaller as compared to EM. We also compared performance of both algorithm on the same flight data while varying the dimensionality of the HSMM parameters (see Figure 8 and Table 2). We can see that although the performance of EM and spectral algorithm is similar across many models, the latter offers significant computational savings.

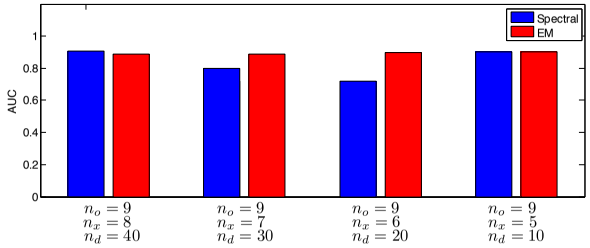


Figure 8: Comparison of AUC scores for EM and spectral algorithm for various model parameters when evaluated on aviation safety data. Both algorithms achieve similar high accuracy across different models.

Parameters		$n_o = 9$ $n_x = 8$ $n_d = 40$	$n_o = 9$ $n_x = 7$ $n_d = 30$	$n_o = 9$ $n_x = 6$ $n_d = 20$	$n_o = 9$ $n_x = 5$ $n_d = 10$
Running Time	Spectral	6.8 hours	6.4 hours	6.4 hours	6.3 hours
	EM	> 2 days	> 2 days	> 2 days	> 2 days

Table 2: Comparison of running time for EM and spectral algorithm for multiple model parameters on the flight data. Spectral algorithm is several orders of magnitude faster as compared to EM, offering significant computational savings.

7. Conclusion

In this paper, we presented a novel spectral algorithm to perform inference in HSMM. We derived an observable representation of the model which can be computed from the data sample moments of size logarithmic in the maximum length of latent state persistence. Based on the representation and exploiting the homogeneity of the model, we presented an efficient approach to inference, which ensures that during the training phase the number of matrix multiplications and inverses is fixed and independent of the sequence length of the observations. Empirical evaluation on synthetic and real flight data sets were presented to illustrate the promise of the proposed spectral algorithm. In particular, the spectral method gets similar or better performance than EM as the size of the training data set increases, and at the same time the spectral method is orders of magnitude faster than EM providing significant computational savings. Going forward, we plan to explore if similar spectral methods can be developed for inference in more general dynamic Bayesian networks.

Acknowledgments

We thank Nikunj Oza and Bryan Matthews at NASA for their helpful comments and suggestions at several stages of the work. We thank the editor and reviewers for helpful comments and suggestions which led to improvements in the paper. We also thank the Minnesota Supercomputing Institute (MSI) for the computing support. This work was supported by NASA grant NNX12AQ39A, and by NSF grants IIS-1563950, IIS-1447566, IIS-1447574, IIS-1422557, CCF-1451986, CNS- 1314560, IIS-0953274, IIS-1029711, and gifts from Adobe, IBM, and Yahoo.

Appendix A. Analysis of Tensor Rank Structure

A.1 Analysis of Algorithm 3

In this Section, we provide analysis of the Algorithm 3 and study the rank structure of matrix $\mathbf{T}$ in order to prove Theorem 2. To understand the analysis, it is important to know how the structure of matrix $\underset{\mathbf{x}_{R_{t+1}|x_t d_t}}{\mathbf{T}}$ evolves across iterations. For this, we present in Figure 9 a schematic description of a few steps of the algorithm. For the analysis we will need to establish certain auxiliary results.

Lemma 6 *Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be a matrix which has no all-zero columns, then the $rank\,(\mathbf{I} \odot \mathbf{A}) = rank\,(\mathbf{A} \odot \mathbf{I}) = n$, where $\odot$ denotes Khatri-Rao product and $\mathbf{I} \in \mathbb{R}^{n \times n}$.*

Proof Let $\mathbf{K} = (\mathbf{I} \odot \mathbf{A}) \in \mathbb{R}^{mn \times n}$. By definition of Khatri-Rao product, $\mathbf{K}(:, j) = \mathbf{e}_j \otimes \mathbf{A}(:, j)$, for $j = 1, \ldots, n$, which consists of zeros, except for rows $(j-1)m+1, \ldots, (j-1)m+m$, containing the column $\mathbf{A}(:, j)$. Here $\otimes$ denotes Kronecker product and $\mathbf{e}_j$ is everywhere zero except for

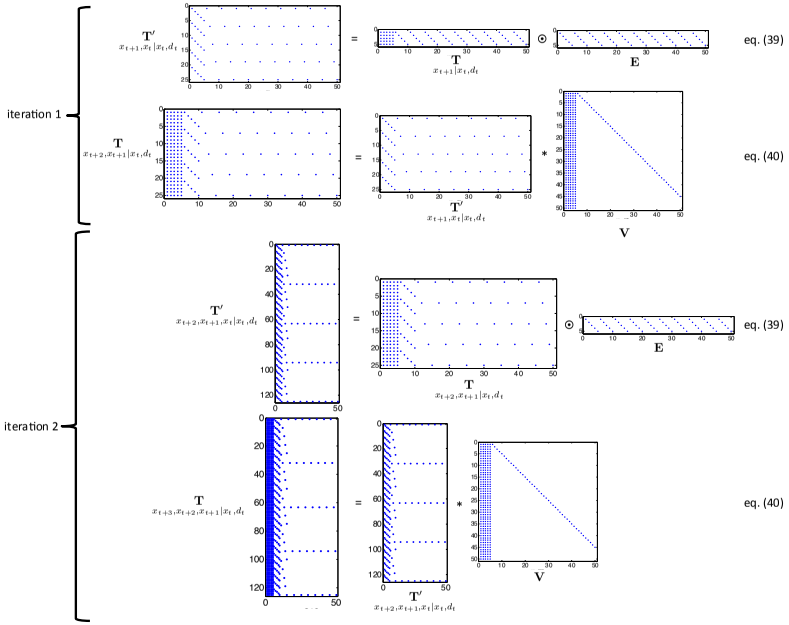


Figure 9: Schematic representation of Algorithm 3. This example illustrates the HSMM with $n_x = 5$ and $n_d = 10$. The non-zero matrix elements are displayed as dots.

position j which is 1. As long as there is no all-zero columns in $\mathbf{A}$, each column of $\mathbf{K}$ is independent of each other and therefore the rank is n. Moreover, since the matrix $\mathbf{A} \odot \mathbf{I}$ is a row-permuted version of $\mathbf{A} \odot \mathbf{I}$, their ranks are the same. ■

Lemma 7 *Define a block-row matrix $\mathbf{M} = [\mathbf{A}_1\ \mathbf{A}_2\ \cdots\ \mathbf{A}_k] \in \mathbb{R}^{m \times kn}$, where each $\mathbf{A}_i \in \mathbb{R}^{m \times n}$. Define by $r_j,\ j = 1, \ldots, n$ the rank of matrix $[\mathbf{A}_1(:, j)\ \cdots\ \mathbf{A}_k(:, j)]$ composed of jth columns of $\mathbf{A}$'s, and let $\mathbf{E} = [\mathbf{I}\,\mathbf{I}\,\cdots\,\mathbf{I}] \in \mathbb{R}^{n \times kn}$, where $\mathbf{I} \in \mathbb{R}^{n \times n}$. Then the rank of matrix $\mathbf{W} = \mathbf{M} \odot \mathbf{E} \in \mathbb{R}^{mn \times kn}$, obtained using a Khatri-Rao product, is $\min(mn, \sum_j r_j)$.*

Proof First note that $\mathbf{M} \odot \mathbf{E}$ and $\mathbf{E} \odot \mathbf{M}$ are row permuted version of each other, so they have the same rank. Therefore, consider $\mathbf{W}' = \mathbf{E} \odot \mathbf{M} = [\mathbf{I} \odot \mathbf{A}_1 \cdots \mathbf{I} \odot \mathbf{A}_k]$. Also, note that $\mathbf{e}_j \otimes [\mathbf{A}_1(:, j) \cdots \mathbf{A}_k(:, j)]$, $j = 1, \ldots, n$ is a matrix which consists of zeros except for rows $(j-1)m+1, \ldots, (j-1)m+m$ where it contains the columns $[\mathbf{A}_1(:, j)\ \cdots\ \mathbf{A}_k(:, j)]$. The rank of these columns is r_j and all other columns in $\mathbf{W}$ are independent of them due to the structure of the Khatri-Rao product. Therefore, each set of such columns adds r_j to the total rank. Since the overall rank of $\mathbf{W}$ cannot exceed either the number of rows or columns, we conclude that $rank(\mathbf{W}) = \min(mn, \sum_j r_j)$. ■

Lemma 8 *Let $V = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ be a set of linearly independent vectors. Define $\mathbf{u} = \sum_{i=1}^{n} c_i \mathbf{v}_i$, where coefficients $c_i \neq 0, i = 1, \ldots, n$. Define U to be a strict subset of V, i.e., $U \subset V$, then a set of vectors $\mathbf{u} \cup U$ is independent.*

Proof Define $\{1, \ldots, n\} = \alpha \cup \bar{\alpha}$, where α denotes a subset of indices for vectors corresponding to U. Then we can write $\mathbf{u} = \sum_{i:i \in \alpha} c_i \mathbf{v}_i + \sum_{j:j \in \bar{\alpha}} c_j \mathbf{v}_j$.

Assuming the opposite, i.e., $\mathbf{u} \cup U$ are dependent, we can write $k_0 \mathbf{u} + \sum_{i:i \in \alpha} k_i \mathbf{v}_i = 0$ where $k_0 \neq 0$ and some of $k_i, i \in \alpha$ are also must be non-zero. Substituting the definition of $\mathbf{u}$ and rearranging the terms, we get:

$$k_0 \sum_{i:i \in \alpha} (c_i + k_i) \mathbf{v}_i + k_0 \sum_{j:j \in \bar{\alpha}} c_j \mathbf{v}_j = 0.$$

Since $c_j \neq 0, j \in \bar{\alpha}$, the above equation claims the linear dependence of vectors in V, which is a contradiction of our assumption and so $\mathbf{u} \cup U$ are independent. ■

We are now ready to analyze Algorithm 3. It can be verified that (36) is of the form:

$$\mathbf{V} = \begin{bmatrix} \Psi \begin{bmatrix} \mathbf{I} & & \\ & \ddots & \\ & & \mathbf{I} \\ \hline \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix} \end{bmatrix} \in \mathbb{R}^{n_x n_d \times n_x n_d} \quad \text{where} \quad \Psi = \begin{bmatrix} \text{diag}\,[D(1,:)]\,\mathcal{X} \\ \text{diag}\,[D(2,:)]\,\mathcal{X} \\ \vdots \\ \text{diag}\,[D(n_d,:)]\,\mathcal{X} \end{bmatrix} \in \mathbb{R}^{n_x n_d \times n_x}, \tag{43}$$

where $\text{diag}\,[D(i,:)]$ is the diagonal matrix with ith row from D on the diagonal. Note that we can also write $\Psi = (D \odot \mathbf{I})\,\mathcal{X}$. Observe that the rank of $\mathbf{V}$ is $n_x n_d$ because the $n_x(n_d-1) \times n_x(n_d-1)$

block diagonal matrix delineated in (43) and the last $n_x \times n_x$ block matrix diag $[D(n_d,:)]\,\mathcal{X}$ in Ψ together comprising $n_x n_d$ independent columns of $\mathbf{V}$. Note that diag $[D(n_d,:)]\,\mathcal{X}$ has rank n_x because $\mathcal{X}$ is full rank and $D(n_d,:)$ is non-zero, which follows from assumptions $A1$ and $A2$. As a side note observe that the requirement to have $D(n_d,:)$ non-zero implies that there is a non-zero probability of maximum state persistence.

In analyzing the Algorithm 3, it would be useful to denote the matrices at iteration i in (39) and (40) as

$$\begin{aligned}\underset{x_{t+i},\ldots,x_{t+1}|x_t,d_t}{\mathbf{T}} &= [\mathbf{A}_1^{(i)} \cdots \mathbf{A}_{n_d}^{(i)}]\\ \underset{x_{t+i},\ldots,x_{t+1},x_t|x_t,d_t}{\mathbf{T}'} &= [\mathbf{B}_1^{(i)} \cdots \mathbf{B}_{n_d}^{(i)}]\\ \underset{x_{t+i+1},\ldots,x_{t+2},x_{t+1}|x_t,d_t}{\mathbf{T}} &= [\mathbf{C}_1^{(i)} \cdots \mathbf{C}_{n_d}^{(i)}].\end{aligned}$$

Moreover, utilizing the structure of matrix $\mathbf{V}$ from (43), the operations involved in step (40) are as follows:

$$\left[\mathbf{C}_1^{(i)}\ \mathbf{C}_2^{(i)}\ \mathbf{C}_3^{(i)}\ \cdots\ \mathbf{C}_{n_d}^{(i)}\right] = \left[[\mathbf{B}_1^{(i)} \cdots \mathbf{B}_{n_d}^{(i)}]\Psi\ \ \mathbf{B}_1^{(i)}\ \mathbf{B}_2^{(i)}\ \cdots\ \mathbf{B}_{n_d-1}^{(i)}\right]. \tag{44}$$

With the above information we can now present the proof of Theorem 2:

Proof of Theorem 2 At the start of the algorithm $\underset{x_{t+1}|x_t,d_t}{\mathbf{T}} = [\mathcal{X}\ \mathbf{I} \cdots \mathbf{I}] = [\mathbf{A}_1^{(1)} \cdots \mathbf{A}_{n_d}^{(1)}]$, which has rank n_x. The rank of matrix $\left[\mathbf{A}_1^{(1)}(:,l) \cdots \mathbf{A}_{n_d}^{(1)}(:,l)\right]$ for $l = 1,\ldots,n_x$ is $r_l = 2$ since among all the columns only two of them are independent. Therefore, according to Lemma 7, the result of operations in (39), has rank $\sum_l r_l = 2n_x$. Moreover, we note that since $[\mathbf{B}_1^{(1)}\ \mathbf{B}_2^{(1)}\ \cdots\ \mathbf{B}_{n_d}^{(1)}] = [\mathcal{X}\odot\mathbf{I}\ \mathbf{I}\odot\mathbf{I} \cdots \mathbf{I}\odot\mathbf{I}]$, it can be seen that its $2n_x$ independent vectors can be formed by the columns $[\mathbf{B}_1^{(1)}\ \mathbf{B}_2^{(1)}]$, so that the rank of $\left[\mathbf{B}_1^{(1)}(:,l) \cdots \mathbf{B}_{n_d}^{(1)}(:,l)\right]$ for $l = 1,\ldots,n_x$ is 2.

Next, since the rank of $\mathbf{V}$ is $n_x n_d$, the operations in (40) produce matrix $[\mathbf{C}_1^{(1)}\ \mathbf{C}_2^{(1)}\ \cdots\ \mathbf{C}_{n_d}^{(1)}]$ with the rank still being $2n_x$. Moreover, the columns of $\mathbf{C}_1^{(1)}$ are linearly dependent on the rest of the columns, $[\mathbf{C}_2^{(1)}\ \cdots\ \mathbf{C}_{n_d}^{(1)}]$, due to (44). However, the rank of $\left[\mathbf{C}_1^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\right]$ is now $r_l = 3$ for $l = 1,\ldots,n_x$. To understand this, note that

$$\begin{aligned}[\mathbf{B}_1^{(1)}\ \mathbf{B}_2^{(1)}\ \cdots\ \mathbf{B}_{n_d}^{1}] &= [\mathcal{X}\odot\mathbf{I}\ \mathbf{I}\odot\mathbf{I} \cdots \mathbf{I}\odot\mathbf{I}]\\ [\mathbf{C}_1^{(1)}\ \mathbf{C}_2^{(1)}\ \mathbf{C}_3^{(1)}\ \cdots\ \mathbf{C}_{n_d}^{(1)}] &= [\mathbf{C}_1^{(1)}\ \mathcal{X}\odot\mathbf{I}\ \mathbf{I}\odot\mathbf{I} \cdots \mathbf{I}\odot\mathbf{I}],\end{aligned}$$

where, according to (44), $\mathbf{C}_1^{(1)} = [\mathbf{B}_1^{(1)} \cdots \mathbf{B}_{n_d}^{(1)}]\Psi$. As we established before, the rank of the matrix $\left[\mathbf{C}_2^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\right] = \left[\mathbf{B}_1^{(1)}(:,l) \cdots \mathbf{B}_{n_d-1}^{(1)}(:,l)\right]$ is $r_l = 2$. Moreover, it can also be checked that $\mathbf{C}_1^{(1)}(:,l)$ is independent of $\left[\mathbf{C}_2^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\right]$ due to Lemma 8. Clearly, then the cumulative rank of $\left[\mathbf{C}_1^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\right]$ is 3 for $l = 1,\ldots,n_x$.

To generalize the above, if at the iteration i the rank of matrix $\left[\mathbf{A}_1^{(i)} \cdots \mathbf{A}_{n_d}^{(i)}\right]$ is in_x while the rank of $\left[\mathbf{A}_1^{(i)}(:,l) \cdots \mathbf{A}_{n_d}^{(i)}(:,l)\right]$ is $(i+1)$, then the operations in step (39) produce $\left[\mathbf{B}_1^{(i)} \cdots \mathbf{B}_{n_d}^{(i)}\right]$

having rank $(i+1)n_x$ due to Lemma 7. The step in (40) keeps the rank of $\left[\mathbf{C}_1^{(i)} \cdots \mathbf{C}_{n_d}^{(i)}\right]$ at $(i+1)n_x$ due to the full rank structure of $\mathbf{V}$. At the same time, this step increases the rank of matrix $\left[\mathbf{C}_1^{(i)}(:,l) \cdots \mathbf{C}_{n_d}^{(i)}(:,l)\right]$ to $(i+2)$ due to Lemma 8, i.e., independence of $\mathbf{C}_1^{(i)}(:,l)$ from $\left[\mathbf{C}_2^{(i)}(:,l) \cdots \mathbf{C}_{n_d}^{(i)}(:,l)\right]$ with the latter having the rank $(i+1)$. Therefore, each iteration increases the rank of matrix $\mathbf{T}$ by n_x and so after $2 \le \ell \le n_d$ steps the rank of the resulting matrix $\underset{\mathbf{x}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ is ℓn_x.

Note that if $\ell = 1$ then the Algorithm 3 is not executed and returns the trivial $\underset{x_{t+1}|x_t,d_t}{\mathbf{T}}$ with rank n_x. On the other hand, if $\ell > n_d$ then the rank of $\underset{\mathbf{x}_{R_{t+1}}|x_t d_t}{\mathbf{T}}$ is $n_x n_d$ since this is the number of columns in that matrix and so is the maximum achievable rank. ■

A.2 Analysis of Algorithm 4

In this Section we analysis of the Algorithm 4 in order to prove Theorem 4. Similarly as in Section A.1, it is instructive to visualize the progress of Algorithm 4. Figure 10 shows a schematic description of a few steps of the algorithm.

We are now ready to present the proof of Theorem 4.

Proof of Theorem 4 For the proof, we refer back to Algorithm 3 and the proof of Theorem 2. Recall, that at iteration $i = 1$, the result of step (39) is a matrix $[\mathbf{B}_1^{(1)} \cdots \mathbf{B}_{n_d}^{(1)}] \in \mathbb{R}^{n_x^2 \times n_x n_d}$, whose rank is $2n_x$, since $\left[\mathbf{A}_1^{(1)}(:,l) \cdots \mathbf{A}_{n_d}^{(1)}(:,l)\right] = [\mathcal{X}\,\mathbf{I} \cdots \mathbf{I}] \in \mathbb{R}^{n_x \times n_x n_d}$ for $l = 1,\ldots,n_x$ had two independent columns. Then, the transformations in step (40) produced $\left[\mathbf{C}_1^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\right]$ for $l = 1,\ldots,n_x$ with rank $3n_x$.

Note that if $n_x > 2$ then $\left[\mathbf{A}_1^{(1)}(:,l) \cdots \mathbf{A}_{n_d}^{(1)}(:,l)\right]$ potentially can have a rank up to n_x, while in Algorithm 3 we only have it equal to 2. It turns out that if we apply step (40) multiple times and use Lemma 8, we can increase the rank of $\left[\mathbf{C}_1^{(1)}(:,l) \cdots \mathbf{C}_{n_d}^{(1)}(:,l)\right]$ for $l = 1,\ldots,n_x$ to n_x.

Specifically, consider step (41). At iteration $i = 1$ we have $[\mathbf{A}_1^{(1)} \cdots \mathbf{A}_{n_d}^{(1)}] = [\mathbf{B}_1^{(1)} \cdots \mathbf{B}_{n_d}^{(1)}]$ and for $l = 1,\ldots,n_x$ the two independent columns are $\left[\mathbf{B}_1^{(1)}(:,l)\ \mathbf{B}_2^{(1)}(:,l)\right] = [\mathcal{X}(:,l)\ \mathbf{I}(:,l)]$. The result of step (41) gives us then three independent columns

$$\left[\mathbf{C}_1^{(1)}(:,l)\ \mathbf{C}_2^{(1)}(:,l)\ \mathbf{C}_3^{(1)}(:,l)\right] = \left[\mathbf{C}_1^{(1)}(:,l)\ \mathcal{X}(:,l)\ \mathbf{I}(:,l)\right],$$

where $\mathbf{C}_1^{(1)} = [\mathcal{X}\,\mathbf{I}\ \cdots\ \mathbf{I}]\Psi$. The independence follows from Lemma 8. The repeated application of step (41) one more time gives four independent columns

$$\left[\mathbf{C}_1^{(2)}(:,l)\ \mathbf{C}_2^{(2)}(:,l)\ \mathbf{C}_3^{(2)}(:,l)\ \mathbf{C}_4^{(2)}(:,l)\right] = \left[\mathbf{C}_1^{(2)}(:,l)\ \mathbf{C}_1^{(1)}(:,l)\ \mathcal{X}(:,l)\ \mathbf{I}(:,l)\right],$$

where $\mathbf{C}_1^{(2)} = [\mathbf{C}_1^{(1)} \cdots \mathbf{C}_{n_d}^{(1)}]\Psi$. Observe that since the number of rows is n_x, we can increase the rank at most up to n_x. Therefore, if in the beginning we had two independent columns and we want to get n_x independent columns, we would need to apply the step (41) $n_x - 2$ times, so as to have the matrix $[\mathbf{C}_1^{(n_x-2)}(:,l)\ \cdots\ \mathbf{C}_{n_d}^{(n_x-2)}(:,l)]$ with rank n_x.

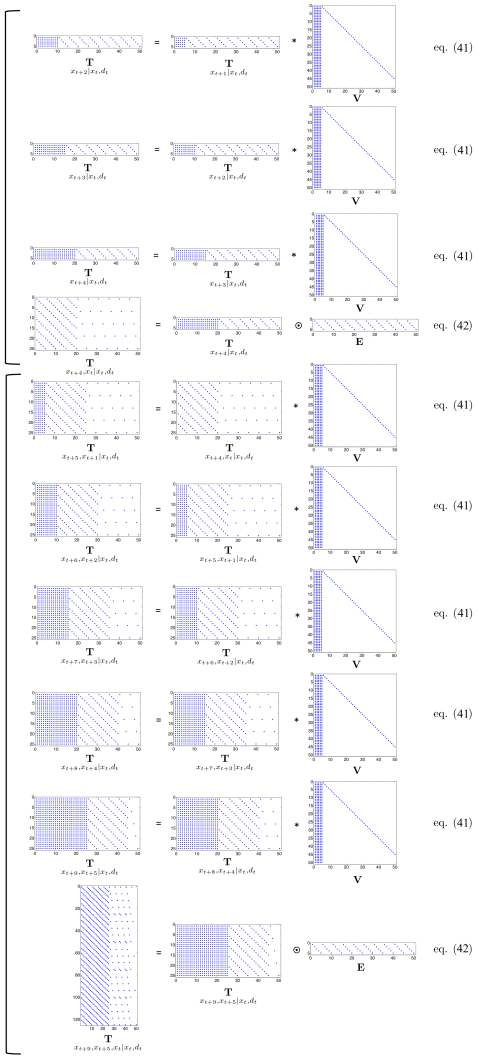


Figure 10: Schematic representation of Algorithm 4. This example illustrates the HSMM with $n_x = 5$ and $n_d = 10$. The non-zero matrix elements are displayed as dots.

If we now apply step (42) it will give us $[\mathbf{A}_1^{(1)} \ \cdots \ \mathbf{A}_{n_d}^{(1)}] \in \mathbb{R}^{n_x^2 \times n_x n_d}$ with rank n_x^2 due to Lemma 7. Continuing in this manner, we can again repeatedly apply the step (41) to create a matrix with a rank at most n_x^2, since there are n_x^2 rows and assuming that $n_x n_d \geq n_x^2$. The number of times we need to apply (41) is now $n_x^2 - n_x$ since we need to go from n_x to n_x^2 independent columns.

In general, the step (41) needs to be applied $n_x^c - n_x^{c-1}$, in order to obtain n_x^c independent columns. The application of step (42) then creates $\mathbf{T}$ with rank n_x^{c+1}. Note, that since $\mathbf{T}$ has $n_x n_d$ columns, the maximum achievable rank is $n_x n_d$. ■

Observe that the above proof also provided the method for selecting the non-sequential observations $\mathbf{X}_{R_{t+1}}$. Specifically, since the set of observations $\mathbf{X}_{R_{t+1}} = \{o_{t+2}, \ldots\}$ must start from observation o_{t+2} and $|\mathbf{X}_{R_{t+1}}| = \ell$, we denote $s = t+2$. Then, ith added observation is $o_{s+(n_d-1)-(n_x^i-1)}$ for $i = 0, \ldots, \ell-2$ and the ℓth observation is $o_s = o_{t+2}$. For tensor $\mathcal{F}_{\mathbf{O}_{R_{t+1}}|x_t d_t}$ to achieve rank $n_x n_d$ we need to add $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$ observations.

Appendix B. Initial and Final Parts of HSMM

In this Section we present the derivations for the initial and final steps of HSMM, which were omitted from the main text. Specifically, this amounts to computing the factor $\mathcal{X}$ for two parts of the model, corresponding to $\mathbb{X}_{root}$ and $\mathbb{X}_T$ in Figures 11 and 12. The derivations for all other parts of HSMM were presented in the main text and this supplement.

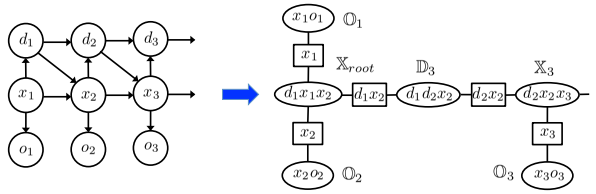


Figure 11: Part of HSMM corresponding to the initial time stamps and the related part of junction tree.

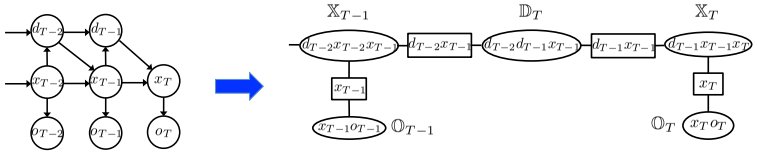


Figure 12: Part of HSMM corresponding to the final time stamps and the related part of junction tree.

To begin, recall the expression for the joint likelihood of the observed sequence:

$$\underset{o_1,\ldots,o_T}{\mathcal{P}} = \prod_t \underset{d_{t-1}|x_{t-1}d_{t-2}}{\mathcal{D}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_t|x_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{o_t|x_t}{\mathcal{O}} \right)$$

and rewrite the above expression by keeping only the initial and final factors:

$$\underset{o_1,\ldots,o_T}{\mathcal{P}} = \left(\underset{o_1|x_1}{\mathcal{O}} \times_{x_1} \left(\underset{x_2x_2|x_1d_1}{\mathcal{X}} \times_{x_2} \underset{o_2|x_2}{\mathcal{O}} \right) \right) \times_{x_2d_1} \underset{d_2|x_2x_2d_1}{\mathcal{D}} \times \cdots$$
$$\cdots \times \underset{d_{T-1}|x_{T-1}x_{T-1}d_{T-2}}{\mathcal{D}} \times_{x_{T-1}d_{T-1}} \left(\underset{x_T|x_{T-1}d_{T-1}}{\mathcal{X}} \times_{x_T} \underset{o_T|x_T}{\mathcal{O}} \right). \tag{45}$$

Introduce the identity tensors into (45), regroup the terms and extract the factors $\mathcal{X}$:

$$\underset{\omega_{x_1}\omega_{x_2}\omega_{x_2d_1}}{\tilde{\mathcal{X}}} = \underset{\omega_{x_1}|x_1}{\mathcal{F}} \times_{x_1} \left(\underset{x_2x_2|x_1d_1}{\mathcal{X}} \times_{x_2} \underset{\omega_{x_2}|x_2}{\mathcal{F}} \right) \times_{x_2d_1} \underset{\omega_{x_2d_1}|x_2d_1}{\mathcal{F}} \tag{46}$$

$$\underset{\omega_{x_{T-1}d_{T-1}}\omega_{x_T}}{\tilde{\mathcal{X}}} = \underset{\omega_{x_{T-1}d_{T-1}}|x_{T-1}d_{T-1}}{\mathcal{F}^{-1}} \times_{x_{T-1}d_{T-1}} \left(\underset{x_T|x_{T-1}d_{T-1}}{\mathcal{X}} \times_{x_T} \underset{\omega_{x_T}|x_T}{\mathcal{F}} \right). \tag{47}$$

Defining the observable sets $\omega_{x_1} = o_1$, $\omega_{x_2} = o_2$ and $\omega_{x_2d_1} = \mathbf{O}_{R_3}$ we can rewrite (46) as follows:

$$\underset{o_1o_2\mathbf{O}_{R_3}}{\tilde{\mathcal{X}}} = \underset{o_1|x_1}{\mathcal{F}} \times_{x_1} \left(\underset{x_2x_2|x_1d_1}{\mathcal{X}} \times_{x_2} \underset{o_2|x_2}{\mathcal{F}} \right) \times_{x_2d_1} \underset{\mathbf{O}_{R_3}|x_2d_1}{\mathcal{F}}. \tag{48}$$

Note that since all the factors participating in (48) are valid probability distributions, the resulting factor, i.e., $\underset{o_1o_2\mathbf{O}_{R_3}}{\tilde{\mathcal{X}}}$ is also a valid probability distribution, so it can be estimated directly from data. This is in contrast to the derivations we made for other parts of the model, where we had to perform additional transformations such as, for example in (10), in order to bring to the form, which could be estimated from the data samples.

In order to estimate (47), we compare it to the similar factor we considered in the main paper:

$$\underset{\omega_{x_{t-1}d_{t-1}}\omega_{x_t}\omega_{x_td_{t-1}}}{\tilde{\mathcal{X}}} = \underset{\omega_{x_{t-1}d_{t-1}}|x_{t-1}d_{t-1}}{\mathcal{F}^{-1}} \times_{x_{t-1}d_{t-1}} \left(\underset{x_tx_t|x_{t-1}x_{t-1}d_{t-1}}{\mathcal{X}} \times_{x_t} \underset{\omega_{x_t}|x_t}{\mathcal{F}} \right) \times_{x_td_{t-1}} \underset{\omega_{x_td_{t-1}}|x_td_{t-1}}{\mathcal{F}}, \tag{49}$$

and observe that the last factor $\underset{\omega_{x_td_{t-1}}|x_td_{t-1}}{\mathcal{F}}$ in (49) is a conditional probability distribution, which has the following marginalization property

$$\underset{\omega_{x_td_{t-1}}|x_td_{t-1}}{\mathcal{F}} \times_{\omega_{x_td_{t-1}}} \underset{\omega_{x_td_{t-1}}}{\mathbf{1}} = \underset{x_td_{t-1}}{\mathbf{1}}, \tag{50}$$

where $\mathbf{1}$ is the tensor, which has all elements equal to 1. The above can also be written in the scalar notations, $\sum_{\omega_{x_td_{t-1}}} p(\omega_{x_td_{t-1}}|x_td_{t-1}) = 1$ for each value of x_td_{t-1}. Therefore, if we apply (50) to (49), we get $\underset{\omega_{x_{t-1}d_{t-1}}\omega_{x_t}}{\tilde{\mathcal{X}}}$, which is the time-shifted version of $\underset{\omega_{x_{T-1}d_{T-1}}\omega_{x_T}}{\tilde{\mathcal{X}}}$. Therefore, to compute (47), we estimate the tensor in (13), i.e.,

$$\underset{\mathbf{O}_{R_t}o_t\mathbf{O}_{R_t}}{\tilde{\mathcal{X}}} = \underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathcal{M}^{-1}} \times_{\mathbf{O}_{L_t}} \underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}o_t}{\mathcal{M}},$$

and marginalize out the right set of modes, corresponding to $\mathbf{O}_{R_t}$. Alternatively, we can use the batch estimate

$$\tilde{\mathcal{X}} = \left(\sum_t \underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}}{\mathcal{M}} \right)^{-1} \times_{\mathbf{O}_L} \left(\sum_t \underset{\mathbf{O}_{L_t}\mathbf{O}_{R_t}o_t}{\mathcal{M}} \right),$$

and similarly perform the marginalization. This concludes our derivations.

References

A. Anandkumar, R. Ge, D. Hsu, and S. M. Kakade. A tensor spectral approach to learning mixed membership community models. In *Conference on Learning Theory*, 2013a.

A. Anandkumar, D. Hsu, M. Janzamin, and S. M. Kakade. When are overcomplete topic models identifiable? Uniqueness of tensor Tucker decompositions with structured sparsity. In *Advances in Neural Information Processing Systems*, pages 1986–1994, 2013b.

A. Anandkumar, A. Javanmard, D. Hsu, and S. M. Kakade. Learning linear Bayesian networks with latent variables. In *Proceedings of the International Conference on Machine Learning*, volume 28, pages 249–257, 2013c.

A. Anandkumar, R. Ge, D. Hsu, and S. M. Kakade. A tensor approach to learning mixed membership community models. *Journal of Machine Learning Research*, 15:2239–2312, 2014a.

A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15(1):2773–2832, 2014b.

R. Bailly, F. Denis, and L. Ralaivola. Grammatical inference as a principal component analysis problem. In *Proceedings of the International Conference on Machine Learning*, pages 33–40, 2009.

B. Balle, A. Quattoni, and X. Carreras. A spectral learning algorithm for finite state transducers. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 156–171, 2011.

D. Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012.

L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *The Annals of Mathematical Statistics*, 37(6):1554–1563, 1966.

B. Boots and G. J. Gordon. Predictive state temporal difference learning. In *Advances in Neural Information Processing Systems*, pages 271–279. 2010.

S. Budalakoti, A. N. Srivastava, and M. E. Otey. Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 39(1):101–113, 2009.

S. Chiappa. Explicit-duration Markov switching models. *Foundations and Trends in Machine Learning*, 7(6):803–886, 2014.

S. B. Cohen, K. Stratos, M. Collins, D. P. Foster, and L. Ungar. Spectral learning of latent-variable PCFGs: Algorithms and sample complexity. *Journal of Machine Learning Research*, 15:2399–2449, 2014.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, pages 1–38, 1977.

P. Dhillon, D. P. Foster, and L. H. Ungar. Multi-view learning of word embeddings via CCA. In *Advances in Neural Information Processing Systems*, pages 199–207, 2011.

E. B. Fox, E. B. Sudderth, M. I. Jordan, and A. S. Willsky. An HDP-HMM for systems with state persistence. In *Proceedings of the International Conference on Machine Learning*, pages 312–319, 2008.

D. Gorinevsky, B. Matthews, and R. Martin. Aircraft anomaly detection using performance models trained on fleet data. In *Proceedings of the Conference on Intelligent Data Understanding*, pages 17–23, 2012.

D. Hsu, S. M. Kakade, and T. Zhang. A spectral algorithm for learning hidden Markov models. *Journal of Computer and System Sciences*, 78(5):1460 – 1480, 2012.

H. Jaeger. Observable operator models for discrete stochastic time series. *Neural Computation*, 12 (6):1371–1398, 2000.

M. J. Johnson and A. S. Willsky. Bayesian nonparametric hidden semi-Markov models. *Journal of Machine Learning Research*, 14:673–701, 2013.

H. A. Kiers. Towards a standardized notation and terminology in multiway analysis. *Journal of chemometrics*, 14(3):105–122, 2000.

T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

D. A. Levin, Y. Peres, and E. Wilmer. *Markov chains and mixing times*. American Mathematical Society, 2009.

B. Matthews, S. Das, K. Bhaduri, K. Das, R. Martin, and N. Oza. Discovering anomalous aviation safety events using scalable data mining algorithms. *Journal of Aerospace Information Systems*, 10(10):467–475, 2013.

I. Melnyk, P. Yadav, M. Steinbach, J. Srivastava, V. Kumar, and A. Banerjee. Detection of precursors to aviation safety incidents due to human factors. In *Workshop on Domain Driven Data Mining (in conjunction with ICDM 2013)*, pages 407–412, 2013.

I. Melnyk, B. Matthews, H. Valizadegan, A. Banerjee, and N. Oza. Vector autoregressive model-based anomaly detection in aviation systems. *Journal of Aerospace Information Systems*, pages 161–173, 2016.

E. Mossel and S. Roch. Learning nonsingular phylogenies and hidden Markov models. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, pages 366–375, 2005.

K. P. Murphy. Hidden semi-Markov models. Available at http://www.cs.ubc.ca/ murphyk/Papers/segment.pdf. 2002.

NASA. Flight data set. Available at https://c3.nasa.gov/dashlink/projects/85/.

A. Parikh, L. Song, and E. Xing. A spectral algorithm for latent tree graphical models. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1065–1072, 2011.

A. Parikh, L. Song, M. Ishteva, G. Teodoru, and E. Xing. A spectral algorithm for latent junction trees. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 675–684, 2012.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

S. M. Siddiqi, B. Boots, and G. J. Gordon. Reduced-rank hidden Markov models. In *Proceedings of the Artificial Intelligence and Statistics Conference*, pages 741–748, 2010.

X. Tan and H. Xi. Hidden semi-Markov model for anomaly detection. *Applied Mathematics and Computation*, 205(2):562 – 567, 2008.

T. L. M. van Kasteren, G. Englebienne, and B. J. A. Krose. Activity recognition using semi-Markov models on real world smart home datasets. *Journal of Ambient Intelligence and Smart Environments*, 2(3):311–325, 2010.

Y. Xie and S.-Z. Yu. A large-scale hidden semi-Markov model for anomaly detection on user browsing behaviors. *IEEE/ACM Transactions on Networking*, 17(1):54–65, 2009.

S.-Z. Yu. Hidden semi-Markov models. *Artificial Intelligence*, 174(2):215 – 243, 2010.

S.-Z. Yu and H. Kobayashi. An efficient forward-backward algorithm for an explicit-duration hidden Markov model. *IEEE Signal Processing Letters*, 10(1):11–14, 2003.

H. Zen, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. A hidden semi-Markov model-based speech synthesis system. *Transactions on Information Systems*, E90-D(5):825–834, 2007.

Journal of Machine Learning Research 18 (2017) 1-30 Submitted 4/16; Revised 3/17; Published 4/17

Simplifying Probabilistic Expressions in Causal Inference

Santtu Tikka santtu.tikka@jyu.fi
Juha Karvanen juha.t.karvanen@jyu.fi
Department of Mathematics and Statistics
P.O.Box 35 (MaD) FI-40014 University of Jyvaskyla, Finland

Editor: Peter Spirtes

Abstract

Obtaining a non-parametric expression for an interventional distribution is one of the most fundamental tasks in causal inference. Such an expression can be obtained for an identifiable causal effect by an algorithm or by manual application of do-calculus. Often we are left with a complicated expression which can lead to biased or inefficient estimates when missing data or measurement errors are involved.

We present an automatic simplification algorithm that seeks to eliminate symbolically unnecessary variables from these expressions by taking advantage of the structure of the underlying graphical model. Our method is applicable to all causal effect formulas and is readily available in the R package causaleffect.

Keywords: simplification, probabilistic expression, causal inference, graphical model, graph theory

1. Introduction

Symbolic derivations resulting in complicated expressions are often encountered in many fields working with mathematical notation. These expressions can be derived manually or they can be outputs from a computer algorithm. In both cases, the expressions may be correct but unnecessarily complex in a sense that some unrecognized identities or properties would lead to simpler expressions.

We will consider simplification in the context of causal inference in graphical models (Pearl, 2009). Advances in causal inference have led to algorithmic solutions to problems such as identifiability of causal effects and conditional causal effects (Huang and Valtorta, 2006; Shpitser and Pearl, 2006a,b), z-identifiability (Bareinboim and Pearl, 2012), transportability and meta-transportability (Bareinboim and Pearl, 2013b,a) among others. The aforementioned algorithmic solutions operate symbolically on the joint distribution of the variables of interest and return expressions for the desired queries. These algorithms have been previously implemented in the R package causaleffect (Tikka and Karvanen, 2017). Another implementation of an identifiability algorithm can be found in the CIBN software by Jin Tian and Lexin Liu freely available from `http://web.cs.iastate.edu/~jtian/Software/CIBN.htm`. However, the algorithms themselves are imperfect in a sense that they often output an expression that is complicated and far from ideal. The question is whether there exists a simpler expression that is still a solution to the original problem.

©2017 Santtu Tikka and Juha Karvanen.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v18/16-166.html`.

Simplification of expressions may provide significant benefits. First, a simpler expression can be understood and reported more easily. Second, evaluating a simpler expression will be less of a computational burden due to reduced dimensionality of the problem. Third, in situations where estimation of causal effects is of interest and missing data is a concern, eliminating variables with missing data from the expression has clear advantages. The same applies to variables with measurement error.

We begin with presenting in Section 2 a general form of probabilistic expressions that are often encountered in causal inference. In this paper probabilistic expressions are formed by products of non-parametric conditional distributions of some variables and summations over the possible values of these variables. Simplification in this case is the process of eliminating terms from these expressions by carrying out summations. As our expressions correspond to causal effects, the expressions themselves take a specific form.

Causal models are typically associated with a directed acyclic graph (DAG) which represents the functional relationships between the variables of interest. In situations where the joint distribution is faithful, meaning that no additional conditional independences are generated by the joint distribution (Spirtes et al., 2000), the conditional independence properties of the variables can be read from the graph itself through a concept known as d-separation (Geiger et al., 1990). We will use d-separation as our primary tool for operating on the probabilistic expressions. The reader is assumed to be familiar with a number of graph theoretic concepts that are explained for example in (Koller and Friedman, 2009) and used throughout the paper.

Our simplification procedure is built on the definition of simplification sets, which is presented in Section 3. We continue by introducing a sound and complete simplification algorithm for probabilistic expressions defined in Section 2 for which these simplification sets exist. The algorithm takes as an input the expression to be simplified and the graph induced by the underlying causal model, and proceeds to construct a joint distribution of the variables contained in the expression by using the d-separation criteria. Higher level algorithms that use this simplification procedure are presented in Section 4. These include an algorithm for the simplification of a nested expression and an algorithm for the simplification of a quotient of two expressions. Section 5 contains examples on the application of these algorithms. We have also updated the causaleffect R-package to automatically apply these simplification procedures to causal effect expressions.

As a motivating example we present an expression of a causal effect given by the ID algorithm of Shpitser and Pearl (2006a) that can be simplified. The complete derivation of this effect can be found in Appendix C. The causal effect of X on Z_1, Z_2, Z_3 and Y is identifiable in the graph of Figure 1 and application of the ID algorithm gives

$$P(Z_1|Z_2,X)P(Z_3|Z_2)\frac{\sum_X P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2)}{\sum_{X,Y} P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2)}\times \sum_{X,Z_3,Y} P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2).$$

It turns out that there exists a significantly simpler expression,

$$P(Z_1|Z_2,X)P(Z_2)\sum_X P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2), \quad (1)$$

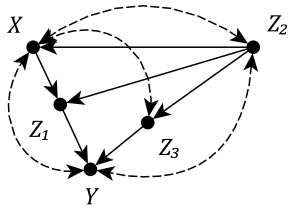


Figure 1: A graph for the introductory example on simplification.

for the same causal effect. This expression can be obtained without any knowledge of the underlying model by using standard probability manipulations. However, this requires that a favorable choice is made for the ordering of the nodes of the graph in the ID algorithm. In the case that we had chosen an ordering where Z_1 precedes Z_3, the term for Z_3 would instead be $P(Z_3|Z_2, Z_1, X)$ and simplification would require knowledge about the underlying graph. We will take another look at this example later in Section 5 where we describe in detail how our procedure can be used to find expression (1).

Our simplification procedure is different from the well-known exact inference method of minimizing the amount of numerical computations when evaluating expressions for conditional and marginal distributions by changing the order of summations and multiplications in the expression. Variants of this method are known by different names depending on the context, such as Bayesian variable elimination (Koller and Friedman, 2009) and the sum-product algorithm (Bishop, 2006) which is a generalization of belief propagation (Pearl, 1988; Lauritzen and Spiegelhalter, 1988). Efficient computational methods exist for causal effects as well, such as (Shpitser et al., 2011). The general principle is the same in all of the variants, and no symbolic simplification is performed.

In our setting simplification can be defined explicitly but in general it is difficult to say what makes one expression simpler than another. Carette (2004) provides a formal definition for simplification in the context of Computer Algebra Systems (CAS) that operate on algebraic expressions. Modern CAS systems such as Mathematica (Wolfram Research Inc., 2015) and Maxima (Maxima, 2014) implement techniques for symbolic simplification. Bailey et al. (2014) and references therein discuss simplification techniques in CAS systems further. However to the best of our knowledge, the symbolic simplification procedures for probabilistic expressions described in this paper have neither been given previous attention nor implemented in any existing system.

2. Probabilistic Expressions

Every expression that we consider is defined in terms of a set of variables $\mathbf{W}$. As we are interested in probabilistic expressions, we also assume a joint probability distribution P for the variables of $\mathbf{W}$. The most basic of expressions are called atomic expressions which will be the main focus of this paper.

Definition 1 (Atomic expression) *Let $\mathbf{W}$ be a set of p discrete random variables and let P be any joint distribution of $\mathbf{W}$. An atomic expression is a pair*

$$A = A[\mathbf{W}] = \langle \mathbf{T}, \mathbf{S} \rangle,$$

where

1. *$\mathbf{T}$ is a set of pairs $\{\langle V_1, \mathbf{C}_1 \rangle, \ldots, \langle V_n, \mathbf{C}_n \rangle\}$ such that for each V_i and $\mathbf{C}_i$ it holds that $V_i \in \mathbf{W}$, $\mathbf{C}_i \subseteq \mathbf{W}$, $V_i \notin \mathbf{C}_i$ and $V_i \neq V_j$ for $i \neq j$.*
2. *$\mathbf{S}$ is a set $\{S_1, \ldots, S_m\} \subseteq \mathbf{W}$ such that for each $i = 1, \ldots, m$ it holds that $S_i = V_j$ for some $j \in \{1, \ldots, n\}$.*

The value of an atomic expression A is

$$P_A = \sum_{\mathbf{S}} \prod_{i=1}^{n} P(V_i|\mathbf{C}_i).$$

The probabilities $P(V_i|\mathbf{C}_i)$ are referred to as the terms of the atomic expression. A term $P(V_i|\mathbf{C}_i)$ is said to contain a variable V if $V_i = V$ or $V \in \mathbf{C}_i$. A term for a variable V refers to a term $P(V|\cdot)$. We also use the shorthand notation $V[A] := \{V_1, \ldots V_n\}$. As $\mathbf{S}$ is a set, we will only sum over a certain variable once. All variables are assumed to be univariate and discrete for clarity, but we may also consider multivariates and situations where some of the variables are continuous and the respective sums are interpreted as integrals instead.

As an example we will construct an atomic expression describing the following formula

$$\sum_{X} P(Y|Z_2, X, Z_3, Z_1)P(Z_3|Z_2, X)P(X|Z_2)P(Z_2),$$

which is a part of the motivating example in the introduction. We let $\mathbf{W} = \{X, Y, Z_1, Z_2, Z_3\}$, which is the set of nodes of the graph of Figure 1. The sets $\mathbf{T}$ and $\mathbf{S}$ can now be defined as

$$\{\langle Y, \{Z_2, X, Z_3, Z_1\}\rangle, \langle Z_3, \{Z_2, X\}\rangle, \langle X, \{Z_2\}\rangle, \langle Z_2, \emptyset\rangle\} \quad \text{and} \quad \{X\},$$

respectively. Next we define a more general probabilistic expression.

Definition 2 (Expression) *Let $\mathbf{W}$ be a set of p variables and let P be the joint distribution of $\mathbf{W}$. An expression is a triple*

$$B = B[\mathbf{W}, n, m] = \langle \mathbf{B}, \mathbf{A}, \mathbf{S} \rangle,$$

where

1. *$\mathbf{S}$ is a subset of $\mathbf{W}$.*
2. *For $m > 0$, $\mathbf{A}$ is a set of atomic expressions*

$$\{\langle \mathbf{T}_1, \mathbf{S}_1 \rangle, \ldots, \langle \mathbf{T}_m, \mathbf{S}_m \rangle\}.$$

If $m = 0$ then $\mathbf{A} = \emptyset$.

3. For $n > 0$, $\mathbf{B}$ is a set of expressions

$$\{B_1[\mathbf{W}_1, n_1, m_1], \ldots, B_n[\mathbf{W}_n, n_n, m_n]\}$$

such that $\mathbf{W}_i \subseteq \mathbf{W}$, $n_i < n, m_i < m$ for all $i = 1, \ldots, n$. If $n = 0$ then $\mathbf{B} = \emptyset$.

The value of an expression B is

$$P_B = \sum_{\mathbf{S}} \prod_{i=1}^{n} P_{B_i} \prod_{j=1}^{m} P_{A_j},$$

where an empty product should be understood as being equal to 1.

The recursive definition ensures the finiteness of the resulting expression by requiring that each sub-expression has fewer sub-expressions of their own than the expression above it. A single value might be shared by multiple expressions, as the terms of the product in the value of the expression are exchangeable. Expressions $B_1[\mathbf{W}, n_1, m_1]$ and $B_2[\mathbf{W}, n_2, m_2]$ are equivalent if their values P_{B_1} and P_{B_2} are equal for all P. Equivalence is defined similarly for atomic expressions. Every expression is formed by nested atomic expressions by definition. Because of this, we focus on the simplification of atomic expressions.

As an example we construct an expression for the causal effect formula (1). We define $\mathbf{W} := \{X, Y, Z_1, Z_2, Z_3\}$ and let the sets $\mathbf{B}$ and $\mathbf{S}$ be empty. We define the set $\mathbf{A}$ to consist of three atomic expressions A_1, A_2 and A_3 defined as follows

$$\begin{aligned}
A_1 &= \langle\{\langle Z_1, \{Z_2, X\}\rangle\}, \emptyset\rangle, \\
A_2 &= \langle\{\langle Z_2, \emptyset\}\rangle\}, \emptyset\rangle, \\
A_2 &= \langle\{\langle Y, \{Z_2, X, Z_3, Z_1\}\rangle, \langle Z_3, \{Z_2, X\}\rangle, \langle X, \{Z_2\}\rangle, \langle Z_2, \emptyset\rangle\}, \{X\}\rangle.
\end{aligned}$$

In the context of probabilistic graphical models, we are provided additional information about the joint distribution of the variables of interest in the form of a DAG. As we are concerned on the simplification of the results of causal effect derivations in such models, the general form of the atomic expressions can be further narrowed down by using the structure of the graph and the ordering of vertices called a topological ordering.

Definition 3 (Topological ordering) *Topological ordering π of a DAG $G = \langle \mathbf{W}, \mathbf{E}\rangle$ is an ordering of its vertices, such that if X is an ancestor of Y in G then $X < Y$ in π.*

The symbol V_j^π is used to denote the subset of vertices of G that are less than V_j in π. For sets we may define $\mathbf{V}^\pi$ to contain those vertices of G that are less than every vertex of $\mathbf{V}$ in π. Consider a DAG $G = \langle \mathbf{W}, \mathbf{E}\rangle$ and a topological ordering π of its vertices. We use the notation $\pi(\cdot)$ to denote indexing over the vertex set $\mathbf{W}$ of G in the ordering given by π, that is $V_{\pi(1)} > V_{\pi(2)} > \cdots > V_{\pi(m)}$ where $m = |\mathbf{W}|$. For any atomic expression $A[\mathbf{V}] = \langle \mathbf{T}, \mathbf{S}\rangle$ such that $\mathbf{V} \subseteq \mathbf{W}$ we also define the induced ordering ω. This ordering is an ordering of the variables in $\mathbf{V}$ such that if $X > Y$ in ω then $X > Y$ also in π. From now on in this paper, any indexing over the variables of an atomic expression will refer to the induced ordering of the set $\mathbf{V}$ when π is given, i.e $V_1 > V_2 > \cdots > V_n$ in ω. In other words, ω is obtained from π by leaving out variables that are not contained in A.

The ID algorithm performs the so-called C-component factorization. These components are subgraphs of the original graph where every node is connected by a path consisting entirely of bidirected edges. The resulting expressions of these factors serve as the basis for our simplification procedure.

Definition 4 (Topological consistency) *Let G' be a DAG with a subgraph $G = \langle \mathbf{W}, \mathbf{E}\rangle$ and let π be a topological ordering of the vertices of G. An atomic expression $A[\mathbf{W}] = \langle \mathbf{T}, \mathbf{S}\rangle$ is topologically consistent (or π-consistent for short) if*

$$An(V_i)_G \subseteq \mathbf{C}_i \subseteq V_i^\pi \text{ for all } i = 1, \ldots, n.$$

Here $An(V_i)_G$ denotes the ancestors of V_i in G. To motivate this definition we note that the outputs of the algorithms of Shpitser and Pearl (2006a,b) can always be represented by using products and quotients of topologically consistent atomic expressions. An expression is topologically consistent when every atomic expression contained by it is topologically consistent with respect to a topological ordering of a subgraph. We provide a proof for this statement in Appendix A. This also shows that any manual derivation of a causal effect can always be represented by a topologically consistent expression. The assumption that $An(V_i)_G \subseteq \mathbf{C}_i$ is not necessary for the simplification to be successful. This assumption is used to speed up the performance of our procedure in Section 3.

3. Simplification

Simplification in our context is the procedure of eliminating variables from the set of variables that are to be summed over in expressions. In atomic expressions, a successful simplification in terms of a single variable should result in another expression that holds the same value, but with the respective term eliminated and the variable removed from the summation. As we are interested in causal effects, we consider only simplification of topologically consistent atomic expressions.

Our approach to simplification is that the atomic expression has to represent a joint distribution of the variables present in the expression to make the procedure feasible. The question is whether the expression can be modified to represent a joint distribution. Before we can consider simplification, we have to define this property explicitly.

Definition 5 (Simplification sets) *Let G' be a DAG and let G be a subgraph of G' over a vertex set $\mathbf{W}$ with a topological ordering π. Let $A[\mathbf{W}] = \langle \mathbf{T}, \mathbf{S}\rangle$, where $\mathbf{T} = \{\langle V_1, \mathbf{C}_1, \rangle, \ldots, \langle V_n, \mathbf{C}_n\rangle\}$, be a π-consistent atomic expression and let $V_j \in \mathbf{S}$. Suppose that $V_{\pi(p)} = V_j$ and that $V_{\pi(q)} = V_1$ and let $\mathbf{M}$ be the set*

$$\{U \in \mathbf{W} \mid U \notin V[A], V_{\pi(q)} > U > V_{\pi(p)}\}.$$

If there exists a set $\mathbf{D} \subset V_j^\pi$ and the sets $\mathbf{E}_U \subseteq \mathbf{W}$ for all $U \in \mathbf{M}$ such that the conditional distribution of the variables $V_{\pi(p)}, \ldots, V_{\pi(q)}$ can be factorized as

$$P(V_{\pi(p)}, \ldots, V_{\pi(q)} | \mathbf{D}) = \prod_{U \in \mathbf{M}} P(U | \mathbf{E}_U) \prod_{V_i \geq V_j} P(V_i | \mathbf{C}_i), \tag{2}$$

and

$$(U \perp\!\!\!\perp V_j | \mathbf{E}_U \setminus \{V_j\})_{G'} \text{ for all } U \in \mathbf{M}. \quad (3)$$

then the sets $\mathbf{D}$ *and* $\mathbf{E}_U, U \in \mathbf{M}$ *are the simplification sets of* A *with respect to* V_j.

This definition is tailored for the next result that can be used to determine the existence of a simpler expression when simplification sets exist. Afterwards we will show how this result can be applied in practice via an example. The definition characterizes π-consistent atomic expressions that represent joint distributions. It is apparent that simplifications sets are not always unique, which can lead to different but still simpler expressions. Henceforth the next result considers simplification in terms of a single variable. The proof is available in Appendix B.

Theorem 6 (Simplification) *Let* G' *be a DAG and let* G *be a subgraph of* G' *over a vertex set* $\mathbf{W}$ *with a topological ordering* π. *Let* $A[\mathbf{W}] = \langle \mathbf{T}, \mathbf{S} \rangle$ *be a* π*-consistent atomic expression and let* $\mathbf{D}$ *and* $\mathbf{E}_U, U \in \mathbf{M}$ *be its simplification sets with respect to a variable* $V_j \in \mathbf{S}$. *Then there exist an expression* $A'[\mathbf{W} \setminus \{V_j\}] = \langle \mathbf{T}', \mathbf{S}' \rangle$ *such that* $V_j \notin \mathbf{S}'$, $P_A = P_{A'}$ *and no term in* A' *contains* V_j.

Note that even if $\mathbf{M} = \emptyset$ in Definition 5, the existence of simplification sets still requires that $\prod_{V_i \geq V_j} P(V_i | \mathbf{C}_i) = P(V_j, \ldots, V_1 | \mathbf{D})$. In many cases there exists variables $U \in \mathbf{M}$ such that the expression does not contain a term for U. Condition (2) of Definition 5 guarantees that if these terms were contained in the expression it would represent a joint distribution. Our goal is thus to introduce these terms into the original expression temporarily, carry out the desired summation, and finally remove the added terms. This can only be achieved if the variables in the set $\mathbf{M}$ are conditionally independent of the variable currently being summed over, hence the assumption $(U \perp\!\!\!\perp V_j | \mathbf{E}_U \setminus \{V_j\})_{G'}$ of condition (3) of Definition 5.

We show how simplification sets can be used in practice to derive a simpler expression via an example. We consider the causal effect of $\{X, Z, W\}$ on Y in the graph G of Figure 2.

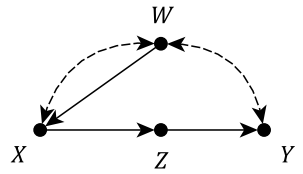


Figure 2: A graph G for the example on the use of simplification sets.

The effect in question is identifiable and the ID algorithm readily gives atomic expression

$$\sum_{X,W} P(Y | X, W, Z) P(X | W) P(W).$$

We consider simplification sets with respect to $V_j = W$. The topological order is $W < X < Z < Y$. The atomic expression does not contain a term for Z so we have $\mathbf{M} = \{Z\}$. By noting that $(Z \perp\!\!\!\perp W | X)_G$ we are able to satisfy condition (3) of Definition 5. We can write

$$P(Y, Z, X, W) = P(Z | X, W) P(Y | X, W, Z) P(X | W) P(W),$$

as required by condition (2) of Definition 5 by setting $\mathbf{E}_Z = \{X, W\}$. Thus, the simplification sets $\mathbf{D}$ and $\mathbf{E}_Z$ for the atomic expression with respect to W are $\emptyset$ and $\{X, W\}$, respectively. Finally, we obtain the simpler atomic expression by carrying out the summation over W:

$$\sum_X P(Y | X, Z) P(X).$$

Neither Definition 5 nor Theorem 6 provide a method to obtain simplification sets or to determine whether they exist in general. To solve this problem we present a simplification algorithm for π-consistent atomic expressions that operates by constructing simplification sets iteratively for each variable in the summation set.

Algorithm 1 always attempts to perform maximal simplification, meaning that as many variables of the set $\mathbf{S}$ are removed as possible. If the simplification in terms of the entire set $\mathbf{S}$ can not be completed, the intermediate result with as many variables simplified as possible is returned. If simplification in terms of specific variables or a subset is preferred, the set $\mathbf{S}$ should be defined accordingly.

The function SIMPLIFY takes three arguments: an atomic expression $A[\mathbf{W}]$ that is to be simplified, a graph G and a topological ordering π of its vertices. A is assumed to be π-consistent.

On line 10 the function INDEX.OF returns the corresponding index i of the term containing S_j. Since A is π-consistent, we only have to iterate through the variables $V_1, \ldots, V_j$ as the terms outside this range contain no relevant information about the simplification of V_j. The variables without a corresponding term in the atomic expression A are retrieved on line 11 by the function GET.MISSING. This function returns the set $\mathbf{M}$ of Definition 5 with respect to the current variable to be summed over.

In order to show that the term of A represent some joint distribution, we proceed in the order dictated by the topological ordering of the vertices. The sets $\mathbf{J}$ and $\mathbf{D}$ keep track of the variables that have been successfully processed and of the conditioning set of the joint term that was constructed on the previous iteration. Similarly, the sets $\mathbf{R}$ and $\mathbf{I}$ keep track of the variables and conditioning sets of the corresponding variables that the atomic expression does not originally contain a term for. Iteration through relevant terms begins on line 13. Next, we take a closer look at the function JOIN which is called next on line 14.

Here $\mathcal{P}(\cdot)$ denotes the power set, $\triangle$ denotes the symmetric difference and $An^*(\cdot)_G$ denotes the ancestors with the argument included. The function JOIN attempts to combine the joint term $P(\mathbf{J} | \mathbf{D})$, obtained from the previous iteration steps, with the term $P(V | \mathbf{C}) := P(V_k | \mathbf{C}_k)$ of the current iteration step. d-separation statements of G are evaluated to determine whether this can be done. In practice this means finding a suitable subset $\mathbf{P}_i$ of $\mathbf{G}$, where $\mathbf{G} \cup An(V)_G$ is the largest possible conditioning set of the new combined term. The set $\mathbf{G}$ is computed on line on line 4 of Algorithm 2. A valid subset $\mathbf{P}_i$ satisfies $P(\mathbf{J} | \mathbf{D}) = P(\mathbf{J} | An^*(V)_G, \mathbf{P}_i)$ and $P(V | \mathbf{C}) = P(V | An(V)_G, \mathbf{P}_i)$ which allow us to write the product $P(\mathbf{J} | \mathbf{D}) P(V | \mathbf{C})$ as $P(\mathbf{J}, V | An(V)_G, \mathbf{P}_i)$.

Algorithm 1 Simplification of an atomic expression $A = \langle \mathbf{T}, \mathbf{S} \rangle$ given graph G and topological ordering π.

```
1:  function SIMPLIFY(A, G, π)
2:      j ← 0
3:      while j < |S| do
4:          B ← A
5:          J ← ∅
6:          D ← ∅
7:          R ← ∅
8:          I ← ∅
9:          j ← j + 1
10:         i ← INDEX.OF(A, j)
11:         M ← GET.MISSING(A, G, j)
12:         k ← 1
13:         while k ≤ i do
14:             ⟨J_new, D_new, R_new⟩ ← JOIN(J, D, V_k, C_k, S_j, M, G, π)
15:             if J_new ⊆ J then
16:                 break
17:             else
18:                 J ← J_new
19:                 D ← D_new
20:                 if R_new ≠ ∅ then
21:                     R ← R ∪ R_new
22:                     I ← I ∪ {D}
23:                     M ← M \ R_new
24:                 else
25:                     k ← k + 1
26:         if k = i + 1 then
27:             A_new ← FACTORIZE(J, D, R, I, A)
28:             if A_new = A then
29:                 A ← B
30:             else
31:                 A ← A_new
32:                 S ← S \ {S_j}
33:                 j ← 0
34:     return A
```

In order to find this valid subset, we compute the sets $\mathbf{A}$ and $\mathbf{B}$ for each candidate on lines 8 and 9. These sets characterize the necessary change in the conditioning sets of the terms $P(\mathbf{J}|\mathbf{D})$ and $P(V|\mathbf{C})$ that would enable a joint term to be formed by these two terms. The validity of the candidate set is finally checked on line 10 which determines if the necessary change is allowed by d-separation criteria in the graph G. If no valid subset $\mathbf{P}_i$ can be found, we can still attempt to insert a missing variable of $\mathbf{M}$ by calling INSERT. If this

Algorithm 2 Construction of the joint distribution of the set $\mathbf{J}$ and a variable V given their conditional sets $\mathbf{D}$ and $\mathbf{C}$ using d-separation criteria in G. S is the current summation variable, $\mathbf{M}$ is the set of variables not contained in the expression and π is a topological ordering.

```
1:  function JOIN(J, D, V, C, S, M, G, π)
2:      if J = ∅ then
3:          return ⟨{V}, C, ∅⟩
4:      G ← J^π \ An*(V)_G
5:      P ← P(G)
6:      n ← |P|
7:      for i = 1 : n do
8:          A ← (An*(V)_G ∪ P_i) △ D
9:          B ← (An(V)_G ∪ P_i) △ C
10:         if (J ⫫ A|D \ A)_G and (V ⫫ B|C \ B)_G then
11:             return ⟨J ∪ {V}, (An(V)_G ∪ P_i), ∅⟩
12:     if M ≠ ∅ then
13:         for M' ∈ M do
14:             if M' ∈ D, M' ∉ C then
15:                 ⟨J_new, D_new, R⟩ ← INSERT(J, D, M', S, G, π)
16:                 if J ⊂ J_new then
17:                     return ⟨J_new, D_new, R⟩
18:     return ⟨J, D, ∅⟩
```

does not succeed either, the original sets $\mathbf{J}$ and $\mathbf{D}$ are returned, which instructs SIMPLIFY to terminate simplification in terms of V_j and attempt simplification in the next variable.

A special case where the first variable of the joint distribution forms $P(\mathbf{J}, \mathbf{D})$ alone is processed on line 2 of Algorithm 2. In this case, we have an immediate result without having to iterate through the subsets of $\mathbf{G}$. The formulation of the set $\mathbf{G}$ ensures that the resulting factorization is π-consistent if it exists. Knowing that the ancestral set $An(V)_G$ has to be a subset of the new conditioning set also greatly reduces the amount of subsets we have to iterate through. In a typical situation, the size of $\mathbf{P}$ is not very large. Let us now inspect the insertion procedure in greater detail.

In essence, the function INSERT is a simpler version of JOIN, because the only restriction on the conditioning set of M' is imposed by the conditioning set of $\mathbf{J}$ and the fact that M' has to be conditionally independent of the current variable S to be summed over. If JOIN or INSERT was unsuccessful in forming a new joint distribution, we have that $\mathbf{J}_{\text{new}} \subset \mathbf{J}$. In this case simplification in terms of the current variable cannot be completed. If we have that $\mathbf{J}_{\text{new}} \not\subset \mathbf{J}$ the iteration continues.

Together the functions JOIN and INSERT capture the two conditions of Definition 5. They are essentially two variations of the underlying procedure of determining whether the terms of the atomic expression actually represent a joint distribution. The only difference is that JOIN is called when we are processing terms that already exist in the expression, and INSERT

Algorithm 3 Insertion of variable M' into the joint term $P(\mathbf{J}|\mathbf{D})$ using d-separation criteria in G. S is the current summation variable and π is a topological ordering.

1: **function** INSERT($\mathbf{J}, \mathbf{D}, M', S, G, \pi$)
2: $\quad\mathbf{G} \leftarrow \mathbf{J}^{\pi} \setminus An^*(M')_G$
3: $\quad n \leftarrow |\mathbf{G}|$
4: $\quad$**for** $i = 1 : n$ **do**
5: $\quad\quad\mathbf{A} \leftarrow (An^*(M')_G \cup \mathbf{P}_i) \triangle \mathbf{D}$
6: $\quad\quad\mathbf{B} \leftarrow (An(M')_G \cup \mathbf{P}_i)$
7: $\quad\quad$**if** $(\mathbf{J} \perp\!\!\!\perp \mathbf{A}|\mathbf{D} \setminus \mathbf{A})_G$ **and** $(M' \perp\!\!\!\perp \mathbf{S}|\mathbf{B} \setminus \mathbf{S})_G$ **then**
8: $\quad\quad\quad$**return** $\langle \mathbf{J} \cup \{M'\}, (An^*(M')_G \cup \mathbf{P}_i), \{M'\}\rangle$
9: $\quad$**return** $\langle \mathbf{J}, \mathbf{D}, \emptyset\rangle$

is called when there are variables without corresponding terms in the expression, that is the set $\mathbf{M}$ of Definition 5 is not empty.

If the innermost while-loop of Algorithm 1 succeeded in iterating through the relevant variables, we are ready to complete the simplification process in terms of S_j. We carry out the summation over S_j which results in $P(\mathbf{J} \setminus \{V_i\}|\mathbf{D})$. This is done on line 27 by calling FACTORIZE($\mathbf{J}, \mathbf{D}, \mathbf{R}, \mathbf{I}, A$) which checks whether the joint term $P(\mathbf{J} \setminus \{V_i\}|\mathbf{D})$ can be factorized back into a product of terms. In practice this means that if the function succeeds, it will return an atomic expression obtained by removing each inserted term $P(R|\mathbf{I}_R)$ such that $R \in \mathbf{R}$ and $\mathbf{I}_R \in \mathbf{I}$ from atomic expression A. The status of the atomic expression is updated on lines 31 and 32 to reflect this. If the function fails, it will return A unchanged.

If the innermost while-loop did not iterate completely through the relevant variables, the simplification was not successful in terms of S_j at this point. In this case we reset A to its original state on line 29 and attempt simplification in terms of the next variable. If there are no further variables to be eliminated, the outermost while-loop will also terminate. In the next theorem, we show that Algorithm 1 is both sound and complete in terms of simplification sets. The proof for the theorem can be found in Appendix D.

Theorem 7 *Let G' be a DAG and let G be a subgraph of G' over a vertex set $\mathbf{W}$ with a topological ordering π. Let $A[\mathbf{W}] = \langle \mathbf{T}, \{V_j\}\rangle$ be a π-consistent atomic expression. Then if* SIMPLIFY(A, G, π) *succeeds, it has constructed a collection of simplification sets of A with respect to V_j. Conversely, if there exists a collection of simplifications sets of A with respect to V_j, then* SIMPLIFY(A, G, π) *will succeed.*

4. High Level Algorithms

In this section, we present an algorithm to simplify all atomic expressions in the recursive stack of an expression. We will also provide a simple procedure to simplify quotients defined by two expressions: one representing the numerator and another representing the denominator. In some cases it is also possible to eliminate the denominator by subtracting common terms. First, we present a general algorithm to simplify topologically consistent expressions.

Algorithm 4 Recursive wrapper for the simplification of an expression $B = \langle \mathbf{B}, \mathbf{A}, \mathbf{S}\rangle$ given graph G and topological ordering π.

1: **function** DECONSTRUCT(B, G, π)
2: $\quad\mathbf{R} \leftarrow \emptyset$
3: $\quad$**for** $Y \in \mathbf{A}$ **do**
4: $\quad\quad\langle\{\langle V_1, \mathbf{C}_1\rangle, \ldots, \langle V_n, \mathbf{C}_n\rangle\}, \mathbf{S}_Y\rangle \leftarrow$ SIMPLIFY(Y, G, π)
5: $\quad\quad$**if** $\mathbf{S}_Y = \emptyset$ **then**
6: $\quad\quad\quad\mathbf{A} \leftarrow \mathbf{A} \cup (\bigcup_{i=1}^{n} \{\langle\{\langle V_i, \mathbf{C}_i\rangle\}, \emptyset\rangle\})$
7: $\quad$**for** $\langle \mathbf{B}_X, \mathbf{A}_X, \mathbf{S}_X\rangle \in \mathbf{B}$ **do**
8: $\quad\quad\langle \mathbf{B}_X, \mathbf{A}_X, \mathbf{S}_X\rangle \leftarrow$ DECONSTRUCT($\langle \mathbf{B}_X, \mathbf{A}_X, \mathbf{S}_X\rangle, G$)
9: $\quad\quad$**if** $\mathbf{B}_X = \emptyset$ **and** $\mathbf{S}_X = \emptyset$ **then**
10: $\quad\quad\quad\mathbf{R} \leftarrow \mathbf{R} \cup \{\langle \mathbf{B}_X, \mathbf{A}_X, \mathbf{S}_X\rangle\}$
11: $\quad\quad\quad\mathbf{A} \leftarrow \mathbf{A} \cup \mathbf{A}_X$
12: $\quad\mathbf{B} \leftarrow \mathbf{B} \setminus \mathbf{R}$
13: $\quad$**return** $\langle \mathbf{B}, \mathbf{A}, \mathbf{S}\rangle$

Algorithm 4 begins by simplifying all atomic expressions contained in the expressions. If an atomic expression contains no summations after the simplification but does contain multiple terms, each individual term is converted into an atomic expression of their own. After this, we iterate through all sub-expressions contained in the expression. The purpose of this is to carry out the simplification of every atomic expression in the stack and collect the results into as few atomic expressions as possible. First, we traverse to the bottom of the stack on line 8 by deconstructing sub-expressions until they have no sub-expressions of their own. Afterwards, it must be the case that $\langle \mathbf{B}_X, \mathbf{A}_X, \mathbf{S}_X\rangle$ consists of atomic sub-expressions only.

If $\langle \mathbf{B}_X, \mathbf{A}_X, \mathbf{S}_X\rangle$ contains no summations on line 9 then the atomic expressions contained in this expression do not require an additional expression to contain them, but can instead be transferred to be a part of the expression above the current one in the recursive stack. On line 6 we lift the atomic expressions contained in the atomic sub-expressions up to the current recursion stage.

There is no guarantee, that the resulting atomic expression is still π-consistent after this procedure. The function DECONSTRUCT operates on the principle of simplifying as many atomic expressions as possible, combining the results into new atomic expressions and simplifying them once more. We do not claim that this procedure is complete in a sense that Algorithm 4 would always find the simplest representation for a given expression. This method in nonetheless sound and finds drastically simpler expressions in almost every situation where such an expression exists.

We may also consider quotients often formed by deriving conditional distributions. For this purpose we need a subroutine to extract terms from atomic sub-expression that are independent of the summation index, that is $V_i \notin \mathbf{S}$ and $\mathbf{C}_i \cap \mathbf{S} = \emptyset$.

The procedure of Algorithm 5 is rather straightforward. First, we attempt to simplify B by using DECONSTRUCT on line 2. Next, we simply recurse as deep as possible without encountering a sum in an expression. If a sum is encountered, extraction is attempted. On any stage where a sum was not encountered, we may still have atomic sub-expression

Algorithm 5 Extraction of terms independent of the summation indices from a expression $B = \langle \mathbf{B}, \mathbf{A}, \mathbf{S} \rangle$ given graph G and topological ordering π.

```
1:  function EXTRACT(B, G, π)
2:      B ← DECONSTRUCT(B, G, π)
3:      if S = ∅ then
4:          for X ∈ B do
5:              X ← EXTRACT(X, G, π)
6:          for ⟨T_A, S_A⟩ ∈ A do
7:              if S_A ≠ ∅ then
8:                  A_E ← ∅
9:                  R ← ∅
10:                 for ⟨V, C⟩ ∈ T_A do
11:                     if V ∉ S_A and C ∩ S_A = ∅ then
12:                         A_E ← A_E ∪ {⟨{⟨V, C⟩}, ∅⟩}
13:                         R ← R ∪ {⟨V, C⟩}
14:                 A ← A ∪ A_E
15:                 T_A ← T_A \ R
16:     else
17:         A_E ← ∅
18:         R ← ∅
19:         for ⟨T_A, S_A⟩ ∈ A do
20:             if S_A = ∅ then
21:                 T_A^(1) ← ∅
22:                 T_A^(2) ← ∅
23:                 for ⟨V, C⟩ ∈ T_A do
24:                     T_A^(1) ← T_A^(1) ∪ {V}
25:                     T_A^(2) ← T_A^(2) ∪ C
26:                 if T_A^(1) ∩ S = ∅ and T_A^(2) ∩ S = ∅ then
27:                     A_E ← A_E ∪ {⟨T_A, S_A⟩}
28:                     R ← R ∪ {⟨T_A, S_A⟩}
29:         A ← A \ R
30:         B_E ← {B}
31:         return ⟨B_E, A_E, ∅⟩
```

that contain sums. Because the recursion had reached this far, we know that there are no summations above them in the stack, so we can attempt extraction on them as well.

Algorithm 6 takes two expressions, B_1 and B_2, and removes any sub-expressions and atomic sub-expressions that are shared by B_1 and B_2. This is of course only feasible when the summation sets are empty for both B_1 and B_2. This condition is checked on line 4.

Algorithm 6 Simplification of a quotient P_{B_1}/P_{B_2} given by the values of two expressions $B_1 = \langle \mathbf{B}_1, \mathbf{A}_1, \mathbf{S}_1 \rangle$ and $B_2 = \langle \mathbf{B}_2, \mathbf{A}_2, \mathbf{S}_2 \rangle$ given graph G and topological ordering π.

```
1:  function q-SIMPLIFY(B_1, B_2, G, π)
2:      B_1 ← EXTRACT(B_1, G, π)
3:      B_2 ← EXTRACT(B_2, G, π)
4:      if S_1 ≠ ∅ or S_2 ≠ ∅ then
5:          return ⟨B_1, B_2⟩
6:      i ← 1
7:      while i ≤ |B_1| and |B_1| > 0 and |B_2| > 0 do
8:          for j = 1 : |B_2| do
9:              if B_1i = B_2j then
10:                 B_1 ← B_1 \ {B_1i}
11:                 B_2 ← B_2 \ {B_2j}
12:                 i ← 0
13:                 break
14:         i ← i + 1
15:     i ← 1
16:     while i ≤ |A_1| and |A_1| > 0 and |A_2| > 0 do
17:         for j = 1 : |A_2| do
18:             if A_1i = A_2j then
19:                 A_1 ← A_1 \ {A_1i}
20:                 A_2 ← A_2 \ {A_2j}
21:                 i ← 0
22:                 break
23:         i ← i + 1
24:     return ⟨B_1, B_2⟩
```

5. Examples

In this section we present examples of applying the algorithms of the previous sections. We denote line number y of algorithm x with Ax:y. We begin with a simple example on the necessity of the INSERT procedure in graph G of Figure 3.

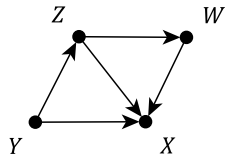


Figure 3: A graph G for the example on the necessity of the insertion procedure.

The causal effect of W on X is identifiable in this graph, and expression

$$\sum_{Z,Y} P(Y)P(Z|Y)P(X|W,Z,Y)$$

is obtained by direct application of the ID algorithm or by the truncated factorization formula for causal effects in Markovian models (Pearl, 2009). We let A be this atomic expression. The topological ordering π is $X > W > Z > Y$ and $\mathbf{M} = \{W\}$. The call to SIMPLIFY(A, G, π) will first attempt simplification in terms of Z, by calling

$$\text{JOIN}(\emptyset, \emptyset, X, \{W, Z, Y\}, Z, \{W\}, G, \pi),$$

which results in $\langle X, \{W, Z, Y\}, \emptyset\rangle$. At the second call

$$\text{JOIN}(\{X\}, \{W, Z, Y\}, Z, Y, Z, \{W\}, G, \pi)$$

we already run into trouble since we cannot find a conditioning set that would allow Z to be joined with $\{X\}$. However, since $\mathbf{M}$ is non-empty and $W \in \{W, Z, Y\}$ and $W \notin \{Z\}$ this means that the next call is

$$\text{INSERT}(\{X\}, \{W, Z, Y\}, W, Z, G, \pi).$$

Insertion fails in this case, as one can see from the fact that no conditioning set exists that would make W conditionally independent of Z. Thus we recurse back to JOIN and back to SIMPLIFY and end up on line A1:15 which breaks out of the while-loop. Thus A cannot be simplified in terms of Z. Simplification is attempted next in terms of Y. The first two calls are in this case

$$\text{JOIN}(\emptyset, \emptyset, X, \{W, Z, Y\}, Y, \{W\}, G, \pi),$$
$$\text{JOIN}(\{X\}, \{W, Z, Y\}, Z, \{Y\}, Y, \{W\}, G, \pi),$$

and in the second call we run into trouble again and have to attempt insertion

$$\text{INSERT}(\{X\}, \{W, Z, Y\}, W, Y, G, \pi).$$

This time we find that we can add a term for W which is $P(W|Z,Y)$ because $(W \perp\!\!\!\perp Y|Z)_G$. The other calls to JOIN also succeed and we can write the value of A as

$$\frac{\sum_{Z,Y} P(Y)P(Z|Y)P(W|Y,Z)P(X|W,Z,Y)}{P(W|Z)}.$$

and complete the summation in terms of Y. After the call to FACTORIZE we are left with the final expression

$$\sum_{Z} P(X|W,Z)P(Z).$$

We continue by considering again graph G depicted in Figure 1. The topological ordering π is $Y > Z_1 > Z_3 > X > Z_2$. Atomic expression A_1 given by

$$\sum_{X,Y} P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2),$$

is a part of the expression to be simplified.

We will first simplify A_1 and take a closer look at how the function JOIN operates. The call to SIMPLIFY(A_1, G, π) will attempt simplification in terms of the set $\{X, Y\}$ in the ordering that agrees with the topological ordering π, which is (Y, X). After initializing the required sets, we find the index of the term with Y as a variable on line 10. There is one missing variable, Z_1, so $\mathbf{M} = \{Z_1\}$ as returned by GET.MISSING on line A1:11. The first call to JOIN results in $\langle Y, \{Z_2, X, Z_3, Z_1\}, \emptyset\rangle$, because line A2:3 is triggered. Condition on line A1:15 is not satisfied since $\mathbf{J}_{\text{new}} = \{Y\} \not\subseteq \emptyset = \mathbf{J}$. Thus we update the status of $\mathbf{J}$ and $\mathbf{D}$ on lines A1:18 and A1:19. Since $\mathbf{R}_{\text{new}} = \emptyset$ on line A1:20 we do not have to update the status of $\mathbf{R}$, $\mathbf{I}$ and $\mathbf{M}$ on lines A1:21, A1:22 and A1:23. The innermost while-loop is now complete and we call FACTORIZE on line A1:27 which succeeds in removing the term $P(Y|Z_2, X, Z_3, Z_1)$ by completing the sum. Now we update the status of the atomic expression on line A1:31 and remove Y from the set of variables to be summed over on line A1:32. The resulting value of the expression at this point is

$$\sum_{X} P(Z_3|Z_2,X)P(X|Z_2)P(Z_2).$$

Next, the summation in terms of X is attempted. JOIN is once again successful, because Z_3 is the first variable to be joined and line A2:3 is triggered. Next we attempt to join the terms $P(Z_3|Z_2, X)$ and $P(X|Z_2)$. Computation of the set $\mathbf{G}$ on line A2:4 results in

$$\{Z_3\}^{\pi} \setminus An^*(X)_G = \{X, Z_2\} \setminus \{X, Z_2\} = \emptyset.$$

The power set computed on line A2:5 contains only the empty set. For $\mathbf{P}_1 = \emptyset$ we have

$$A = (An(X)^*_G \cup \mathbf{P}_1) \triangle \mathbf{D} = (\{X, Z_2\} \cup \emptyset) \triangle \{X, Z_2\} = \emptyset$$

on line 8, and

$$B = (An(X)_G \cup \mathbf{P}_1) \triangle \mathbf{C} = (\{Z_2\} \cup \emptyset) \triangle \{Z_2\} = \emptyset$$

on line 9. The condition on line A2:10 evaluates to true and we return with $\langle \{Z_3, X\}, \{Z_2\}, \emptyset\rangle$. The innermost while-loop terminates allowing the summation over X to be performed. The function FACTORIZE provides us with the final expression

$$P(Z_3|Z_2)P(Z_2). \tag{4}$$

Next, we will consider the full example and see how q-SIMPLIFY is applied. Using the ID algorithm we obtain the causal effect of X on Z_1, Z_2, Z_3 and Y in graph G of Figure 1 and it is

$$P(Z_1|Z_2,X)P(Z_3|Z_2)\frac{\sum_X P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2)}{\sum_{X,Y} P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2)} \times \sum_{X,Z_3,Y} P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2).$$

We will represent this as a quotient of expression using Definition 2. Let A_1 be the atomic expression of the previous example and let A_2 also be an atomic expression given by

$$\sum_{X} P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2),$$

which is essentially the same as A_1, but with the variable Y removed from the summation set $\mathbf{S}$. Similarly, we let A_3 be an atomic expression given by

$$\sum_{X,Z_3,Y} P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2).$$

We also define the atomic expressions A_4 with the value $P(Z_3|Z_2)$ and A_5 with the value $P(Z_1|Z_2,X)$. Now, we define two expressions B_1 and B_2 for the quotient P_{B_1}/P_{B_2} as follows:

$$B_1 = \langle \emptyset, \{A_2, A_3, A_4, A_5\}, \emptyset \rangle, \quad B_2 = \langle \emptyset, \{A_1\}, \emptyset \rangle.$$

We now call q-SIMPLIFY(B_1, B_2, G, π). First, we must trace the calls to EXTRACT for both expressions on lines A6:2 and A6:3. For B_1 and B_2 this immediately results in a call to DECONSTRUCT on line A5:2. First, the function applies SIMPLIFY to each atomic expression contained in the expressions on line A4:4.

Let us first consider the simplification of A_2. As before with A_1, we have that JOIN first succeeds in forming $\langle Y, \{Z_2, X, Z_3, Z_1\}, \emptyset \rangle$, but this time Y is not in the summation set, so we continue. Next, the algorithm attempts to join $P(Y|Z_2,X,Z_3,Z_1)$ with $P(Z_3|Z_2,X)$. The set $\mathbf{G}$ is defined as

$$\{Y\}^\pi \setminus An^*(Z_3)_G = \{Z_3, Z_1, X, Z_2\} \setminus \{Z_3, Z_2\} = \{Z_1, X\}$$

and its subsets are $\{Z_1, X\}$, $\{Z_1\}$, $\{X\}$ and $\emptyset$. For the first subset $\mathbf{P}_1 = \emptyset$ we have that

$$A = (An^*(Z_3) \cup \mathbf{P}_1) \triangle \mathbf{D} = \{Z_2, Z_3\} \triangle \{Z_2, X, Z_3, Z_1\} = \{X, Z_1\}$$

and since $(Y \not\perp\!\!\!\perp X, Z_1 | Z_3, Z_2)_G$ the condition on line A2:10 is not satisfied. We continue with $\mathbf{P}_2 = \{X\}$ and obtain

$$A = (An^*(Z_3) \cup \mathbf{P}_2) \triangle \mathbf{D} = \{X, Z_2, Z_3\} \triangle \{Z_2, X, Z_3, Z_1\} = \{Z_1\}$$

and since $(Y \not\perp\!\!\!\perp Z_1 | X, Z_3, Z_2)_G$ the condition on line A2:10 is still not satisfied. Next, for $\mathbf{P}_3 = \{Z_1\}$ we have

$$A = (An^*(Z_3) \cup \mathbf{P}_3) \triangle \mathbf{D} = \{Z_2, Z_3, Z_1\} \triangle \{Z_2, X, Z_3, Z_1\} = \{X\}$$

and since $(Y \not\perp\!\!\!\perp X | Z_1, Z_3, Z_2)_G$ the condition on line A2:10 is again, not satisfied. Finally, for $\mathbf{P}_4 = \{Z_1, X\}$ we have

$$A = (An^*(Z_3) \cup \mathbf{P}_4) \triangle \mathbf{D} = \{Z_2, X, Z_3, Z_1\} \triangle \{Z_2, X, Z_3, Z_1\} = \{X\}$$

and

$$B = (An(Z_3) \cup \mathbf{P}_4) \triangle \mathbf{C} = \{Z_2, X, Z_1\} \triangle \{Z_2, X\} = \{Z_1\}.$$

Both conditions on line A2:10 are now satisfied ny noting that $(Z_3 \perp\!\!\!\perp Z_1 | X, Z_2)_G$. Afterwards we obtain

$$P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X) = P(Y,Z_3|Z_1,Z_2,X)$$

and continue in an attempt to join the term $P(X|Z_2)$ with this result. The set $\mathbf{G}$ is now defined as

$$\{Y, Z_3\}^\pi \setminus An^*(X)_G = \{Z_1, X, Z_2\} \setminus \{X, Z_2\} = \{Z_1\}$$

and its subsets are $\{Z_1\}$ and $\emptyset$. Starting with $\mathbf{P}_1 = \emptyset$ we have that

$$A = (An^*(X) \cup \mathbf{P}_1) \triangle \mathbf{D} = \{X, Z_2\} \triangle \{Z_1, Z_2, X\} = \{Z_1\}$$

and since $(Y, Z_3 \not\perp\!\!\!\perp Z_1 | X, Z_2)_G$ the condition on line A2:10 is not satisfied. Continuing with $\mathbf{P}_2 = \{Z_1\}$ we have

$$B = (An(X) \cup \mathbf{P}_2) \triangle \mathbf{C} = \{Z_2, Z_1\} \triangle \{Z_2\} = \{Z_1\}.$$

Again, the condition on line A2:10 is not satisfied by noting that $(X \not\perp\!\!\!\perp Z_1 | Z_2)_G$. We have exhausted the possible subsets, which means that we enter the loop on line A2:13 since the set $\mathbf{M} = \{Z_1\}$ is not empty of line A2:12.

In this case INSERT is called to bring Z_1 into the expression because $Z_1 \in \mathbf{D} = \{Z_1, Z_2, X\}$ and $Z_1 \notin \mathbf{C} = \{Z_2\}$. The set $\mathbf{G}$ is constructed on line A3:2 and it is

$$\mathbf{J}^\pi \setminus An^*(Z_1)_G = \{Y, Z_3\}^\pi \setminus \{X, Z_1, Z_2\} = \emptyset.$$

For the only subset $\mathbf{P}_1 = \emptyset$ we have

$$B = (An(Z_1)_G \cup \mathbf{P}_1) = \{X, Z_2\}$$

on line A3:6, and since $(Z_1 \not\perp\!\!\!\perp X | Z_2)_G$ the condition on line A3:7 is not satisfied and we return with $\langle \mathbf{J}, \mathbf{D}, \emptyset \rangle$ unchanged on line 9 of Algorithm 3, which causes JOIN to also return with the same output on line A3:18. The condition on line A1:15 is now satisfied and we cannot simplify A_2.

The atomic expression A_3 can be simplified. First, Y is eliminated exactly as it was removed from A_1. Following the same principle we can see that whenever a variable in the summation set is the largest one in the topological order of the variables contained in the atomic expression, it will be removed successfully. From this we obtain that the value of A_3 is in fact simply $P(Z_2)$. Let us call the atomic expression with this value E, that is $P_E = P(Z_2)$. The atomic expression A_1 can also be simplified, and its value is given by (4). Furthermore, since this value is made of two product terms, it is split into two atomic expressions respectively. Let these be called D_1 and D_2 such that $P_{D_1} = P(Z_3|Z_2)$ and $P_{D_2} = P(Z_2)$.

Applying SIMPLIFY to A_4 and A_5 simply returns the original expressions, since they do not contain any summations and the loop on line A1:3 is never entered. The set of atomic expressions is afterwards updated on line A4:6. Neither B_1 nor B_2 contain any sub-expressions or summations on line A4:9, so DECONSTRUCT(B_1, G, π) returns $\langle \emptyset, \{A_2, E, A_4, A_5\}, \emptyset \rangle$ and DECONSTRUCT(B_2, G, π) returns $\langle \emptyset, \{D_1, D_2\}, \emptyset \rangle$. The lack of summations on line A5:3 of causes EXTRACT to iterate through the atomic expression contained in B_1 and B_2 directly on line A5:6, since neither of them have any sub-expressions of their own.

Only A_2 contains a sum at this point. The iteration over the terms of A_2 on line A5:10 finds that the only term that does not contain X is $P(Z_2)$ on line A5:11. Let us denote the atomic expression with the value $P(Z_2)$ as C_1 and the atomic expression resulting from the extraction as C_2 which now has the value

$$\sum_X P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2).$$

This completes the extraction and results in an expression B_1' such that

$$B_1' = \langle \emptyset, \{C_1, C_2, E, A_4, A_5\}, \emptyset \rangle.$$

The expression B_2 remains unchanged.

q-SIMPLIFY is now able to proceed. Neither B_1' nor B_2 contain sub-expression so the loop on line A6:7 is not entered, and we are only subtracting their common atomic expressions in the loop on line A6:16. It is easy to see that $A_4 = D_1$ and $C_1 = D_2$, so they are removed from both B_1' and B_2. Finally, the expressions corresponding to the numerator and denominator are returned.

To summarize, we began with the expression

$$P(Z_1|Z_2,X)P(Z_3|Z_2)\frac{\sum_X P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2)}{\sum_{X,Y} P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2)} \times \sum_{X,Z_3,Y} P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2)P(Z_2).$$

and successfully simplified it into

$$P(Z_1|Z_2,X)P(Z_2)\sum_X P(Y|Z_2,X,Z_3,Z_1)P(Z_3|Z_2,X)P(X|Z_2).$$

6. Discussion

We have presented a formal definition of topologically consistent atomic expressions and simplification sets and provided a sound and complete algorithm to find these sets for a given expression. We also discussed some general techniques that apply to a more general class of these expressions. Algorithm 7 and Algorithm 8, presented in Appendix A, have been previously implemented in the R package causaleffect (Tikka and Karvanen, 2017). We have updated the package to include all of the simplification procedures presented in this paper and they can be applied to all causal effect and conditional causal effect expressions derived from identification procedures. Our definition of topologically consistent atomic expressions is similar to g-functionals that can be used to characterize identifiability results under special conditions (Shpitser and Tchetgen Tchetgen, 2016).

It is plausible that these procedures could also be extended into other causal inference results, such as formulas for z-identifiability, transportability and meta-transportability of causal effects. The extensions are non-trivial however, since transportability formulas contain terms with distributions from multiple domains and z-identifiable causal effects contain do-operators in the conditioning sets which would require the implementation of the rules of do-calculus into Algorithm 1. Do-calculus consists of three inference rules that can be used to manipulate probabilities involving the do-operator (Pearl, 2009). Currently, we operate only on expressions that do not involve the do-operator. In fact, in our procedure it is not required to know the original causal query that produced the result.

Simpler expressions have many useful properties. They can help in understanding and communicating results and evaluating them saves computational resources. Estimation accuracy can also be improved in some cases when variables that are present in the original expression suffer from missing data or measurement error. One example where the benefits

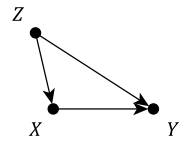


Figure 4: A graph G for a situation where simplification fails

of simplification are realized can be found in (Hyttinen et al., 2015), where expressions of causal effects are derived and repeatedly evaluated for a large number of causal models.

Our approach to simplification stems from the nature of causal effect expressions. In our setting, a question still remains whether simplification sets completely characterize all situations where a variable can be eliminated from an atomic expression. One might also consider simplification in a general setting, where we do not assume topological consistency or any other constraints for the atomic expressions. In this case a 'black box' definition for simplification could be considered, where we simply require that when the sum over a variable of interest is completed we are again left with another atomic expression without this variable in the summation set. This framework is theoretically interesting but we are not aware of any potential applications.

The worst case time complexity of Algorithm 1 is difficult to gauge and is a topic for further research. One can observe that the performance of the algorithm is highly dependent on the size of the differences of the conditioning sets between adjacent terms. Both Algorithm 2 and Algorithm 3 iterate through the subsets of these differences and check d-separation criteria for each subset. Thus dynamic programming solutions could be implemented to further improve performance by collecting the results of these checks. Previously determined conditional independences would not need to be checked again and could be retrieved from memory instead.

In some cases, simplification has some apparent connections to identifiability. Consider the graph G of Figure 4. In this graph the causal effect of X on Y is identifiable, and its expression is

$$\sum_Z P(Y|Z,X)P(Z).$$

If we let Z be an unobserved variable instead, then G depicts the well-known bow-arc graph, where the same causal effect is unidentifiable. This corresponds to an unsuccessful attempt to remove Z from the expression of the causal effect. However, we cannot know beforehand whether an expression for a causal effect is going to be atomic or not, so we cannot use our algorithm to derive identifiability in general.

A reviewer suggested a simplification algorithm where the ID algorithm would be applied to latent projections (Pearl and Verma, 1991) onto the variables to be marginalized. This algorithm would be able to solve many, but not all, simplification tasks. Importantly, in the example presented in Figure 1, we cannot make any variables latent, as we are interested in the causal effect of X on all of the other variables. A reviewer also suggested that simplified

expressions could be categorized into those that are obtained through latent projections and those that are not. This categorization might give additional insight into the topic.

Acknowledgments

We thank the anonymous reviewers for their constructive feedback that greatly helped us to improve this paper. We also thank Lasse Leskelä for his comments.

Appendix A. Topological Consistency of Causal Effect Formulas

We prove the statement that every causal effect formula returned by the algorithms of Shpitser and Pearl (2006a,b) can be represented by using products and quotients of π-consistent atomic expressions such that π is a topological ordering of G.

We use the notation $G[\mathbf{X}]$ to denote an induced subgraph, which is obtained from G by removing all vertices not in $\mathbf{X}$ and by keeping all edges between the vertices of $\mathbf{X}$ in G. Here $G_{\overline{\mathbf{X}},\underline{\mathbf{Z}}}$ means the graph that is obtained from G by removing all incoming edges of $\mathbf{X}$ and all outgoing edges of $\mathbf{Z}$. We say that G is an *I-map* of P if P admits the causal Markov factorization with respect to $G = \langle \mathbf{V}, \mathbf{E} \rangle$, which is

$$P = \prod_{i=1}^{n} P(V_i | Pa^*(V_i)_G) \prod_{j=1}^{k} P(U_j),$$

where $Pa^*(\cdot)$ contains unobserved parents as well.

Consider first lines 2, 3, 4 and 7 of Algorithm 7 where recursive calls occur and let π_r be the topological ordering of the graph in the previous recursion step. Line 2 limits the identification procedure to the ancestors of $\mathbf{Y}$ so we can still obtain an expression that topologically consistent with respect to a topological ordering obtained from π_r by removing non-ancestors. Lines 3 and 4 make no changes to the distribution P and the graph G. On line 7 the induced subgraph $G[\mathbf{S}']$ in the next call is a C-component, but the joint distribution in this case is a π_r-consistent expression

$$P(\mathbf{S}') = \prod_{V_i \in \mathbf{S}'} P(V_i | V_i^{\pi_r} \cap \mathbf{S}', v_i^{\pi_r} \setminus \mathbf{s}'),$$

since every conditioning set is of the form $V_i^{\pi_r}$ when we only consider variables instead of their values, so we obtain

$$P(\mathbf{S}') = \prod_{V_i \in \mathbf{S}'} P(V_i | V_i^{\pi_r}),$$

Furthermore, any expression returned from line 7 will now be π_r-consistent. Thus all recursive calls retain topological consistency with respect to π.

Consider now the non-recursive terminating calls on lines 1 and 6. Consider line 1 first. If line two was triggered previously, we can factorize $P(\mathbf{V})$ in such a way that each variable is conditioned by its ancestors, since the ancestors of ancestors of $\mathbf{Y}$ are by definition ancestors of $\mathbf{Y}$. If line 7 was triggered previously we already know that the joint distribution was previously factorized in a π_r-consistent fashion. If line 3 or 4 was triggered previously, we

know that they have not imposed any changes on P of G. Line 6 clearly produces a π_r consistent end result. Lines 4, 6 and 7 can only produce either products of quotients. By noting that π_r-consistency implies π-consistency, we have that the result of the algorithm can always be represented by using products and quotients of π-consistent atomic expressions.

Algorithm 7 The causal effect of intervention $do(\mathbf{X} = \mathbf{x})$ on $\mathbf{Y}$ (Shpitser and Pearl, 2006a).

INPUT: Value assignments $\mathbf{x}$ and $\mathbf{y}$, joint distribution $P(\mathbf{v})$ and a DAG $G = \langle \mathbf{V}, \mathbf{E} \rangle$. G is an I-map of P.

OUTPUT: Expression for $P_{\mathbf{x}}(\mathbf{y})$ in terms of $P(\mathbf{v})$ or **FAIL**(F, F').

function ID$(\mathbf{y}, \mathbf{x}, P, G)$
1: **if** $\mathbf{x} = \emptyset$, **then**
  return $\sum_{v \in \mathbf{v} \setminus \mathbf{y}} P(\mathbf{v})$.
2: **if** $\mathbf{V} \neq An(\mathbf{Y})_G$, **then**
  return ID$(\mathbf{y}, \mathbf{x} \cap An(\mathbf{Y})_G, P(An(\mathbf{Y})_G), G[An(\mathbf{Y})_G])$.
3: **Let** $\mathbf{W} = (\mathbf{V} \setminus \mathbf{X}) \setminus An(\mathbf{Y})_{G_{\overline{\mathbf{X}}}}$.
  if $\mathbf{W} \neq \emptyset$, **then**
  return ID$(\mathbf{y}, \mathbf{x} \cup \mathbf{w}, P, G)$.
4: **if** $C(G[\mathbf{V} \setminus \mathbf{X}]) = \{G[\mathbf{S}_1], \ldots, G[\mathbf{S}_k]\}$, **then**
  return $\sum_{v \in \mathbf{v} \setminus (\mathbf{y} \cup \mathbf{x})} \prod_{i=1}^{k}$ **ID**$(\mathbf{s}_i, \mathbf{v} \setminus \mathbf{s}_i, P, G)$.
  if $C(G[\mathbf{V} \setminus \mathbf{X}]) = \{G[\mathbf{S}]\}$, **then**
5: **if** $C(G) = \{G\}$, **then**
  throw FAIL$(G, G[\mathbf{S}])$.
6: **if** $G[\mathbf{S}] \in C(G)$, **then**
  return $\sum_{v \in \mathbf{s} \setminus \mathbf{y}} \prod_{V_i \in \mathbf{S}} P(v_i | v_i^{\pi})$.
7: **if** $(\exists \mathbf{S}') \mathbf{S} \subset \mathbf{S}'$ such that $G[\mathbf{S}'] \in C(G)$, **then**
  return ID$(\mathbf{y}, \mathbf{x} \cap \mathbf{s}', \prod_{V_i \in \mathbf{S}'} P(V_i | V_i^{\pi} \cap \mathbf{S}', v_i^{\pi} \setminus \mathbf{s}'), G[\mathbf{S}'])$.

The claim is now apparent for Algorithm 8 since line 2 is eventually called for every conditional causal effect.

Algorithm 8 The causal effect of intervention $do(\mathbf{X} = \mathbf{x})$ on $\mathbf{Y}$ given $\mathbf{Z}$ (Shpitser and Pearl, 2006b).

INPUT: Value assignments $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$, joint distribution $P(\mathbf{v})$ and a DAG $G = \langle \mathbf{V}, \mathbf{E} \rangle$. G is an I-map of P.

OUTPUT: Expression for $P_{\mathbf{x}}(\mathbf{y}|\mathbf{z})$ in terms of $P(\mathbf{v})$ or **FAIL**(F, F').

function IDC$(\mathbf{y}, \mathbf{x}, \mathbf{z}, P, G)$
1: **if** $\exists Z \in \mathbf{Z}$ such that $(\mathbf{Y} \perp\!\!\!\perp Z | \mathbf{X}, \mathbf{Z} \setminus \{Z\})_{G_{\overline{\mathbf{X}},\underline{\mathbf{Z}}}}$ **then**
  return IDC$(\mathbf{y}, \mathbf{x} \cup \{z\}, \mathbf{z} \setminus \{z\}, P, G)$.
2: **else let** $P' =$ **ID**$(\mathbf{y} \cup \mathbf{z}, \mathbf{x}, P, G)$.
  return $P' / \sum_{y \in \mathbf{y}} P'$

Appendix B. Proof of Theorem 6

Proof By direct calculation we obtain

$$\begin{aligned}
P_A &= \sum_{V_j} \prod_{i=1}^{n} P(V_i|\mathbf{C}_i) \\
&= \prod_{V_i<V_j} P(V_i|\mathbf{C}_i) \sum_{V_j} \prod_{V_i \geq V_j} P(V_i|\mathbf{C}_i) \\
&= \prod_{V_i<V_j} P(V_i|\mathbf{C}_i) \sum_{V_j} \frac{P(V_{\pi(p)}, \ldots, V_{\pi(q)}|\mathbf{D})}{\prod_{U\in\mathbf{M}} P(U|\mathbf{E}_U)} \\
&= \prod_{V_i<V_j} P(V_i|\mathbf{C}_i) \frac{P(V_{\pi(p+1)}, \ldots, V_{\pi(q)}|\mathbf{D})}{\prod_{U\in\mathbf{M}} P(U|\mathbf{E}_U)} \\
&= \prod_{V_i<V_j} P(V_i|\mathbf{C}_i) \prod_{V_i>V_j} P(V_i|\mathbf{D}_i) := P_{A'},
\end{aligned}$$

where the sets $\mathbf{D}_i$ are obtained from the factorization of the joint term such that A' is a π^*-consistent where π^* is obtained from π by removing V_j from the ordering. To justify the equalities, we first note that terms of variables $V_i < V_j$ do not contain V_j and can be brought outside the sum.

To obtain the third equality, we multiply by $[\prod_{U\in\mathbf{M}} P(U|\mathbf{E}_U)]/[\prod_{U\in\mathbf{M}} P(U|\mathbf{E}_U)]$ and apply condition (2) of Definition 5 on the right-hand side as licensed by condition (3) of the definition. To obtain the fourth equality, we simply carry out the summation in terms of V_j. Conditions (2) and (3) of Definition 5 make it possible to refactorize the joint term into product terms so that the terms corresponding to variables $U \in \mathbf{M}$ remain unchanged and can be divided out once more. Thus we obtain the last equality, and an expression that no longer contains V_j and has the same value as A. ■

Appendix C. Derivation of the Causal Effect in the Introductory Example

We present the derivation of the causal effect of X on Y, Z_3, Z_2, Z_1 in the graph G of Figure 1 using Algorithm 7. We fix topological ordering of G as $Z_2 < X < Z_1 < Z_3 < Y$. The original call $\mathrm{ID}(\{Y, Z_1, Z_2, Z_3\}, \{X\}, P(\mathbf{V}), G)$ fires line 4 and results in three new recursive calls. We have

$$P_X(Y, Z_3, Z_1, Z_2) = P_{Y,Z_3,X,Z_2}(Z_1) P_{Y,Z_1,X,Z_2}(Z_3) P_{Z_3,Z_1,X}(Y, Z_2), \tag{5}$$

as the graph $G[\mathbf{V}\setminus\{X\}]$ has three C-components formed by the sets $\{Z_1\}$, $\{Z_3\}$ and $\{Y, Z_2\}$, respectively.

The first recursive call $\mathrm{ID}(\{Z_1\}, \{Y, Z_3, X, Z_2\}, P(\mathbf{V}), G)$ fires line 2 because Z_3 and Y are not ancestors of Z_1. The next call $\mathrm{ID}(\{Z_1\}, \{X, Z_2\}, P(Z_1, X, Z_2), G[\{Z_1, X, Z_2\}])$ fires line 6 because $C(G[\{Z_1\}])$ contains only one C-component and it is not part of a larger

C-component in the graph of the current recursion stage. We have

$$P_{Y,Z_3,X,Z_2}(Z_1) = P_{X,Z_2}(Z_1) = P(Z_1|X, Z_2). \tag{6}$$

To obtain $P_{Y,Z_1,X,Z_2}(Z_3)$ we call $\mathrm{ID}(\{Z_3\}, \{Y, Z_1, X, Z_2\}, P(\mathbf{V}), G)$ which also fires line 2 because X, Z_1 and Y and not ancestors of Z_3. Calling $\mathrm{ID}(\{Z_3\}, \{Z_2\}, P(Z_3, Z_2), G[\{Z_3, Z_2\}])$ fires line 6 $C(G[\{Z_3\}])$ contains only one C-component and it is not part of a larger C-component in the graph of the current recursion stage. We have

$$P_{Y,Z_1,X,Z_2}(Z_3) = P_{Z_2}(Z_3) = P(Z_3|Z_2). \tag{7}$$

To obtain the last term we call $\mathrm{ID}(\{Y, Z_2\}, \{Z_3, Z_1, X\}, P(\mathbf{V}), G)$. The subgraph $G[\mathbf{V}\setminus\{Z_3, Z_1, X\}] = G[\{Y, Z_2\}]$ has only one C-component, but it is part of a larger C-component formed by the set $\mathbf{S}' = \{Y, Z_3, X, Z_2\}$ in the current graph G. Line 7 is fired resulting in

$$\mathrm{ID}(\{Y, Z_2\}, \{Z_3, X\}, P(Y|Z_3, Z_1, X, Z_2) P(Z_3|Z_2, X) P(X|Z_2) P(Z_2), G[\mathbf{S}']). \tag{8}$$

This call fires line 2 since X is not an ancestor of Y in the graph $G[\mathbf{S}']$. Letting $\mathbf{T} = \mathbf{S}'\setminus\{X\} = \{Y, Z_3, Z_2\}$ the next call is

$$\mathrm{ID}(\{Y, Z_2\}, \{Z_3\}, \sum_X P(Y|Z_3, Z_1, X, Z_2) P(Z_3|Z_2, X) P(X|Z_2) P(Z_2), G[\mathbf{T}]). \tag{9}$$

This time we trigger line 6 because $G[\mathbf{T}\setminus\{Z_3\}]$ has only one C-component and there is no larger C-component of $G[\mathbf{T}]$ that would contain it. We obtain

$$\begin{aligned}
P_{Z_3,Z_1,X}(Y, Z_2) &= P_{Z_3,Z_1,X}(Y, Z_2) \\
&= P_{Z_3,X}(Y, Z_2) \\
&= P_{Z_3}(Y, Z_2) \\
&= P^*(Y|Z_3, Z_2) P^*(Z_2),
\end{aligned} \tag{10}$$

where P^* is the distribution of the current recursion stage, that is

$$P^*(Y, Z_3, Z_2) = \sum_X P(Y|Z_3, Z_1, X, Z_2) P(Z_3|Z_2, X) P(X|Z_2) P(Z_2).$$

In order to represent the conditional probability on the last line of (10), we write

$$\begin{aligned}
P^*(Y|Z_3, Z_2) P^*(Z_2) &= \frac{P^*(Y, Z_3, Z_2)}{P^*(Z_3, Z_2)} P^*(Z_2) \\
&= \frac{P^*(Y, Z_3, Z_2)}{\sum_Y P^*(Y, Z_3, Z_2)} \sum_{Y,Z_3} P^*(Y, Z_3, Z_2) \\
&= \frac{\sum_X P(Y|Z_2, X, Z_3, Z_1) P(Z_3|Z_2, X) P(X|Z_2) P(Z_2)}{\sum_{X,Y} P(Y|Z_2, X, Z_3, Z_1) P(Z_3|Z_2, X) P(X|Z_2) P(Z_2)} \times \\
&\quad \sum_{X,Z_3,Y} P(Y|Z_2, X, Z_3, Z_1) P(Z_3|Z_2, X) P(X|Z_2) P(Z_2).
\end{aligned} \tag{11}$$

Finally, we gather the results of our subproblems in (6), (7) and (11), and insert them back into the equation in (5) which yields

$$P_X(Y, Z_3, Z_1, Z_2) = P(Z_1|Z_2, X)P(Z_3|Z_2)\times \frac{\sum_X P(Y|Z_2, X, Z_3, Z_1)P(Z_3|Z_2, X)P(X|Z_2)P(Z_2)}{\sum_{X,Y} P(Y|Z_2, X, Z_3, Z_1)P(Z_3|Z_2, X)P(X|Z_2)P(Z_2)}\times \sum_{X,Z_3,Y} P(Y|Z_2, X, Z_3, Z_1)P(Z_3|Z_2, X)P(X|Z_2)P(Z_2)$$

as the formula for the causal effect.

Appendix D. Proof of Theorem 7

Proof (i) Suppose that SIMPLIFY(A, G, π) has returned an expression with variable V_j eliminated. Because the computation completed successfully, we have that each application of JOIN and INSERT succeed. We can rewrite the value of A as

$$\prod_{V_i<V_j} P(V_i|\mathbf{C}_i)\sum_{V_j}\prod_{V_i\geq V_j} P(V_i|\mathbf{C}_i),$$

where the terms $P(V_i|\mathbf{C}_i)$ such that $V_i < V_j$ can be brought outside the sum over V_j, because they cannot contain V_j. The functions JOIN and INSERT use only standard rules of probability calculus, which can be seen on line 10 of Algorithm 2 and line 7 of Algorithm 3, and thus every new formation of a joint distribution $P(\mathbf{J}|\mathbf{D})$ has been valid. Once again we rewrite the value of A as

$$\prod_{V_i<V_j} P(V_i|\mathbf{C}_i)\sum_{V_j} P(\mathbf{J}|\mathbf{D}),$$

which means that condition (2) of Definition 5 is now satisfied, as we have obtained a joint term from the original product terms.Because $V_j \in \mathbf{J}$ we can carry out the summation which yields

$$\prod_{V_i<V_j} P(V_i|\mathbf{C}_i)\cdot P(\mathbf{J}\setminus\{V_j\}|\mathbf{D}),$$

Because Algorithm 1 succeeds, we know that every insertion is canceled out by FACTORIZE. To complete the procedure we obtain a new factorization without V_j resulting in an atomic expression A' that no longer contains V_j. Condition (3) of Definition 5 is satisfied by the definition of INSERT, because the function always checks the conditional independence with the current summation variable on line 7. Both conditions for simplification sets have been satisfied by construction.

(ii) Suppose that there exists a collection of simplification sets of A with respect to V_j. For the sake of clarity, assume further that $V_n = V_j$. This assumption lets us only consider those terms that are relevant to the simplification of V_j, as we can always move conditionally independent terms outside the summation and consider only the expression remaining inside the sum. Let us first assume that $\mathbf{M} = \emptyset$. In this case condition (2) simply reads

$$\prod_{V_i\geq V_j} P(V_i|\mathbf{C}_i) = P(V_j, \ldots, V_1|\mathbf{D}),$$

and that the product terms are a factorization of the joint term. However, we want to show that they also provide a factorization that agrees with the topological ordering. Because A is $\pi-$consistent, for any two variables $V > W$ we have that $\mathbf{C}_W \subseteq V^\pi$ which enables us to consider the summations from V_k up to V_1 for $k = 1, \ldots, j-1$, which results in

$$\sum_{V_k,\ldots,V_1}\prod_{V_i\geq V_j} P(V_i|\mathbf{C}_i) = \sum_{V_k,\ldots,V_1} P(V_j, \ldots, V_1|\mathbf{D}) = P(V_j, \ldots, V_{k+1}|\mathbf{D}).$$

We obtain for $k = j-1, \ldots, 1$

$$\begin{aligned} P(V_j|\mathbf{C}_j) &= P(V_j|\mathbf{D}) \\ P(V_j|\mathbf{C}_j)P(V_{j-1}|\mathbf{C}_{j-1}) &= P(V_j, V_{j-1}|\mathbf{D}) \\ &\vdots \\ P(V_j|\mathbf{C}_j)\cdots P(V_2|\mathbf{C}_2) &= P(V_j, \ldots, V_2|\mathbf{D}) \\ P(V_j|\mathbf{C}_j)\cdots P(V_2|\mathbf{C}_2)P(V_1|\mathbf{C}_1) &= P(V_j, \ldots, V_1|\mathbf{D}). \end{aligned} \tag{12}$$

From the last and second to last equation we can obtain

$$P(V_j, \ldots, V_2|\mathbf{D})P(V_1|\mathbf{C}_1) = P(V_j, \ldots, V_1|\mathbf{D}),$$

and by dividing with the first term from the left hand side we obtain

$$P(V_1|\mathbf{C}_1) = P(V_1|V_j, \ldots, V_2, \mathbf{D}).$$

In fact, we can do this for any two subsequent equations in (12) to obtain

$$P(V_i|\mathbf{C}_i) = P(V_i|V_j, \ldots, V_{i+1}, \mathbf{D}), \quad i = 1, \ldots, j-1$$

Algorithm 1 operates by starting from V_1, so we still have to show it succeeds in constructing the joint term. Using the previous results we can rewrite the original equation as

$$\prod_{V_i\geq V_j} P(V_i|\mathbf{C}_i) = \prod_{V_i\geq V_j} P(V_i|\mathbf{C}_i^*),$$

where $\mathbf{C}_i^* = \mathbf{D}\cup\{V_j, \ldots, V_{i+1}\}$ for $i < j$ and $\mathbf{C}_j^* = \mathbf{D}$. From this we obtain

$$\begin{aligned} P(V_1|\mathbf{C}_1) &= P(V_1|\mathbf{C}_1^*) \\ P(V_1|\mathbf{C}_1^*)P(V_2|\mathbf{C}_2) &= P(V_1, V_2|\mathbf{C}_2^*) \\ &\vdots \\ P(V_1, \ldots, V_{j-1}|\mathbf{C}_{j-1}^*)P(V_j|\mathbf{C}_j) &= P(V_j, \ldots, V_1|\mathbf{C}_j^*). \end{aligned} \tag{13}$$

The function JOIN will succeed every time since the for-loop starting on line 7 of Algorithm 2 will discover the conditional independence properties allowing the previous equalities in (13) to take place. Thus Algorithm 1 will return an atomic expression with the variable V_j eliminated from the summation set.

Assume now that $\mathbf{M} \neq \emptyset$ and let $\mathbf{V} = V[A]$ and. In this case condition (2) allows us to write

$$\prod_{U \in \mathbf{M}} P(U|\mathbf{E}_U) \prod_{V_i \geq V_j} P(V_i|\mathbf{C}_i) = P(\mathbf{V}, \mathbf{M}|\mathbf{D}),$$

and furthermore, we have that these product terms are a factorization of the joint term. First, we aim to reduce the number of variables in $\mathbf{M}$ to be considered. This is done because Algorithm 1 always starts and finishes the construction of the joint term with a variable in $\mathbf{V}$. We categorize each $U \in \mathbf{M}$ into three disjoint sets. We define

$$\mathbf{M}^- := \{U \in \mathbf{M} \mid U \notin \bigcup_{k=1}^{j} \mathbf{C}_k\}, \mathbf{M}^+ := \{U \in \mathbf{M} \mid U \in \bigcap_{k=1}^{j} \mathbf{C}_k\} \text{ and}$$

$$\mathbf{M}^* := \mathbf{M} \setminus (\mathbf{M}^- \cup \mathbf{M}^+).$$

First, we show that we can ignore variables in $\mathbf{M}^-$ by obtaining a new factorization without them. It follows from the definition of $\mathbf{M}^-$ and (2) that we can compute the marginalization as follows

$$\begin{aligned} P(\mathbf{V}, \mathbf{M} \setminus \mathbf{M}^-|\mathbf{D}) &= \sum_{U \in \mathbf{M}^-} P(\mathbf{V}, \mathbf{M}|\mathbf{D}) \\ &= \sum_{U \in \mathbf{M}^-} \prod_{U \in \mathbf{M}} P(U|\mathbf{E}_U) \prod_{V_i \geq V_j} P(V_i|\mathbf{C}_i) \\ &= \prod_{V_i \geq V_j} P(V_i|\mathbf{C}_i) \sum_{U \in \mathbf{M}^-} \prod_{U \in \mathbf{M}} P(U|\mathbf{E}_U) \\ &= \prod_{U \in \mathbf{M} \setminus \mathbf{M}^-} P(U|\mathbf{E}_U) \prod_{V_i \geq V_j} P(V_i|\mathbf{C}_i). \end{aligned}$$

We have a new factorization without any variables in $\mathbf{M}^-$. Similarly, we can eliminate the variables in $\mathbf{M}^+$ from our factorization. It follows from the definition of $\mathbf{M}^+$ that for all $U \in \mathbf{M}^+$ we have that $\mathbf{E}_U \subseteq \mathbf{D}$. From this we obtain

$$\prod_{U \in \mathbf{M}^+} P(U|\mathbf{E}_U) = P(\mathbf{M}^+|\mathbf{D}).$$

We can now write

$$\begin{aligned} P(\mathbf{V}, \mathbf{M}^*|\mathbf{D}, \mathbf{M}^+) &= \frac{P(\mathbf{V}, \mathbf{M} \setminus \mathbf{M}^-)}{P(\mathbf{M}^+|\mathbf{D})} \\ &= \frac{\prod_{U \in \mathbf{M} \setminus \mathbf{M}^-} P(U|\mathbf{E}_U) \prod_{V_i \geq V_j} P(V_i|\mathbf{C}_i)}{\prod_{U \in \mathbf{M}^+} P(U|\mathbf{E}_U)} \\ &= \prod_{U \in \mathbf{M}^*} P(U|\mathbf{E}_U) \prod_{V_i \geq V_j} P(V_i|\mathbf{C}_i). \end{aligned}$$

Thus it suffices to consider the factorization given by

$$\prod_{U \in \mathbf{M}^*} P(U|\mathbf{E}_U) \prod_{V_i \geq V_j} P(V_i|\mathbf{C}_i) = P(\mathbf{V}, \mathbf{M}^*|\mathbf{D}^*), \tag{14}$$

where $\mathbf{D}^* = \mathbf{D} \cup \mathbf{M}^+$.

Next, we will order the variables in $\mathbf{M}^*$. For each $U \in \mathbf{M}^*$ we find the largest index $u \in \{1, \ldots, j-1\}$ such that $U \in \mathbf{C}_u$. This choice is well defined, since by definition at least one such index exists. Furthermore, as the product terms in (14) are a factorization of the joint term, the conditioning sets are increasing and we have that $U \notin \mathbf{C}_i$ for all $i \geq u+1$. In the case that multiple variables $U_i \in \mathbf{M}^*$ for some set of indices $i \in \mathbf{I}$ share the same index u, we may redefine M^* such that $U_i, i \in \mathbf{I}$ are replaced by a single variable U_I such that $\prod_{i \in I} P(U_i|\mathbf{E}_{U_i}) = P(U_I|\mathbf{E}_{U_I})$, where $\mathbf{E}_{U_I} = \cap_{i \in \mathbf{I}} \mathbf{E}_{U_i}$. Thus we can assume that for any two variables $U_1, U_2 \in \mathbf{M}^*$ we have that $u_1 \neq u_2$. We can now order the variables in $\mathbf{M}^*$ by their respective indices u such that $U_1 > U_2 > \ldots > U_m$ and $u_1 < u_2 < \ldots < u_m$.

Nest we will extend the ordering to include all of the variables in the set $\mathbf{V}$. We let $\mathbf{Q} := \mathbf{V} \cup \mathbf{M}^*$ and find an ordering of this set such that it agrees with induced ordering ω of the variables in $\mathbf{V}$ and with the ordering of the indices $u_1, \ldots, u_m$. A new factorization given by this ordering can be defined as follows:

$$Q_k = \begin{cases} V_{k-m} & k > u_m, \\ V_{k-l} & u_l < k < u_{l+1}, \\ V_k & k < u_1, \\ U_l & k = u_l. \end{cases} \quad \mathbf{D}_k = \begin{cases} \mathbf{C}_{k-m} & k > u_m, \\ \mathbf{C}_{k-l} & u_l < k < u_{l+1}, \\ \mathbf{C}_k & k < u_1, \\ \mathbf{E}_{U_l} & k = u_l. \end{cases}$$

We can now rewrite the factorization of (14) as

$$\prod_{k=1}^{n+m} P(Q_k|\mathbf{D}_k) = P(\mathbf{Q}|\mathbf{D}^*), \tag{15}$$

We can now apply the same procedures as in the case of $\mathbf{M} = \emptyset$ with the exception that INSERT succeeds where JOIN fails with terms containing Q_k and Q_{k+1} when $k = l - 1$ for all $l = 1, \ldots, m$. The success of INSERT is guaranteed by condition (3), as the function will find this conditional independence on line 10 of Algorithm 3. Also, FACTORIZE will remove all additional terms that were introduced in the process, which is made possible by condition (3) and the definition of the factorization of $P(\mathbf{Q}|\mathbf{D}^*)$. After the summation over V_j is carried out, the conditional independence between V_j and the variables $U \in \mathbf{M}^*$ ensures that their respective terms are equal to the original factorization before the summation was carried out when the new factorization is constructed so that it agrees with the ordering of the set $\mathbf{Q}$. Thus an atomic expression is returned with the variable V_j eliminated with the same value as the original atomic expression. ■

References

D. H. Bailey, Borwein J. M., and D. A. Kaiser. Automated simplification of large symbolic expressions. *Journal of Symbolic Computation*, 60:120–136, 2014.

E. Bareinboim and J. Pearl. Causal inference by surrogate experiments: z-identifiability. In N. de Freitas and K. Murphy, editors, *Proceedings of the Twenty-Eight Conference on Uncertainty in Artificial Intelligence*, pages 113–120. AUAI Press, 2012.

E. Bareinboim and J. Pearl. Meta-transportability of causal effects: a formal approach. In *Proceedings of the 16th International Conference on Artificial Intelligence and Statistics*, pages 135–143, 2013a.

E. Bareinboim and J. Pearl. A general algorithm for deciding transportability of experimental results. *Journal of Causal Inference*, 1:107–134, 2013b.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

J. Carette. Understanding expression simplification. In *Proceedings of the 2004 International Symposium on Symbolic and Algebraic Computation*, pages 72–79, New York, 2004. ACM.

D. Geiger, T. Verma, and J. Pearl. Identifying independence in Bayesian networks. *Networks*, 20(5):507–534, 1990.

Y. Huang and M. Valtorta. Pearl's calculus of intervention is complete. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, pages 217–224. AUAI Press, 2006.

A. Hyttinen, F. Eberhardt, and M. Järvisalo. Do-calculus when the true graph is unknown. In *Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence*, pages 395–404. AUAI Press, 2015.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, 2009.

S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B (Methodological)*, 50(2):157–224, 1988.

Maxima. Maxima, a computer algebra system. version 5.34.1, 2014. URL `http://maxima.sourceforge.net/`.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, 1988.

J. Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, New York, 2nd edition, 2009.

J. Pearl and T. S. Verma. A theory of inferred causation. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Second International Conference*, pages 441–452, 1991.

I. Shpitser and J. Pearl. Identification of joint interventional distributions in recursive semi-Markovian causal models. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 2*, pages 1219–1226. AAAI Press, 2006a.

I. Shpitser and J. Pearl. Identification of conditional interventional distributions. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, pages 437–444. AUAI Press, 2006b.

I. Shpitser and E. Tchetgen Tchetgen. Causal inference with a graphical hierarchy of interventions. *Annals of Statistics*, 44(6):2433–2466, 2016.

I. Shpitser, T. S. Richardson, and J. M. Robins. An efficient algorithm for computing interventional distributions in latent variable causal models. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, pages 661–670. AUAI Press, 2011.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT press, 2nd edition, 2000.

S. Tikka and J. Karvanen. Identifying causal effects with the R package causaleffect. *Journal of Statistical Software*, 76(12):1–30, 2017.

Wolfram Research Inc. Mathematica, version 10.3, 2015.

Journal of Machine Learning Research 18 (2017) 1-22 Submitted 4/16; Published 4/17

Nearly optimal classification for semimetrics*

Lee-Ad Gottlieb leead@ariel.ac.il
Department of Computer Science
Ariel University
Ariel, Israel

Aryeh Kontorovich karyeh@cs.bgu.ac.il
Department of Computer Science
Ben-Gurion University
Beer Sheva, Israel

Pinhas Nisnevitch pinhasn@mail.tau.ac.il
Department of Computer Science
Tel-Aviv University
Tel-Aviv, Israel

Editor: Kevin Murphy

Abstract

We initiate the rigorous study of classification in semimetric spaces, which are point sets with a distance function that is non-negative and symmetric, but need not satisfy the triangle inequality. We define the *density dimension* dens and discover that it plays a central role in the statistical and algorithmic feasibility of learning in semimetric spaces. We compute this quantity for several widely used semimetrics and present nearly optimal sample compression algorithms, which are then used to obtain generalization guarantees, including fast rates.

Our claim of near-optimality holds in both computational and statistical senses. When the sample has radius R and margin γ, we show that it can be compressed down to roughly $d = (R/\gamma)^{\mathsf{dens}}$ points, and further that finding a significantly better compression is algorithmically intractable unless P=NP. This compression implies generalization via standard Occam-type arguments, to which we provide a nearly matching lower bound.

Keywords: semimetric, classification, compression, generalization

1. Introduction

The problem of learning in non-metric spaces has been of significant recent interest, being the subject of a 2010 COLT workshop and a central topic of all three SIMBAD conferences. In this paper, we initiate the study of efficient statistical learning in *semimetric* spaces, which are point sets endowed with a distance function that is non-negative and symmetric but may not satisfy the triangle inequality (Wilson, 1931)[1]. Without the latter, quite a bit of structure is lost — for example, semimetric spaces admit convergent sequences without a Cauchy subsequence (Burke, 1972). We are not aware of any rigorous learning results in semimetric spaces prior to this work.

Background and motivation. Much of the existing machinery for classification algorithms, as well as generalization bounds, depends strongly on the data residing in a Hilbert space. For some important applications, this structural constraint severely limits the applicability of existing methods. Indeed, it is often the case that the data is naturally endowed with some distance function strongly dissimilar to the familiar Euclidean norm.

Consider images, for example. Although these can be naively represented as coordinate-vectors in $\mathbb{R}^d$, the Euclidean (or even ℓ_p) distance between the representative vectors does not correspond well to the one perceived by human vision. Instead, the earthmover distance is commonly used in vision applications (Rubner et al., 2000). Yet representing earthmover distances using any fixed ℓ_p norm unavoidably introduces very large interpoint distortion (Naor and Schechtman, 2007), potentially corrupting the data geometry before the learning process has even begun. Nor is this issue mitigated by kernelization, as kernels necessarily embed the data in a Hilbert space, again incurring the aforementioned distortion. A similar issue arises for strings: These can be naively treated as vectors endowed with different ℓ_p metrics, but a much more natural metric over strings is the edit distance, which is similarly known to be strongly non-Euclidean (Andoni and Krauthgamer, 2010). Additional limitations of kernel methods are articulated in Balcan et al. (2008b).

These concerns have led researchers to seek out algorithmic and statistical approaches that apply in greater generality. A particularly fruitful recent direction has focused on metric spaces, which are point sets endowed with a distance function that is non-negative and symmetric, and also satisfies the triangle equality. While metric spaces are significantly more general than Hilbertian ones, they still do not capture many common distance functions used by practitioners. These non-metric distances include the Jensen-Shannon divergence, which appears in statistical applications (Fuglede and Topsøe, 2004; Goodfellow et al., 2014), the k-median Hausdorff distances, and the ℓ_p distances with $0 < p < 1$, which appear in vision applications (M. Dubuisson, A. Jain, 1994; Jacobs et al., 2000) — all of which are semimetrics. An additional line of work (M. Dubuisson, A. Jain, 1994; Jacobs et al., 2000, 1998; Weinshall et al., 1998) underscored the effectiveness of non-metric distances in various applications (mainly vision), and among these, semimetrics again play a prominent role (Basri et al., 1995; Cox et al., 1996; Gdalyahu and Weinshall, 1999; Huttenlocher et al., 1993; Jain and Zongker, 1997; J. Puzicha, J. Buhmann, Y. Rubner, C. Tomasi, 1999).

Main results. We initiate the rigorous study of classification for semimetric spaces. We define the *density dimension* ($\mathsf{dens} = \mathsf{dens}(\mathcal{X})$) of a semimetric space $\mathcal{X}$ as the logarithm of the *density constant* $\mu = \mu(\mathcal{X})$, which intuitively is the smallest number such that any r-radius open ball in $\mathcal{X}$ contains at most μ points at mutual interpoint distance at least $r/2$; a formal definition is given in Equation (2). We then demonstrate that dens plays a central role in the statistical and algorithmic feasibility of learning in this setting by showing that it controls the packing numbers of $\mathcal{X}$. Crucially for learning, this insight implies that there is

*. An extended abstract of this paper appeared in AISTATS 2016. L. Gottlieb was supported in part by the Israel Science Foundation (grant No. 755/15). A. Kontorovich was partially supported by the Israel Science Foundation (grants No. 1141/12 and 755/15) and a Yahoo Faculty award.

1. Some authors use the term "semimetric" to mean *pseudometrics*. These preserve much of the structure of metrics, the only difference being that they allow distinct points to have distance 0. Our usage appears to be more standard.

©2017 Lee-Ad Gottlieb,Aryeh Kontorovich, Pinhas Nisnevitch.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-217.html.

one standard technique that survives violations of the triangle inequality — namely, sample compression. Denoting by R and γ the sample radius and margin, respectively, we can achieve the latter by extracting a γ-net from the sample (Theorem 2). This compresses the sample from size n to $(R/\gamma)^{O(\mathsf{dens})}$, which is nearly optimal unless P=NP (Theorem 4).

On the statistical front, we give a compression-based generalization bound that smoothly interpolates between the consistent $(R/\gamma)^{O(\mathsf{dens})}/n$ and agnostic $\sqrt{(R/\gamma)^{O(\mathsf{dens})}/n}$ decay regimes (Theorem 8). This "fast rate" holds for general compression schemes. Applied to margin-based semimetric sample-compression schemes, it becomes amenable to efficient Structural Risk Minimization. The lower bound in Theorem 11 shows that even under margin assumptions, there exist adversarial distributions forcing the sample complexity to be exponential in dens.

To demonstrate the applicability of our framework, we compute the density dimension of the three popular semimetrics enumerated above: Jensen-Shannon divergence, ℓ_p distances with $0 < p < 1$, and k-median Hausdorff distances (Theorem 13). Along the way, we discover that the latter (for $k = 1$) is in fact universal for *all* semimetrics; this surprising fact may be of independent interest (Lemma 12).

Related work. In a series of papers, Balcan et al. (2008c,a,b) developed a theory of learning with similarity functions, which resemble kernels but relax the requirement of being positive definite. Learning is accomplished by embedding the data into an appropriate Euclidean space and performing large-margin separation. Hence, this approach effectively extracts the implicit Euclidean structure encoded in the similarity function, but does not seem well-suited for inherently non-Euclidean data. In Wang et al. (2007) this framework was extended to dissimilarity functions, obtaining analogous results.

For metric spaces, it is known that a sample of size exponential in the doubling dimension (ddim) suffices to achieve low generalization error (von Luxburg and Bousquet, 2004; Gottlieb et al., 2014a; Shalev-Shwartz and Ben-David, 2014; Kontorovich and Weiss, 2014), and that exponential dependence on ddim is in general unavoidable (Shalev-Shwartz and Ben-David, 2014). As for algorithmic runtimes, the naive nearest-neighbor classifier evaluates queries in $O(n)$ time (where n is the sample size); however, an approximate nearest neighbor can be found in time $2^{O(\mathsf{ddim})} \log n$. If one desires runtimes depending not on n but on the geometry (say, margin γ) of the data, one may achieve a sample compression scheme of size $\gamma^{-O(\mathsf{ddim})}$, and it is NP-hard to achieve a significantly better compression (Gottlieb et al., 2014b).

As we show in the Appendix, the above results characterizing learning in metric space do not carry over to semimetrics. More precisely, the doubling dimension of a semimetric does not control its packing numbers, as it does in metric spaces. Although we succeed in showing that the density constant does indeed control the packing numbers even in semimetrics, this does not necessarily imply portability of learning algorithms for metric spaces into semimetrics. For example, although the nearest-neighbor classifier is still well-defined in semimetric spaces, and may naively be evaluated on queries in $O(n)$ time, relaxing to approximate nearest neighbors no longer provides the exponential speedup in query time that it does in metric spaces (Lemma 17). Simply put, without the triangle inequality, the hierarchy-based search methods, such as Beygelzimer et al. (2006); Gottlieb et al. (2014a) and related approaches, all break down.

Paper outline. After presenting our basic definitions in Section 2, we give packing bounds and net-construction algorithms in Section 3. In Section 4 we give upper and lower bounds on sample complexity for learning in semimetrics. The density dimension of some common semimetircs is computed in Section 5. The Appendix contains a detailed discussion demonstrating the inapplicability of existing metric-space techniques to the semimetric case.

2. Preliminaries

Semimetric spaces. Throughout this paper, our instance space $\mathcal{X}$ will be endowed with a semimetric $\rho : \mathcal{X} \times \mathcal{X} \to [0, \infty)$, which is a non-negative symmetric function verifying $\rho(x, x') = 0 \iff x = x'$ for all $x, x' \in \mathcal{X}$. If the semimetric space $(\mathcal{X}, \rho)$ additionally satisfies the triangle inequality, $\rho(x, x') \leq \rho(x, x'') + \rho(x'', x')$ for all $x, x', x'' \in \mathcal{X}$, then ρ is a *metric*. The distance between two sets A, B in a semimetric space is defined by $\rho(A, B) = \inf_{x \in A, x' \in B} \rho(x, x')$. For $x \in \mathcal{X}$ and $r > 0$, denote by $B_r(x) = \{y \in \mathcal{X} : \rho(x, y) < r\}$ the open r-ball about x. The *radius* of a set is the radius of the smallest ball containing it, $\mathrm{rad}(A) = \inf\{r > 0 : \exists x \in A, A \subseteq B_r(x)\}$ and $\mathrm{diam}(A) := \sup_{x,x' \in A} \rho(x, x')$.

Doubling and density dimensions. Let $\lambda = \lambda(\mathcal{X})$ be the smallest number such that every open ball in $\mathcal{X}$ can be covered by λ open balls of half the radius, where all balls are centered at points of $\mathcal{X}$. Formally,

$$\lambda(\mathcal{X}) = \min\{\lambda \in \mathbb{N} : \forall x \in \mathcal{X}, r > 0 \; \exists x_1, \ldots, x_\lambda \in \mathcal{X} : B_r(x) \subseteq \cup_{i=1}^{\lambda} B_{r/2}(x_i)\}.$$

Then λ is the *doubling constant* of $\mathcal{X}$, and the *doubling dimension* of $\mathcal{X}$ is $\mathsf{ddim}(\mathcal{X}) = \log_2 \lambda$.

An r-*net* of a set $A \subseteq \mathcal{X}$ is any *maximal* subset A having mutual interpoint distance at least r. The r-packing number $\mathcal{M}(r, A)$ of A is the maximum size of any r-net of A:

$$\mathcal{M}(r, A) = \max\{|E| : E \subseteq A, (x, y \in E) \wedge (x \neq y) \implies \rho(x, y) \geq r\}. \tag{1}$$

The *density constant* $\mu(\mathcal{X})$ was defined in Gottlieb and Krauthgamer (2013) as the smallest number such that any open r-radius ball in $\mathcal{X}$ contains at most μ points at mutual interpoint distance at least $r/2$:

$$\mu(\mathcal{X}) = \min\{\mu \in \mathbb{N} : (x \in \mathcal{X}) \wedge (r > 0) \implies \mathcal{M}\left(\frac{r}{2}, B_r(x)\right) \leq \mu\}, \tag{2}$$

and we define the *density dimension* of $\mathcal{X}$ by $\mathsf{dens}(\mathcal{X}) = \log_2 \mu(\mathcal{X})$. A closely related notion was introduced by Vol'berg and Konyagin (1987) under the term *uniform metric dimension*.

An important property of the density dimension is that it is *hereditary*: for $S \subset \mathcal{X}$, we have $\mu(S) \leq \mu(\mathcal{X})$; the doubling dimension is only approximately hereditary (Gottlieb and Krauthgamer, 2013).

It will be convenient to define $\mathrm{Log}\,(x) := \log_2 \lceil x \rceil$, and we will make frequent use of the identity

$$\mu(S)^{\mathrm{Log}(\alpha)} = \lceil \alpha \rceil^{\mathsf{dens}(S)}. \tag{3}$$

Learning model. We work in the standard *agnostic* learning model (Mohri et al., 2012; Shalev-Shwartz and Ben-David, 2014), whereby the learner receives a sample S consisting of n labeled examples (X_i, Y_i), drawn iid from an unknown distribution over $\mathcal{X} \times \{-1, 1\}$. All subsequent probabilities and expectations will be with respect to this distribution. Based on the training sample S, the learner produces a *hypothesis* $h : \mathcal{X} \to \{-1, 1\}$, whose *empirical error* is defined by $\widehat{\mathrm{err}}(h) = n^{-1} \sum_{i=1}^{n} \mathbf{1}_{\{h(X_i) \neq Y_i\}}$ and whose *generalization error* is defined by $\mathrm{err}(h) = \mathbb{P}(h(X) \neq Y)$.

Sub-sample, margin, and induced 1-NN. In a slight abuse of notation, we will blur the distinction between $S \subset \mathcal{X}$ as a collection of points in a semimetric space and $S \in (\mathcal{X} \times \{-1, 1\})^n$ as a sequence of labeled examples. Thus, the notion of a *sub-sample* $\tilde{S} \subset S$ partitioned into its positively and negatively labeled subsets as $\tilde{S} = \tilde{S}_+ \cup \tilde{S}_-$ is well-defined. The *margin* of $\tilde{S}$, defined by $\mathrm{marg}(\tilde{S}) = \rho(\tilde{S}_+, \tilde{S}_-)$, is the minimum distance between a pair of opposite-labeled points (see Fig. 1). In degenerate cases where one of $\tilde{S}_+, \tilde{S}_-$ is empty, $\mathrm{marg}(\tilde{S}) = \infty$. (For ease of presentation, we assume that the margin is strictly less than $\mathrm{rad}(S)$, and so $\mathrm{rad}(S)/\mathrm{marg}(S) > 1$. In the case of equality, substituting any value less than the margin will cause the relevant claim to hold.) A sub-sample $\tilde{S}$ naturally induces the 1-NN classifier $h_{\tilde{S}}$, via $h_{\tilde{S}}(x) = \mathrm{sign}(\rho(x, \tilde{S}_-) - \rho(x, \tilde{S}_+))$.

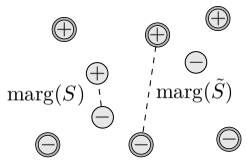


Figure 1: In this example, the sub-sample $\tilde{S} \subset S$ is indicated by double circles. It is always the case that $\mathrm{marg}(\tilde{S}) \geq \mathrm{marg}(S)$.

The problem of *nearest-neighbor condensing* is to produce the minimal subsample $\tilde{S} \subset S$ so that the 1-NN classifier $h_{\tilde{S}}$ is *consistent* with S, i.e. has zero training error. In the *inconsistent* version of this problem, one is given a parameter $\varepsilon > 0$ and tasked with producing a minimal subsample $\tilde{S} \subset S$ whose induced $h_{\tilde{S}}$ has training error at most ε.

3. Packing bounds and algorithms

The central contribution of this section is the following lemma, which demonstrates that for a semimetric space, a bound on its density dimension implies one on its packing numbers.

Lemma 1 *For any point set S in a semimetric space $\mathcal{X}$ and $r > 0$, the size of any r-net of S is $k = \lceil \mathrm{rad}(S)/r \rceil^{\mathsf{dens}(S)}$, and furthermore, such an r-net can be computed in time $O(k|S|)$.*

Proof To bound the size of a maximal r-net $C \subset S$, suppose its radius is R. Partition C into clusters by extracting from C an arbitrary net D with minimum interpoint distance $R/2$, and assigning each point $p \in C$ to a cluster centered at the nearest neighbor of p in

D. Then apply the procedure recursively to each cluster, halving the previous radius, until reaching point sets with minimum interpoint distance at least r. By repeatedly applying the definition of the density constant, the size of C is bounded by $\mu(S)^{\mathrm{Log}(\mathrm{rad}(S)/r)} = \lceil \mathrm{rad}(S)/r \rceil^{\mathsf{dens}(S)}$.

The actual r-net is constructed in a greedy fashion. Initialize set $C = \emptyset$, and for every point in S, add it to C if its closest neighbor in C is at distance r or greater. Since $|C| \leq k$, the total runtime is $O(k|S|)$. See Algorithm 1. ■

Algorithm 1 Brute-force net construction

Require: sample S, margin r
Ensure: C is an r-net for S
  for $x \in S$ **do**
    if $\rho(x, C) \geq r$ **then**
      $C = C \cup \{x\}$
    end if
  end for

Having demonstrated the existence of a small r-net, we can now consider the problems of producing both consistent (lossless) and inconsistent (lossy) 1-NN classifiers for the sample (see Section 2).

Consistent case. For a labeled sample S, recall that the *margin* of S is the minimum distance between oppositely labeled points in S, as defined formally in Section 2. The margin of a given sample can be computed in time $\Theta(|S|^2)$ by considering all pairs of points. We begin with a consistent classifier, whose generalization performance with explicit constants is analyzed in Theorem 7(i). Informally, the latter states that a 1-nearest neighbor classifier induced by a sub-sample of size k has generalization error $O\left(\frac{k \log n + \log \frac{1}{\delta}}{n}\right)$.

Theorem 2 *Let S be a sample set equipped with a semimetric distance function, and let the margin $\gamma = \mathrm{marg}(S)$ be given. In time $O(k|S|)$ we can construct a nearest-neighbor classifier that achieves zero training error on S, where $k = (\mathrm{rad}(S)/\gamma)^{O(\mathsf{dens}(S))}$. The evaluation on a test point requires $\Theta(k)$ distance computations, and with probability $1 - \delta$, the resulting classifier has generalization error $O\left(\frac{k \log n + \log \frac{1}{\delta}}{n}\right)$.*

Proof We build a γ-net C for S in time $O(k|S|)$, as in Lemma 1. Since γ is the margin, by construction every point in S has the same label as its nearest neighbor in C, and so the nearest neighbor classifier with respect to C has zero sample error.

Given a test point x, we assign it the same label as its nearest neighbor in C. Then the generalization bounds follow from Theorem 7(i). For the runtime, $O(k)$ operations are clearly sufficient to find the nearest neighbor (see also Lemmas 16 and 17). ■

Remark 3 *If the margin is not known in advance, then it must be computed, and the runtime in Theorem 2 grows to $O(n^2)$. In this case we can give an alternate construction*

that achieves the runtime of $O(k|S|)$ of the Theorem. Extract sets $S_+, S_- \in S$ of oppositely labeled points in S, and for each set build a subset in a manner similar to the proof of Lemma 1: Let R be the maximum among the radii of S_+, S_-. Partition S_+ into clusters by extracting from S_+ an arbitrary net D with minimum interpoint distance $R/2$, and assigning each point $p \in S_+$ to a cluster centered at the nearest neighbor of p in D. Then apply the procedure recursively to each cluster in D, halving the previous radius. This construction is done to S_- in parallel, and terminates when the union of the subsets for S_+ and S_- is consistent with S. This must occur within $O(\log(\text{rad}(S)/\text{marg}(S)))$ iterations, producing a consistent set of size k in time $O(kn \log(\text{rad}(S)/\text{marg}(S))) = O(kn)$ (where equality follows from the $\log(\text{rad}(S)/\text{marg}(S))$ term being subsumed in the definition of k).

The procedure in Theorem 2 compresses S, producing a consistent sub-sample C. Immediate from the theorem is that the smaller the compressed set C, the better the generalization bounds of the classifier. However, as Gottlieb et al. (2014b) recently demonstrated, even in metric spaces, it is NP-hard to approximate the size of the minimum consistent subset to within a factor $2^{O(\mathsf{ddim}(S)\log(\text{rad}(S)/\text{marg}(S)))^{1-o(1)}} = 2^{O(\mathsf{dens}(S)\log(\text{rad}(S)/\text{marg}(S)))^{1-o(1)}}$ (where the equality follows from Lemma 15). This means that choosing the net of Lemma 1 is close to the optimal construction for a consistent subset of S.

Inconsistent case. It is natural to ask whether allowing the classifier nonzero sample error results in improved generalization bounds. This is indeed generally the case, as the bound in Theorem 8 indicates. Informally, the latter shows that a 1-nearest neighbor classifier induced by a sub-sample of size k with sample error ε has generalization error $Q(k,\varepsilon) = O\left(\varepsilon + \frac{1}{n}\log\frac{n^k}{\delta} + \sqrt{\frac{\varepsilon}{n}\log\frac{n^k}{\delta}}\right)$. Optimizing this bound is an instance of Structural Risk Minimization (SRM). Unfortunately, we can show SRM to be infeasible for this problem, and that the generalization guarantees of Theorem 2 are nearly the best that can be obtained by way of Theorem 8:

Theorem 4 *Given a set S equipped with a metric or semimetric distance function, let $S^* \subset S$ be a sub-sample for which the generalization bound $Q(k,\varepsilon)$ in Theorem 8 (for a fixed constant δ) is minimized. Then it is NP-hard to compute any subset of S achieving a generalization bound within factor $2^{O((\mathsf{dens}(S)\log(\text{rad}(S)/\text{marg}(S)))^{1-o(1)}}$ of the generalization bound induced by S^*.*

Proof The proof is via reduction from the minimum consistent subset problem mentioned above. Fix the confidence level δ in the bound, let T be an instance of the minimum consistent subset problem, and put $m = |T|$. For some large value p, replace each point $t_i \in T$ with a (similarly labeled) set of p points $s_{i,1}, \ldots, s_{i,p}$ obeying the line metric, with $\rho(s_{i,a}, s_{i,b}) = \phi|a-b|$ for an infinitesimally small ϕ. Put $\rho(s_{i,a}, s_{j,b}) = \rho(t_i, t_j)$. The new set is S, with $n = |S| = pm$.

Consider a subset $S' \subset S$. If the 1-NN rule on S' misclassifies a point of S, say $s_{i,a}$, then in fact it must misclassify all p points $s_{i,b}$, $b \in [1,p]$. So an inconsistent subset of S achieves a value of $Q(|S'|, p/n) = \Omega(p/n)$ in the generalization bound.

Now consider the consistent subset of S consisting of $m = n/p$ points $s_{i,1}$ for $i \in [1,m]$. By Theorem 7(i), this classifier achieves a generalization bound of $O\left(\frac{m\log n}{n}\right) = O\left(\frac{\log n}{p}\right)$.

So when $p = \Omega(\sqrt{n\log n})$, this consistent classifier is better than any inconsistent classifier. Now a consistent subset of size $d \le m$ has generalization bound $O\left(\frac{d\log n}{n}\right)$. As it is NP-hard to find a subset whose size is within a factor $2^{O(\mathsf{dens}(S)\log(\text{rad}(S)/\text{marg}(S)))^{1-o(1)}}$ of the smallest consistent subset, it is NP-hard to find a consistent subset with generalization bound within a factor $2^{O(\mathsf{dens}(S)\log(\text{rad}(S)/\text{marg}(S)))^{1-o(1)}}$ of the optimal consistent subset, and the theorem follows. ■

In light of the hardness result established in Theorem 4, we specialize the goal from one of seeking a small nearly consistent sub-sample to one where the sub-sample must be a γ-net. In this case, the relevant generalization bound is provided by Theorem 10. As before, we wish to perform SRM for this bound. Fortunately, we are able to compute the latter exactly in polynomial time, and even more efficiently if we are willing to settle for a solution within a constant factor of the optimal. The proof of the following theorem follows the lines of Gottlieb et al. (2014a).

Theorem 5 *Given a sample set S equipped with a semimetric:*

(a) A nearest-neighbor classifier minimizing the generalization bound of Theorem 10 can be computed in randomized time $O(|S|^{4.373})$.

(b) A nearest-neighbor classifier whose generalization bound is within factor 2 of optimal can be computed in deterministic time $O(|S|^2 \log|S|)$.

Each of these classifiers can be evaluated on test points in time $\left(\frac{\text{rad}(S)}{\gamma}\right)^{O(\mathsf{dens}(S))}$, where γ is the margin imposed by the SRM procedure.

Proof For each of these solutions, we enumerate and sort in increasing order the distances between all oppositely labeled point pairs in S, in total time $O(|S|^2\log|S|)$. Each distance constitutes a separate guess for the optimal margin to "impose" on S. That is, for each distance γ, we will remove from S some points to ensure that all opposite labeled pairs are more than γ far apart.

To accomplish this, we iteratively build a new graph G. We initialize G with vertices representing the points of S. At each round we add to G an edge between the next closest pair of opposite labeled points, as given by the sorted enumeration above. This distance is the margin of the current round: Points connected by an edge in G represent pairs that are too close together for the current margin, and we need to compute how many points would need to be removed from G in order for no edge to remain in the graph. (However, no points or edges will actually removed from G.) As observed by Gottlieb et al. (2014a), this task is precisely the problem of bipartite vertex cover. By König's theorem, the minimum vertex cover problem in bipartite graphs is equivalent to the maximum matching problem, and a maximum matching in bipartite graphs can be computed in randomized time $O(n^{2.373})$ (Mucha and Sankowski, 2004; Williams, 2012). So for each candidate margin, we can compute in $O(n^{2.373})$ time the number of points that must be removed from the current graph G in order to remove all edges. For $O(n^2)$ possible margins, this amounts to $O(n^{4.373})$ time. Having computed for each interpoint distance the number of points required to be

deleted to achieve this distance, we choose the distance-number pair which minimizes the bound of Theorem 10. We then remove the corresponding points from S, and use the algorithm of Lemma 1 to construct a net satisfying the margin bound.

The runtime improvement in (b) comes from a faster vertex-cover computation. It is well known that a 2-approximation to vertex cover can be computed (in arbitrary graphs) by a greedy algorithm in time linear in the graph size $O(|V^+ \cup V^-| + |E|) = O(n^2)$, see e.g. Bar-Yehuda and Even (1981). This algorithm simply chooses any edge and removes both endpoints, until no edges remain. We apply this algorithm to our setting: A new edge is added to G only if both endpoints survive in the already computed cover, and then both endpoints are marked as removed in the solution to the new graph. Having computed for each interpoint distance the number of points required to be deleted to achieve this distance, we choose the distance-number pair which minimizes the bound of Theorem 10. We then remove the corresponding points from S, and use the algorithm of Lemma 1 to construct a net satisfying the margin bound. The runtime is dominated by the time required to sort the distances.

For both algorithms, a new point is classified by finding its nearest neighbor in the extracted net. ■

Remark 6 *It is worth asking whether a succesful learner in (semi)metric space must necessarily perform some type of compression. Moran and Yehudayoff (2016) recently demonstrated that for classifiers of finite VC-dimension, learning and compression are in fact equivalent. However, no such result is known — or even conjectured — for infinite VC-dimension classifiers such as 1-NN and its variants.*

If a (semi)metric learner does choose to learn by compression, he is not necessarily restricted to learning via 1-NN, but can make use of any proximity-based function. But even then, Theorems 10 and 11 below demonstrate that from an information-theoretic standpoint, roughly $(\mathrm{rad}(S)/\mathrm{marg}(S))^{\mathsf{dens}(\mathcal{X})}$ *examples are both necessary in the worst case and always sufficient. One could still hope to exhibit a learning problem in a (semi)metric space where* any *learner achieving small generalization must solve some hard problem. Results of this type are known for finite automata under cryptographic assumptions (Kearns and Vazirani, 1997, Section 1.4); could something similar be shown for proximity-based learning?*

4. Generalization guarantees

In this section, we provide general sample compression bounds, which then will be specialized to the nearest-neighbor classifier proposed above. Theorem 8 presents a smooth interpolation between two classic bounds: the consistent case with rate $\tilde{O}(1/n)$, and the agnostic case with rate $\tilde{O}(1/\sqrt{n})$. Applied to margin-based semimetric sample-compression schemes, this result yields the efficiently computable and optimizable bound in Theorem 10, which is nearly optimal (as shown in Theorem 4). Finally, the lower bound in Theorem 11 shows that even under margin assumptions, there exist adversarial distributions forcing the sample complexity to be exponential in dens.

4.1 Sample compression schemes

We use the notion of a *sample compression scheme* in the sense of Graepel et al. (2005), where it is treated in full rigor. Informally, a learning algorithm maps a sample S of size n to a hypothesis h_S. It is a d-sample compression scheme if a sub-sample of size d suffices to produce a hypothesis that agrees with the labels of all the n points. It is an ε-*lossy* d-sample compression scheme if a sub-sample of size d suffices to produce a hypothesis that disagrees with the labels of at most εn of the n sample points. At this stage, we are not assuming anything about the computational efficiency of a compression scheme.

The algorithm need not know d and ε in advance. We say that the sample S is (d, ε)-*compressible* if the algorithm succeeds in finding an ε-lossy d-sample compression scheme for this particular sample. In this case:

Theorem 7 (Graepel et al. (2005)) *For any distribution over $\mathcal{X} \times \{-1, 1\}$, any $n \in \mathbb{N}$ and any $0 < \delta < 1$, with probability at least $1 - \delta$ over the random sample S of size n, the following holds:*

(i) If S is $(d, 0)$-compressible, then $\mathrm{err}(h_S) \leq \dfrac{1}{n-d}\left((d+1)\log n + \log\dfrac{1}{\delta}\right)$.

(ii) If S is (d, ε)-compressible, then $\mathrm{err}(h_S) \leq \dfrac{\varepsilon n}{n-d} + \sqrt{\dfrac{(d+1)\log n + \log\frac{1}{\delta}}{2(n-d)}}$.

The generalizing power of sample compression was independently discovered by Littlestone and Warmuth (1986); Devroye et al. (1996), and later elaborated upon by Graepel et al. (2005). The bounds above are already quite usable, but they feature an abrupt transition from the $(\log n)/n$ decay in the lossless ($\varepsilon = 0$) regime to the $\sqrt{(\log n)/n}$ decay in the lossy regime. We now provide a smooth interpolation between the two (such results are known in the literature as "fast rates" (Boucheron et al., 2005); see also a related result in Shalev-Shwartz and Ben-David (2014)):

Theorem 8 *Fix a distribution over $\mathcal{X} \times \{-1, 1\}$, an $n \in \mathbb{N}$ and $0 < \delta < 1$. With probability at least $1 - \delta$ over the random sample S of size n, the following holds for all $0 \leq \varepsilon \leq \frac{1}{2}$: If S is (d, ε)-compressible, then, putting $\tilde{\varepsilon} = \varepsilon n/(n-d)$, we have*

$$\begin{aligned} \mathrm{err}(h_S) &\leq \tilde{\varepsilon} + \frac{2}{3(n-d)}\log\frac{n^{d+1}}{\delta} + \sqrt{\frac{9\tilde{\varepsilon}(1-\tilde{\varepsilon})}{2(n-d)}\log\frac{n^{d+1}}{\delta}} \\ &=: Q(d, \varepsilon). \end{aligned} \tag{4}$$

Proof We closely follow the argument in Graepel et al. (2005, Theorem 2), with the twist that instead of Hoeffding's inequality, we use Bernstein's. The particular form of the latter is due to Dasgupta and Hsu (2008, Lemma 1): if $\hat{p} \sim \mathrm{Bin}(n, p)/n$ and $\delta > 0$, then the following holds with probability at least $1 - \delta$:

$$p \leq \hat{p} + \frac{2}{3n}\log\frac{1}{\delta} + \sqrt{\frac{9\hat{p}(1-\hat{p})}{2n}\log\frac{1}{\delta}}. \tag{5}$$

Now suppose that S is $(d, k/n)$-compressible, as witnessed by some sub-sample $\tilde{S} \subset S$ of size d. In particular, the hypothesis $h_{\tilde{S}}$ induced by the sub-sample $\tilde{S}$ makes k or fewer mistakes on the $n-d$ points in $S \setminus \tilde{S}$. Substituting $p = \mathrm{err}(h_{\tilde{S}})$ and

$$\hat{p} = \widehat{\mathrm{err}}_{S \setminus \tilde{S}}(h_{\tilde{S}}) := \frac{1}{|S \setminus \tilde{S}|} \sum_{x \in S \setminus \tilde{S}} \mathbf{1}_{\{h_{\tilde{S}} \text{ errs on } x\}} \leq \frac{k}{n-d} = \tilde{\varepsilon}$$

into (5) yields that for fixed $\tilde{S}$ and random $S \setminus \tilde{S}$, with probability at least $1-\delta$,

$$\mathrm{err}(h_{\tilde{S}}) \quad \leq \quad \widehat{\mathrm{err}}_{S \setminus \tilde{S}}(h_{\tilde{S}}) + \frac{2}{3(n-d)} \log \frac{1}{\delta} + \sqrt{\frac{9\tilde{\varepsilon}(1-\tilde{\varepsilon})}{2(n-d)} \log \frac{1}{\delta}}, \tag{6}$$

where we used the monotonicity of $t \mapsto t(1-t)$ on $[0, \frac{1}{2}]$. To see that (6) follows from (5), note that when $\tilde{S}$ of size d is fixed and $S \setminus \tilde{S}$ is drawn iid $\sim \mathbb{P}$, we have $(n-d)\widehat{\mathrm{err}}_{S \setminus \tilde{S}}(h_{\tilde{S}}) \sim \mathrm{Bin}(n-d, \mathrm{err}(h_{\tilde{S}}))$. To make (6) hold simultaneously for all $\tilde{S} \subseteq S$, divide δ by n^d — the number of ways to choose a (multi)set $\tilde{S}$ of size d. To make the claim hold for all $d \in [n]$ and all $0 \leq \varepsilon < 1$, stratify (as in Graepel et al. (2005, Lemma 1)) over the n possible choices of d, which amounts to dividing δ by an additional factor of n. Unlike in Graepel et al. (2005), there is no need to stratify over the possible values of ε, since (5) holds for random (and not just fixed) $\hat{p}$. ■

4.2 Margin-based nearest neighbor compression

We now specialize the general sample compression result of Theorem 8 to our setting, where $h_{S'}$ induced by a sub-sample $S' \subset S$ is given by the 1-NN classifier defined in Section 2. Any sample S of size n is trivially $(n, 0)$-compressible and $(0, \frac{1}{2})$-compressible — the former is achieved by not compressing at all, and the latter by a constant predictor. Now d and ε cannot simultaneously be made arbitrarily small, and for non-degenerate samples S, the bound Q in Theorem 8 will have a nontrivial minimal value Q^*. Theorem 4 shows that computing Q^* is intractable and the algorithm in Theorem 5 solves a tractable modification of this problem. For $k \in \mathbb{N}$ and $\gamma > 0$, let us say that the sample S is (k, γ)-*separable* if it admits a sub-sample $S' \subset S$ such that $|S \setminus S'| \leq k$ and $\mathrm{marg}(S') > \gamma$, and observe that separability implies compressibility:

Lemma 9 *If S is (k, γ)-separable then it is $\left(\lceil \mathrm{rad}(S)/\gamma \rceil^{\mathsf{dens}(S)}, k/|S|\right)$-compressible.*

Proof Suppose $S' \subset S$ is a witness of (k, γ)-separability. Being pessimistic, we will allow our lossy sample compression scheme to mislabel all of $S \setminus S'$, but not any of S', giving it a sample error $\varepsilon \leq k/|S|$. Now by construction, S' is $(0, \gamma)$-separable, and thus a γ-net $\tilde{S} \subset S'$ suffices to recover the correct labels of S' via 1-nearest neighbor. Lemma 1 provides the estimate

$$|\tilde{S}| \leq \mu(S)^{\mathrm{Log}(\mathrm{rad}(S)/\gamma)} = \lceil \mathrm{rad}(S)/\gamma \rceil^{\mathsf{dens}(S)},$$

whence the compression bound. ■

These observations culminate in an efficiently optimizable margin-based generalization bound:

Theorem 10 *Fix a distribution over $\mathcal{X}$, an $n \in \mathbb{N}$ and $0 < \delta < 1$. With probability at least $1-\delta$ over the random sample S of size n, the following holds for all $0 \leq k \leq n/2$: If S is (k, γ)-separable with witness S', then*

$$\mathrm{err}(h_{S'}) \leq Q(d, k/n) =: R(k, \gamma),$$

where Q is defined in (4) and

$$d = \mu(S')^{\mathrm{Log}(\mathrm{rad}(S')/\gamma)} = \lceil \mathrm{rad}(S')/\gamma \rceil^{\mathsf{dens}(S)}.$$

Furthermore, the minimizer (k^*, γ^*) of $R(\cdot, \cdot)$ is efficiently computable.

4.3 Sample complexity lower bound

The following result shows that even under margin assumptions, a sample of size exponential in dens will be required for some distributions. (See also the recent agnostic lower bound for passive 1-NN, Kontorovich et al. (2016, Theorem 5), based on (Berend and Kontorovich, 2015, Equation (25)) an exact PAC lower bound (Kontorovich and Pinelis, 2016, Theorem 2.1).)

Theorem 11 *There are universal constants $c, \delta > 0$ such that for every semimetric space $(\mathcal{X}, \rho)$ with $\mathsf{dens}(\mathcal{X}) > 6$ and any learning algorithm mapping samples S of size n to hypotheses $h_n : \mathcal{X} \to \{-1, 1\}$, there is a distribution $\mathbb{P}$ over $\mathcal{X}$ and a target concept $f : \mathcal{X} \to \{-1, 1\}$, such that $\mathrm{err}(f) = 0$ yet*

$$\mathbb{P}\left(\mathrm{err}(h_n) \geq \frac{c \lceil \mathrm{rad}(S)/\mathrm{marg}(S) \rceil^{\mathsf{dens}(\mathcal{X})}}{n} \right) \geq \delta.$$

Proof The definition of the density constant implies the existence of an $r > 0$ and an $M \subseteq \mathcal{X}$ satisfying

(a) $m := |M| = \mu(\mathcal{X}) = 2^{\mathsf{dens}(\mathcal{X})}$

(b) M is contained in a ball of radius $2r$

(c) $\rho(x, x') \geq r$ for $x \neq x' \in M$

The assumption $\mathsf{dens}(\mathcal{X}) > 6$ implies $m \geq 65$. Taking $0 < \varepsilon < \frac{1}{8}$, recall the standard VC lower bound argument (Blumer et al., 1989; A. Ehrenfeucht, D. Haussler, M. Kearns, L. Valiant, 1989), which constructs a distribution $\mathcal{D}$ over M by putting a mass of $1 - 8\varepsilon$ on $x_1 \in M$ and distributing the remaining mass uniformly over the other $m-1$ points. Suppose the target $f : M \to \{-1, 1\}$ is drawn uniformly at random from $F = \{-1, 1\}^M$, and some learning algorithm is given a sample $S = (X_1, \ldots, X_n) \sim \mathcal{D}^n$, where $n = (m-1)/(32\varepsilon)$.

First, we argue that with probability $\Omega(1)$, S will contain exactly 2 distinct points. Let E_1 be the event that S consists of 1 point; then

$$\mathbb{P}(E_1) = (1-8\varepsilon)^n + (m-1)\left(\frac{8\varepsilon}{m-1}\right)^n. \tag{7}$$

Since $n = (m-1)/(32\varepsilon)$ and $m \geq 65$, the first term in the right-hand side of (7) is at most $e^{-(m-1)/32} \leq e^{-2} < 0.1353354$. The second term therein is (i) monotonically decreasing in m (for $m \geq 65$) and (ii) monotonically increasing in ε; at $m = 65$ and $\varepsilon = 1/8$, we have the upper bound $65/64^{16} < 10^{-27}$. Thus $\mathbb{P}(E_1) < 0.1353355$.

Let E_2 be the event that S has more than 2 points. We decompose E_2 into two cases: E_2', where $x_1 \in S$ and E_2'', where $x_1 \notin S$. Clearly, $\mathbb{P}(E_2'') \leq (8\varepsilon/(m-1))^3 < 0.000004$. To analyze E_2', define $\xi_i = \mathbf{1}_{\{X_i \neq x_1\}}$ and note that under E_2', the random quantity $1 + Z := \sum_{i=1}^m \xi_i$ upper-bounds the number of distinct points in S. Hence, $\mathbb{P}(E_2') \leq \mathbb{P}(Z \geq 2)$. Now $\mathbb{E}[Z] = n(8\varepsilon/(m-1)) = 1/4$, and by the multiplicative Chernoff's bound,

$$\mathbb{P}(Z \geq 2) = \mathbb{P}(Z \geq 4\mathbb{E}Z) \leq e^{-\mathbb{E}Z} = e^{-1/4}.$$

We conclude that $\mathbb{P}(E_2) < 0.0003 + e^{-1/4} < 0.7789$ and hence $\mathbb{P}(\mathbb{E}_1 \cup E_2) < 0.91415$.

Let E_3 be the event that S will consist exacly of 2 distinct points x, x', and furthermore, that the randomly drawn f has $f(x) \neq f(x')$; then $\mathbb{P}(E_3) \geq \frac{1}{2}(1 - \mathbb{P}(\mathbb{E}_1 \cup E_2)) > 0.04$. Conditioned on E_3, the labels assigned by f to the remaining $m-2$ points in M are iid Bernoulli(1/2) and their total mass under $\mathcal{D}$ is at least $(m-2)(8\varepsilon/(m-1)) > 7.875\varepsilon$, since $m \geq 65$. Since the labels are random, no learning algorithm can do better than flipping a coin, and the algorithm will be correct on $\frac{1}{2}(m-2)$ of the unseen points, in expectation. The probability of being correct on more than $\frac{3}{4}(m-2)$ of the points is bounded by $e^{-(m-2)/8} < 0.000381$. Thus, with probability at least $\delta := 0.039$, any learning algorithm that observes n labeled examples sampled from $\mathcal{D}$ will achieve an error of at least $(7.875/4)\varepsilon > 1.95\varepsilon$.

The above argument proves a sample complexity lower bound of $\Omega(m/\varepsilon)$. Inverting the bound for $\varepsilon = \mathrm{err}(h_n)$ implies that with probability at least δ, the following events occur:

(i) $\mathrm{err}(h_n) > 1.95 \cdot \frac{m-1}{32n} > \frac{0.05m}{n}$

(ii) the sample S contains exactly two distinct points, x, x'

(iii) the randomly chosen f has $f(x) \neq f(x')$.

To finish the proof, it remains to establish the requisite relationship between the deterministic quantity $m = |M|$ and the random quantity

$$\mu(\mathcal{X})^{\mathrm{Log}(\mathrm{rad}(S)/\mathrm{marg}(S))} = \lceil \mathrm{rad}(S)/\mathrm{marg}(S) \rceil^{\mathsf{dens}(\mathcal{X})},$$

where $S \sim \mathcal{D}^n$. By construction, $\mathrm{rad}(M) \leq 2r$, and by (ii,iii), $r \leq \mathrm{marg}(S) \leq 2r$, whence $\mathrm{Log}\,(\mathrm{rad}(S)/\mathrm{marg}(S)) = 1$ and

$$m = \mu(\mathcal{X})^{\mathrm{Log}(\mathrm{rad}(S)/\mathrm{marg}(S))}.$$

and the claimed lower bound follows. ■

5. Density dimension of some common semimetrics

In this section we demonstrate the utility of the density dimension by calculating its value under some common semimetric distance functions on d-dimensional vectors. The first of these functions is the Jensen-Shannon divergence, equivalent (Fuglede and Topsøe, 2004) to the ℓ_2-squared distance function $\ell_2^2(x,y) = \sum_{i=1}^d |x_i - y_i|^2$. We also consider the non-metric ℓ_p-spaces for $0 < p < 1$, $\ell_p(x,y) = (\sum_{i=1}^d |x_i - y_i|^p)^{1/p}$. Finally, we consider the k-median Hausdorff distance.

Recall that the usual Hausdorff distance is a metric defined on any two point sets A and B, and we shall make the simplifying assumption that $|A| = |B| = m$. Let $l(a,b)$ for all $a \in A$ and $b \in B$ be a vector distance function — for simplicity we shall assume the Euclidean ℓ_2 distance — and $l(a,B)$ be the distance from $a \in A$ to its nearest neighbor in B. The Hausdorff distance is the maximum distance between a point in A (or B) and its nearest neighbor in B (respectively, in A): $\max\{\max_{a\in A} l(a,B), \max_{b\in B} l(b,A)\}$. Huttenlocher et al. (1993) define the k-median Hausdorff distance (in the terminology of Jacobs et al. (2000), but perhaps more aptly termed the k-rank Hausdorff distance) by setting $h_k(A,B)$ to be the k-th smallest value in the vector $v = (l(a_1,B), \ldots, l(a_d,B))$, and then the k-rank Hausdorff distance is $H_k(A,B) = \max\{h_k(A,B), h_k(B,A)\}$. Note that $H_m(A,B)$ recovers the classic metric Hausdorff distance (and we require $k \leq m$). On the other hand, we can show that $H_1(A,B)$ is sufficiently robust to be *universal* for all semimetrics — that is, any semimetric can be realized by the distance function $H_1(A,B)$:

Lemma 12 *If ρ is a semimetric on a point set X of size n, then ρ can be realized as the H_1 distance (induced by $l = \ell_2$ as above) over subsets of $\mathbb{R}$ of size n.*

Proof Put $D = \mathrm{diam}(X)$ and replace each point $x_i \in X$ with a set $A_i \subset \mathbb{R}$ of size n as follows. For $a_{i,j} \in A_i$, if $j \geq i$, then set $a_{i,j} = 2D((i+1)n + j)$, and otherwise set $a_{i,j} = a_{j,i} + \rho(x_i, x_j)$.

Consider any pair $x_i, x_j \in X$ for $i < j$. Clearly $\ell_2(a_{i,j} - a_{j,i}) = \ell_2(a_{i,j} - a_{i,j} - \rho(x_i,x_j)) = \rho(x_i,x_j)$, so $H_1(A_i,A_j)$ is at most this value. On the other hand, for any k,p we can show that $\ell_2(a_{i,k} - a_{j,p}) \geq D$ whenever $k \neq j$ or $p \neq i$): If $i \leq k$, we have $a_{i,k} = 2D((i+1)n+k)$, and otherwise $2D((k+1)n+i) \leq a_{i,k} \leq 2D((k+1)n+i) + D$. Similarly, if $j \leq p$ we have $a_{j,p} = 2D((j+1)n+p)$, and otherwise $2D((p+1)n+j) \leq a_{j,p} \leq 2D((p+1)n+j) + D$. Since by assumption $i \neq j$, the two terms differ by at least D unless both $j = k$ and $i = p$. ■

We bound the density dimension under these three distance functions.

Theorem 13 *A set of m d-dimensional vectors has density dimension: $O(d)$ under ℓ_2-squared, $O(d/p^2)$ under ℓ_p $(0 < p < 1)$, and $O(k(d + \log m))$ under H_k.*

Proof We begin with a standard proof that a set of d-dimension Euclidean vectors has density dimension $O(d)$. Take any radius 4 ball, and we bound the size of a 2-net of points within this ball. By the triangle inequality, 1-radius balls centered at the 2-net points do not intersect, and so the density constant of the space is bounded by the number of 1-radius balls whose centers can be packed into the 4-radius ball. Since a piece of a 1-radius ball may escape the 4-radius ball, by the triangle inequality this term is bounded by the number of

1-radius balls that can be packed into a 5-radius ball. The ratio of the volume of a 5-radius ball to that of a 1-radius ball is 5^d, which bounds the density constant of d-dimensional Euclidean space.

For ℓ_2^2, we embed this space into ℓ_2 by simply retaining the vectors and changing only the distance function. In other words, we take the square root of all the distances, which is known as a *snowflake* operator. To bound the number of 2-net points within a ball of radius 4 in ℓ_2^2, consider instead a larger 1-net in the 4-radius ball. After the embedding, it is a 2-radius Euclidean ball containing a 1-net, and so the density constant of ℓ_2^2 is $2^{O(d)}$ as well.

For ℓ_p $(0 < p < 1)$, let us consider a snowflake of this function, that is $\ell_p^{p/2}(x, y) = (\sum_{i=1}^d |x_i - y_i|^p)^{1/2}$. We can show that the vectors under this distance function can be embedded into $O(d/p^2)$-dimensional Euclidean space with only constant distortion: Considering each coordinate separately, the distance operator $|x_i - y_i|^{p/2}$ on a single coordinate has the effect of embedding all points on a line into a helix. It is known that this embedding can be realized in $O(1/p^2)$-dimensional Euclidean space with arbitrarily small distortion (see Talagrand (1992) for $\frac{1}{2} < p < 1$, and Gottlieb and Krauthgamer (2011); Bartal et al. (2011) for $0 < p \leq \frac{1}{2}$). We create such an embedding for each coordinate and then concatenate the coordinate embeddings into a single vector. This yields an embedding from d-dimensional $\ell_p^{p/2}$ into $O(d/p^2)$-dimensional Euclidean space with arbitrarily small distortion. Then a 1-net inside a $2^{2/p}$-radius ball in the original ℓ_p-space is a 1-net inside a $(2+\varepsilon)$-radius ball in the target Euclidean space (for arbitrarily small ε), and so its density dimension is $O(d/p^2)$.

For the k-rank Hausdorff distance, first note that since h_k is non-decreasing under deletions, we have for all vector sets A, B and subsets $A' \subset A$ and $B' \subset B$ that $H_k(A', B') \geq H_k(A, B)$. Also, we can show there always exist A', B' each of size exactly k that satisfy $H_k(A', B') = H_k(A, B)$: To see this, assume without loss of generality that $h_k(A, B) \geq h_k(B, A)$. Let $h_k(A, B)$ be determined by a set $A_1 \subset A$ of size exactly k, along with a set $B_1 \subset B$ of size $k_1 \leq k$, where the points of B_1 are the nearest neighbors of A_1. Similarly, let $h_k(B, A)$ be determined by set $B_2 \subset B$ of size exactly k, along with nearest-neighbor set $A_2 \subset A$ of size $k_2 \leq k$. Now add to B' the k_1 vectors of B_1, and for each $b \in B_1$, add to A' a distinct vector $a \in A_1$ for which b is its nearest neighbor. Further add to A' $\min\{k_2, k - k_1\}$ vectors of A_2, and add to B' $\min\{k_2, k - k_1\}$ distinct vectors of B_2 that are the nearest neighbors of the relevant vectors in A_2. If the resulting sets A', B' are of size k then we are done, and have $H_k(A', B') = H_k(A, B)$. Otherwise, we add to A' and B' additional points of A_1 and B_2, respectively, until both sets are of size k – as A', B' already contain the respective nearest neighbors of B_2, A_1, the additional points do not affect the distance.

Now consider a set $\mathcal{A}$ of vector sets all within distance 2 of some center set $A_c \in \mathcal{A}$ and at mutual inter-set distance at least 1. We will show that $|\mathcal{A}| = 2^{O(kd)}m^k$, from which the item follows. To prove this, take in turn each subset $A_c' \subset A_c$ of size k (there are $\binom{m}{k} < m^k$ such subsets), and let $\mathcal{A}'$ contain all sets $A_i' \subset A_i$ of size k for which $H_k(A_c', A_i') = H_k(A_c, A_i)$. $\mathcal{A}'$ has radius 2 and inter-set distance at least 1. To complete the proof, we will show that $n = |\mathcal{A}'| \leq 2^{O(kd)}$, from which it follows that $|\mathcal{A}| < |\mathcal{A}'|m^k = 2^{O(kd)}m^k$:

Since $H_k(A_c', A_i') \leq 2$ for all $A_i' \in \mathcal{A}'$, we have that $h_k(A_i', A_c') \leq 2$, and so every vector of A_i' is within Euclidean distance 2 of one of the k vectors of A_c'. Let each vector of A_c' be the center of a 2-radius Euclidean ball. Clearly, the vectors of each A_i' fall into at most

k different Euclidean balls, and so there are at most 2^k different sets of k balls (i.e., ball configurations) into which A_i' may fall. Let $C \subset \mathcal{A}'$ $(p = |C|)$ include all vertex sets falling into some specific ball configuration. For each pair of vertex sets in C, there must be a pair of vectors at distance at least 2 falling into the same 2-radius Euclidean ball. Then $p = 2^{O(kd)}$ and $n \leq 2^k p = 2^{O(kd)}$. ■

We leave it as an open problem to improve on the dependence of k in our bound of the density dimension of the Hausdorff distance.

We conclude this section with an illustration of how the theory developed in this paper explains the success of the greedy net-based compression algorithm, even in the case of semimetrics. We present results for the Hausdorff semimetric applied to the Covertype dataset, found in the UCI Machine Learning Repository.[2] This dataset contains 7 different label types, which we treated as 21 separate binary classification problems; we report representative results below.

data set	**original size**	**% compressed down to**
Covertype 2 vs. 5	2000	97
Covertype 1 vs. 4	2000	25
Covertype 4 vs. 7	2000	2

Figure 2: Summary of the performance of semimetric sample compression algorithm.

Acknowledgments

Thanks to Daniel Hsu for bringing Equation (5) to our attention and Shay Moran for pointing out the analogous result of Shalev-Shwartz and Ben-David (2014). A special thanks to Yevgeni Korsunsky for carefully reading the manuscript and providing very helpful comments.

References

A. Ehrenfeucht, D. Haussler, M. Kearns, L. Valiant. A general lower bound on the number of examples needed for learning. *Inf. Comp.*, 82(3):247 – 261, 1989. ISSN 0890-5401. doi: http://dx.doi.org/10.1016/0890-5401(89)90002-3. URL `http://www.sciencedirect.com/science/article/pii/0890540189900023`.

Noga Alon, Shai Ben-David, Nicolò Cesa-Bianchi, and David Haussler. Scale-sensitive dimensions, uniform convergence, and learnability. *Journal of the ACM*, 44(4):615–631, 1997. URL `citeseer.ist.psu.edu/alon97scalesensitive.html`.

Alexandr Andoni and Robert Krauthgamer. The computational hardness of estimating edit distance. *SIAM J. Comput.*, 39(6):2398–2429, April 2010. doi: 10.1137/080716530.

2. `http://tinyurl.com/cover-data`

Maria-Florina Balcan, Avrim Blum, and Nathan Srebro. Improved guarantees for learning via similarity functions. In *21st Annual Conference on Learning Theory - COLT 2008, Helsinki, Finland, July 9-12, 2008*, pages 287–298, 2008a. URL http://colt2008.cs.helsinki.fi/papers/86-Balcan.pdf.

Maria-Florina Balcan, Avrim Blum, and Nathan Srebro. A theory of learning with similarity functions. *Machine Learning*, 72(1-2):89–112, 2008b. doi: 10.1007/s10994-008-5059-5. URL http://dx.doi.org/10.1007/s10994-008-5059-5.

Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. A discriminative framework for clustering via similarity functions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 671–680, 2008c. doi: 10.1145/1374376.1374474. URL http://doi.acm.org/10.1145/1374376.1374474.

Reuven Bar-Yehuda and Shimon Even. A linear-time approximation algorithm for the weighted vertex cover problem. *J. Algorithms*, 2(2):198–203, 1981. doi: 10.1016/0196-6774(81)90020-1. URL http://dx.doi.org/10.1016/0196-6774(81)90020-1.

Yair Bartal, Ben Recht, and Leonard J. Schulman. Dimensionality reduction: Beyond the johnson-lindenstrauss bound. In *SODA*, 2011.

R. Basri, L. Costa, D. Geiger, and D. Jacobs. Determining the similarity of deformable shapes. In *Physics-Based Modeling in Computer Vision*, 1995. doi: 10.1109/PBMCV.1995.514678.

Daniel Berend and Aryeh Kontorovich. A finite sample analysis of the naive bayes classifier. *Journal of Machine Learning Research*, 16:1519–1545, 2015. URL http://dl.acm.org/citation.cfm?id=2886797.

Alina Beygelzimer, Sham Kakade, and John Langford. Cover trees for nearest neighbor. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 97–104, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2. doi: http://doi.acm.org/10.1145/1143844.1143857.

Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *J. Assoc. Comput. Mach.*, 36(4):929–965, 1989. ISSN 0004-5411.

Stéphane Boucheron, Olivier Bousquet, and Gábor Lugosi. Theory of classification: a survey of some recent advances. *ESAIM: Probability and Statistics*, 9:323–375, 2005. ISSN 1262-3318. doi: 10.1051/ps:2005018. URL http://dx.doi.org/10.1051/ps:2005018.

Dennis K. Burke. Cauchy sequences in semimetric spaces. *Proc. AMS*, 33(1):161–164, 1972. ISSN 00029939. URL http://www.jstor.org/stable/2038192.

I. Cox, M. Miller, S.M. Omohundro, and P.N. Yianilos. Pichunter: Bayesian relevance feedback for image retrieval. In *Intl. Conf. Pat. Rec.*, 1996. doi: 10.1109/ICPR.1996.546971.

Sanjoy Dasgupta and Daniel Hsu. Hierarchical sampling for active learning. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, pages 208–215, 2008. doi: 10.1145/1390156.1390183. URL http://doi.acm.org/10.1145/1390156.1390183.

Luc Devroye, László Györfi, and Gábor Lugosi. *A probabilistic theory of pattern recognition*, volume 31 of *Applications of Mathematics (New York)*. Springer-Verlag, New York, 1996. ISBN 0-387-94618-7.

Bend Fuglede and Flemming Topsøe. Jensen-shannon divergence and hilbert space embedding. In *International Symposium on Information Theory (ISIT)*, June 2004. doi: 10.1109/ISIT.2004.1365067.

Y. Gdalyahu and D. Weinshall. Flexible syntactic matching of curves and its application to automatic hierarchical classification of silhouettes. *IEEE Trans. PAMI*, 21(12):1312–1328, 1999. ISSN 0162-8828. doi: 10.1109/34.817410.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014. URL http://papers.nips.cc/paper/5423-generative-adversarial-nets.

Lee-Ad Gottlieb and Robert Krauthgamer. A nonlinear approach to dimension reduction. In *SODA*, 2011.

Lee-Ad Gottlieb and Robert Krauthgamer. Proximity algorithms for nearly doubling spaces. *SIAM J. Discrete Math.*, 27(4):1759–1769, 2013.

Lee-Ad Gottlieb, Aryeh Kontorovich, and Robert Krauthgamer. Efficient classification for metric data. *IEEE Transactions on Information Theory*, 60(9):5750–5759, 2014a. doi: 10.1109/TIT.2014.2339840. URL http://dx.doi.org/10.1109/TIT.2014.2339840.

Lee-Ad Gottlieb, Aryeh Kontorovich, and Pinhas Nisnevitch. Near-optimal sample compression for nearest neighbors. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 370–378, 2014b. URL http://papers.nips.cc/paper/5528-near-optimal-sample-compression-for-nearest-neighbors.

Lee-Ad Gottlieb, Aryeh Kontorovich, and Pinhas Nisnevitch. Nearly optimal classification for semimetrics. In *Artificial Intelligence and Statistics (AISTATS)*, 2016.

Thore Graepel, Ralf Herbrich, and John Shawe-Taylor. PAC-bayesian compression bounds on the prediction error of learning algorithms for classification. *Machine Learning*, 59 (1-2):55–76, 2005.

D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge. Comparing images using the hausdorff distance. *IEEE Trans. PAMI*, 15(9):850–863, 1993. ISSN 0162-8828. doi: 10.1109/34.232073.

J. Puzicha, J. Buhmann, Y. Rubner, C. Tomasi. Empirical evaluation of dissimilarity measures for color and texture. In *ICCV*, 1999. doi: 10.1109/ICCV.1999.790412.

David W. Jacobs, Daphna Weinshall, and Yoram Gdalyahu. Condensing image databases when retrieval is based on non-metric distances. In *ICCV*, pages 596–601, 1998.

David W. Jacobs, Daphna Weinshall, and Yoram Gdalyahu. Classification with non-metric distances: Image retrieval and class representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(6):583–600, 2000. doi: 10.1109/34.862197. URL `http://doi.ieeecomputersociety.org/10.1109/34.862197`.

A. Jain and D. Zongker. Representation and recognition of handwritten digits using deformable templates. *IEEE Trans. PAMI*, 19(12):1386–1391, 1997. ISSN 0162-8828. doi: 10.1109/34.643899. URL `http://dx.doi.org/10.1109/34.643899`.

Micheal Kearns and Umesh Vazirani. *An Introduction to Computational Learning Theory*. The MIT Press, 1997.

Aryeh Kontorovich and Iosif Pinelis. Exact lower bounds for the agnostic probably-approximately-correct (PAC) machine learning model. *CoRR*, abs/1606.08920, 2016. URL `http://arxiv.org/abs/1606.08920`.

Aryeh Kontorovich and Roi Weiss. Maximum margin multiclass nearest neighbors. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 892–900, 2014. URL `http://jmlr.org/proceedings/papers/v32/kontorovichb14.html`.

Aryeh Kontorovich, Sivan Sabato, and Ruth Urner. Active nearest-neighbor learning in metric spaces, arxiv:1605.06792. *CoRR*, abs/1605.06792, 2016. URL `http://arxiv.org/abs/1605.06792`.

Robert Krauthgamer and James R. Lee. Navigating nets: Simple algorithms for proximity search. In *15th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 791–801, January 2004.

Nick Littlestone and Manfred K. Warmuth. Relating data compression and learnability, unpublished. 1986.

M. Dubuisson, A. Jain. A modified hausdorff distance for object matching. In *Intl. Conf. Pat. Rec.*, 1994. doi: 10.1109/ICPR.1994.576361.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations Of Machine Learning*. The MIT Press, 2012.

Shay Moran and Amir Yehudayoff. Sample compression schemes for VC classes. *J. ACM*, 63(3):21:1–21:10, 2016. doi: 10.1145/2890490. URL `http://doi.acm.org/10.1145/2890490`.

Marcin Mucha and Piotr Sankowski. Maximum matchings via gaussian elimination. In *FOCS '04: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 248–255, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7695-2228-9. doi: http://dx.doi.org/10.1109/FOCS.2004.40.

Assaf Naor and Gideon Schechtman. Planar earthmover is not in l_1. *SIAM J. Comput.*, 37: 804–826, June 2007. doi: 10.1137/05064206X.

Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

Michel Talagrand. Approximating a helix in finitely many dimensions. *Annales de l'institut Henri Poincaré (B) Probabilités et Statistiques*, 28(3):355–363, 1992.

A. L. Vol'berg and S. V. Konyagin. On measures with the doubling condition. *Izv. Akad. Nauk SSSR Ser. Mat.*, 51(3):666–675, 1987. ISSN 0373-2436.

Ulrike von Luxburg and Olivier Bousquet. Distance-based classification with Lipschitz functions. *Journal of Machine Learning Research*, 5:669–695, 2004.

L. Wang, C. Yang, and J. Feng. On learning with dissimilarity functions. In *ICML*, 2007. doi: 10.1145/1273496.1273621. URL `http://doi.acm.org/10.1145/1273496.1273621`.

Daphna Weinshall, David W. Jacobs, and Yoram Gdalyahu. Classification in non-metric spaces. In *Advances in Neural Information Processing Systems 11, [NIPS Conference, Denver, Colorado, USA, November 30 - December 5, 1998]*, pages 838–846, 1998. URL `http://papers.nips.cc/paper/1581-classification-in-non-metric-spaces`.

Virginia Vassilevska Williams. Breaking the Coppersmith-Winograd barrier. In *STOC '12: Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, New York, NY, USA, 2012. ACM Press.

Wallace Alvin Wilson. On semi-metric spaces. *American Journal of Mathematics*, 53(2): 361–373, 1931.

Appendix A. Inapplicability of metric techniques to semimetrics

Our proof that the density constant controls the size of a net in semimetric spaces (Lemma 1) is non-constructive, and as a result the construction times for nets in semimetrics are necessarily inferior to those of metric spaces. Likewise, estimating a margin of a point set is more challenging for semimetrics and requires a more sophisticated approach than for metric spaces (see Remark 3).

Unlike in metric spaces, where the covering numbers $\mathcal{N}(\cdot)$ and the packing numbers $\mathcal{M}(\cdot)$ are related via $\mathcal{M}(2\varepsilon) \leq \mathcal{N}(\varepsilon) \leq \mathcal{M}(\varepsilon)$ (see e.g., Alon et al. (1997)), violating the triangle inequality breaks this connection between covering and packing. Particularly, for

semimetrics, a doubling constant (while well-defined) does not imply a packing property such as that of Lemma 1. As a consequence, the bounds in the host of results constituting the theory of learning in doubling metric spaces are not applicable to semimetrics. This fact is captured in the following lemma:

Lemma 14 *In semimetric spaces, a bound on the doubling constant does not imply a packing number. In particular, There exist semimetric spaces $\mathcal{X}$ of arbitrary cardinality with a universally bounded doubling constant $\lambda(\mathcal{X}) = O(1)$, such that $\mathcal{X}$ contains a* $\mathrm{rad}(\mathcal{X})$*-net C of size $\Theta(|\mathcal{X}|)$.*

Proof Let $\mathcal{X}$ be composed of two sets, A and A'. Put $A = \{a_1, \ldots, a_n\}$, endowed with the line metric $\rho(a_i, a_j) = |i-j|$, so the maximum distance in A is $n-1$. Note that $\lambda(A) = O(1)$. Define A' to consist of n points, such that

$$\rho(a'_i, a_j) = \rho(a_i, a_j) + \phi \mathbf{1}_{\{i=j\}}, \qquad (\phi > 0 \text{ infinitesimal}),$$

while $\rho(a'_i, a'_j) = n - 1$. This defines a semimetric on $\mathcal{X}$.

Clearly, A' forms a $\mathrm{rad}(\mathcal{X})$-net of size $|\mathcal{X}|/2$, and yet we can show that $\lambda(\mathcal{X}) = O(1)$. Indeed, consider any ball $B_r(x)$ in $\mathcal{X}$. Then all points in $B_r(x)$ can be covered by the same $\lambda(A) = O(1)$ balls of radius $\frac{r}{2}$ that cover $A \cap B_r(x)$. The claim follows. ■

In fact, a deeper principle underlies the results above: In metric spaces, the doubling and density constants are almost the same, while in semimetric spaces there may be a large gap between them. This is captured in the following lemma, which delineates the relationship between the doubling constant and density constant. (The first half of the lemma is due to Gottlieb and Krauthgamer (2013).)

Lemma 15 *Let $\mathcal{X}$ be point set endowed with a metric distance function. Then*

(a) $\lambda(\mathcal{X}) \leq \mu(\mathcal{X})$,

(b) $\sqrt{\mu(\mathcal{X})} \leq \lambda(\mathcal{X})$.

Let $\mathcal{Y}$ be a point set endowed with a semimetric distance function. Then

(c) $\lambda(\mathcal{Y}) \leq \mu(\mathcal{Y})$,

(d) $\mu(\mathcal{Y})$ *may be as large as* $\Theta(|\mathcal{Y}|)$*, even when* $\lambda(\mathcal{Y}) = O(1)$.

Proof To prove (a) and (c), that $\lambda \leq \mu$: Consider any open ball $B_r(x) \in \mathcal{X}$. Let C be a maximal collection of points at mutual interpoint distance at least $\frac{r}{2}$, and note that by definition $|C| \leq \mu(\mathcal{X})$. By the maximality of C, $|C|$ balls of radius $\frac{r}{2}$ centered at points of C cover all of $B_r(x)$, so $\lambda(\mathcal{X}) \leq |C| \leq \mu(\mathcal{X})$. For (b): again, consider any open ball $B_r(x) \in \mathcal{X}$, and let C be a maximal collection of points at mutual interpoint distance at least $\frac{r}{2}$. Now, by definition $\mathcal{X}$ may be covered by $\lambda(\mathcal{X})$ balls of radius $\frac{r}{2}$, and each of these smaller balls may be covered by $\lambda(\mathcal{X})$ balls of radius $\frac{r}{4}$, so there exists a set of $\lambda^2(\mathcal{X})$ balls of radius $\frac{r}{4}$ covering all of X, and in particular C. By the triangle inequality, each ball of radius $\frac{r}{4}$ can cover at most one point of C, and so $|C| \leq \lambda^2(\mathcal{X})$. Finally, (d) follows

immediately from Lemma 14. ■

In metric spaces, the following bounds on exact and approximate nearest neighbor search are well-known.

Lemma 16 *Given a point set S equipped with a metric distance function, and a query point x:*

(a) Locating the exact nearest neighbor of x in S requires $\Theta(|S|)$ comparisons in the worst case.

(b) A $(1+\varepsilon)$-approximate nearest neighbor of x in S can be found in time $2^{\mathsf{ddim}(S)} \log |S| + \varepsilon^{-O(\mathsf{ddim}(S))}$.

Proof To prove (a), let S be a set of points obeying the line metric, i.e. the distance between $s_i, s_j \in S$ is $|i-j|$. Suppose x is at distance $n = |S|$ from s_i, and at distance $n+1$ from all other points of S. Then s_i can be any point of S, and cannot be located without inspecting each point. The claim in (b) is the result of Krauthgamer and Lee (2004). ■

For semimetric spaces, we demonstrate that the situation is much worse:

Lemma 17 *Given a point set S equipped with a semimetric distance function, discovering an exact or approximate nearest neighbor requires $\Theta(|S|)$ comparisons in the worst case.*

Proof For the upper bound, trivially $O(|S|)$ time is sufficient to consider every point in S.

For the lower bound, suppose the query point q is at an infinitesimally small distance from a single point $s_0 \in S$, and at distance $\mathrm{rad}(S)$ from all other points of S. Then s_0 can be any point in S, and cannot be located without inspecting each point: Without the triangle inequality, the distance between one pair of points has no bearing on any other distance. ■

Journal of Machine Learning Research 18 (2017) 1-39 Submitted 5/16; Revised 3/17; Published 4/17

Bridging Supervised Learning and Test-Based Co-optimization

Elena Popovici elena@icosystem.com
Icosystem Corp.
222 Third Street, Suite 0142
Cambridge, MA 02142, USA

Editor: Una-May O'Reilly

Abstract

This paper takes a close look at the important commonalities and subtle differences between the well-established field of supervised learning and the much younger one of co-optimization. It explains the relationships between the problems, algorithms and views on cost and performance of the two fields, all throughout providing a two-way dictionary for the respective terminologies used to describe these concepts. The intent is to facilitate advancement of both fields through transfer and cross-pollination of ideas, techniques and results. As a proof of concept, a theoretical study is presented on the connection between existence / lack of free lunch in the two fields, showcasing a few ideas for improving computational complexity of certain supervised learning approaches.

Keywords: supervised learning, active learning, co-optimization, free lunch, optimal algorithms

1. Introduction

It is not uncommon in science that separate fields independently develop similar ideas but describe them with different language, or that they need to solve similar problems but approach them from different perspectives and thus derive different methods of solving them. Either field would benefit from learning about the latest and greatest in the other, but achieving such benefits requires overcoming a couple of barriers: lack of awareness of the existence of another field concerned with similar issues and lack of understanding of its language and perspective. The goal of this paper is to break such barriers between the fields of supervised learning and co-optimization.

Supervised learning has been around for a long time, thus knowledge and understanding of it has permeated other disciplines. Loosely speaking, it is concerned with generalizing from given examples to unseen ones. Co-optimization is a much newer area, not widely known outside its community. The term has been introduced by Service and Tauritz (2008a), though the ideas had already been floating around in the field of coevolution (Popovici et al., 2010), a sub-field of evolutionary computation (De Jong, 2006) that has been around since the early 90's (Hillis, 1990). There are multiple possible goals in co-optimization, of which a common one is finding solutions that would perform well in many different situations (tests), given the ability to do only a limited number of performance assessments. This type of goal is the focus of *test-based co-optimization*. To draw a quick parallel to supervised learning, un-assessed situations are similar in nature to unseen examples, but their space can and must be explored. The present work delves into this parallel in much greater detail.

We start in Section 2.1 with a handful of examples of problems from both fields, first presented in natural language and then formalized to highlight both the commonalities and the differences between them. Then in Section 2.2 we discuss the relationship between the typical algorithmic approaches applied to solve such problems in their respective fields. Two themes thread through both sections: how much and what kind of information is available to the problem-solver, and what are the various kinds of cost the problem-solver might incur. The performance-versus-cost issue is further detailed in Section 2.3, highlighting how both fields lack a performance-evaluation approach encompassing all cost types.

Thus Section 2 should provide a useful guide to those newcomers to one of the fields that already have a background in the other, but it is *not a detailed survey* of either field; many books and review papers are available for that purpose, e.g., by Abu-Mostafa et al. (2012) for supervised learning, Popovici et al. (2010) for co-optimization. Rather, it presents a *mapping between the core issues and concepts* that researchers in the two fields concern themselves with and the language they use in the process (summarized in Table 3). State-of-the-art approaches in either field should be describable in terms of such core notions; while not the main focus, a couple of supervised learning examples, such as deep learning (LeCun et al., 2015) and the reusable holdout (Dwork et al., 2015), are mentioned in sections 2.3.1 and 2.3.2.

The motivation behind this type of presentation is to enable researchers in the two fields to better understand each other and provide a bridge over which methods, ideas and results can travel both ways and contribute to the advancement of both fields. In particular, for machine learning researchers, the co-optimization paradigm should reveal the formalization of new and non-trivial learning tasks and of methods for assessing performance on those tasks which offer interesting free lunches. To illustrate the potential, in Section 3 we derive the counterpart for binary classification of a recent line of work in co-optimization concerning the exploitation of free lunch to design algorithms that have optimal performance with respect to a particular type of cost. We discuss the implications for the other cost types described in Section 2.3 and also point out how this new paradigm suggests alternative approaches to instance selection and active learning.

2. Parallel Between Fields

There are three main aspects of any computer science field: the problems to be solved, the algorithms used to solve them and the paradigms used to assess and compare different algorithms' performance with respect to the problem-solving goal. For each of these aspects, we explain what's common and what's different between supervised learning and co-optimization. In the process, we also build a two-way dictionary between the terminologies of these two fields, summarized in Table 3.

2.1 Problems

We first present some problem examples through their real-world formulations, but explaining or replacing domain jargon with simple words. This should make it easier to spot both commonalities and differences, which are then formalized later in Section 2.1.3.

©2017 Elena Popovici.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-223.html.

2.1.1 Examples of Co-optimization Problems

Resilient Piping – Automated design of robust physical systems (Co-optimization). Consider the problem of placing valves in a network of water-carrying pipes so as to maximize the resilience of the network in case of damage. Such piping networks carry water from pumps to a number of usage sites; this could be for human consumption, for equipment cooling or for putting out fires (Popovici et al., 2007). Should damage occur somewhere in the network, such as broken pipes causing local leaks and global pressure drop, the damaged areas need to be isolated by automatically closing nearby "smart" valves, so as to stop the leak and restore pressure to as much of the network as possible. Smart valves are expensive and can only be placed sparingly throughout the network. Where they are placed has a big impact on how tightly we can isolate a given damage, i.e., how resilient the network is to that damage. For any given placement of valves and any given damage, we can compute this resilience value using graph-traversal algorithms to identify the valves closest to damaged spots and to determine, once these valves are closed, how many usage sites still have an unobstructed path to a pump and thus can get water. The goal is to find, for a given piping network and number of smart valves, a placement that provides as good as possible resilience across many/all damage scenarios. More broadly, the problem described here is that of automating the design of physical systems that perform well under many different circumstances.

Sorting Networks – Automated algorithm design (Co-search). Consider the task of sorting in increasing order a sequence of numbers by repeatedly comparing pairs of numbers and, if the first is bigger than the second, swapping them. An algorithm performing such sorting is called a sorting network, because it can be represented as a network of compare-and-swap gates. Sorting networks have both software and hardware applications. Each gate incurs a cost, either computational or actual, so one would like to use as few of them as possible. For a given number of available gates and a given length of input sequences, the goal is to find a network with that number of gates that correctly sorts all input sequences of that length (Hillis, 1990). More broadly, the problem described here is that of automating the design of computer programs that for any possible input produce the correct output, where correctness is according to the computational task at hand (e.g., sorting).

2.1.2 Examples of Supervised Learning Problems

Photo Labeling – Automated item classification (Supervised binary classification). Consider the problem of automating the labeling of photos as to whether or not they contain a particular thing, say a balloon. We are provided with a set of examples of photos that have been manually labeled by humans as 'yes' if they contain a balloon or 'no' if they don't. The goal is to find a computer program that outputs the correct label for any possible photo.

Protein Structure – Outcome prediction (Discrete regression). Consider the problem of predicting higher-level protein structure. The primary structure of a protein is specified by a chain of amino-acids of 20 possible types. In nature, this chain folds into a convoluted shape, such that neighboring amino-acids become part of local formations of three different types: coil, helix and beta-strand. Determining this secondary-structure sequence of a given protein (i.e., what type of formation each amino-acid becomes part of) requires lab experiments and it is known only for a small percentage of all known proteins. The goal is to find a computer program that outputs the correct secondary-structure sequence for any amino-acid sequence (Cheng et al., 2008).

More broadly, these two problems are about learning from samples in order to generalize to new, as yet unseen cases.

2.1.3 Problem Comparisons and Formalisms

While these problems span a varied set of domains and at first glance may not appear particularly related, closer inspection reveals common aspects as well as key differentiators.

All of the descriptions above state that the problem-solving goal is finding something (a placement of valves, a network of compare-and-swap gates, a computer program). This 'object' of interest needs to 'perform' in a certain way (or have some desirable properties) across all/many 'contexts' (damages, number sequences, photos, proteins). While not explicitly stated in the above descriptions, for each of these problems the number of contexts of interest can be very large: there are millions of proteins with known amino-acid sequence but unknown structure; there is an infinite number of photos, or at least a combinatorial number of them if we restrict the size; there is an exponential number of sorting-relevant sequences;[1] and the number of possible damage locations in a water-delivery network can also be significant, with the number of damage scenarios increasing in combinatorial fashion if we account for co-occurrences at different locations (which can happen for instance in case of earthquakes).

Additionally, for each of these problems, the set of objects of interest can also be very large: if there are n possible locations where we could place valves, but we can only afford $m < n$ valves, then the number of possible placements is $\binom{n}{m}$; the number of possible networks with n compare-and-swap gates is also combinatorial in n; and the number of input-output mappings for the computer programs predicting photo labels or protein structures is exponential.

The descriptions above also hint at differences in the information available to help us solve the respective problem. To see this better, we turn to formalisms. Table 1 provides a summary of the notation. Throughout the rest of Section 2 we underline key terminology which is then collated in Table 3.

Co-optimization Formalisms. Consider the problem of designing resilient piping networks. Let S denote the set of all possible placements of a given number of valves in a given piping network. Let T denote the set of all possible/likely damages to that piping network. For any valve placement $s \in S$ and any damage $t \in T$ we can compute the resilience of s to t, as described earlier. Let this computation be denoted by a metric $M : S \times T \to V \subset \mathbb{R}$. The goal is to find a valve placement $s \in S$ that maximizes, say, the *average* resilience over all damages $t \in T$, $g(s) = avg_{t \in T} M(s,t)$.[2] We call this goal the maxiavg solution concept (i.e., definition of what constitutes a solution) and we call g the quality function; $g(s)$ is the quality of potential solution s. Equivalently, we could be concerned with the *total* resilience over all damages, $g(s) = \sum_{t \in T} M(s,t)$, a solution concept called maxisum. Other interpretations of 'as good as possible resilience across many/all

1. Specifically, 2^n, where n is the length of the sequence, since any network which correctly sorts all binary sequences of length n also correctly sorts any arbitrary numeric sequence of length n.
2. Assuming such average (or perhaps an expectation) is well defined and that g has a maximum.

damage scenarios' include, for instance, aggregating over damages by taking the worst value rather than the average (worst-case optimization, also called maximin solution concept) or requiring Pareto-dominance, like in multi-objective optimization with the damages playing the role of objectives (Popovici et al., 2010).

Thus, resilient piping design is clearly an optimization problem: we have a set of potential solutions S and we are searching for the element of S that maximizes a function $g : S \to \mathbb{R}$. More specifically, in the field of optimization, this type of problem is called a test-based co-optimization problem. This is because the function g is in fact defined via the metric M, whose definition involves an additional set of entities T. The elements of T are called tests: they do not contribute to the definition of the set of potential solutions S, but they help test which elements of S are solutions and which are not, via the metric M. For brevity, from here on we will use just the word co-optimization when we actually mean test-based co-optimization; co-optimization problems that are not test-based exist, but are outside the scope of this paper.[3]

Note also that we do not have a closed-form mathematical formula for M. Rather, the resilience of a design to a damage is determined through a computational procedure (this could involve graph-traversal algorithms as mentioned above, or, in some cases, a fluid dynamics simulation). While we might know what this procedure is, this knowledge does not easily (or even at all) provide us with a roadmap for solving the problem. What we *can* always do though is to evaluate M to get its output for one input-pair $\langle s, t\rangle$ at a time; $\langle s, t\rangle$ is typically called an interaction or event and determining $M(s, t)$ is referred to as performing a metric evaluation or M-evaluation or, for short, an evaluation; we also say that s has seen t (and t has seen s). For each such evaluation we incur the computational cost of running M (ranging from milliseconds for graph-traversal algorithms to minutes or hours for hydraulics simulations); this is the basic unit of cost for co-optimization algorithms. We say that M is accessible in black-box fashion. In some domains, M may in fact be the outcome of a real-world experiment which we do not know how to simulate computationally, in which case M is truly black-box and even more costly (e.g., days and dollars).

A key point in co-optimization is that the function whose maximum we are trying to find is not the black-box M, but g, which is an aggregate of $|T|$ values of M. As previously discussed, the size of T can be such that computing the actual value of g for even a single $s \in S$ can be very costly or completely infeasible. Even estimating $g(s)$ will typically require evaluating multiple interactions involving s. This is in contrast with traditional black-box optimization, where the quality of a potential solution is given by a single evaluation, since it is potential solutions that are the object of black-box evaluation, not interactions. It is also in contrast with multi-objective optimization, where the number of objectives is so small that evaluating each potential solution on *all* objectives is not an issue and may in fact be considered the basic unit of cost; in co-optimization the set T is something that needs to be explored. Single- and multi-objective optimization could also be seen as instantiations of the more generic framework of co-optimization.[4]

3. In particular, the field of co-optimization also comprises a subfield concerned with so-called *compositional* co-optimization problems (Popovici et al., 2010), some of which have been approached via cooperative coevolutionary algorithms (Potter, 1997).
4. But see also work mapping test-based co-optimization to multi-objective optimization with fewer *underlying* objectives than the size of T (Bucci et al., 2004).

With a bit of reshaping, the sorting network problem can similarly be cast. Let S denote the set of all possible networks made up of a fixed number of compare-and-swap gates and taking as inputs sequences of a fixed length. Let T denote the set of all binary sequences of that length (see footnote 1 on page 4). For any network $s \in S$ and any input sequence $t \in T$, we can determine sorting correctness by running the sequence through the network, then checking if the output is in fact sorted. Let $M : S \times T \to \{0, 1\} \subset \mathbb{R}$ be a metric such that $M(s, t) = 1$ if s correctly sorts t and 0 otherwise. Like for the piping network design problem, M is not given by a closed-form mathematical formula, but for any input-pair $\langle s, t\rangle$ we can compute $M(s, t)$ via the above procedure. As originally stated, the goal for sorting network design is to find a network $s \in S$ such that $M(s, t) = 1$ for all $t \in T$. This is an example of a test-based co-search problem, in that we are searching for an element with a particular property. A co-optimization problem is a co-search problem in which the property of the element we're looking for is that the element optimizes some function. The sorting network problem can be re-phrased as a test-based co-optimization problem, since an s satisfying the above property also maximizes $g : S \to \mathbb{R}$, $g(s) = \sum_{t \in T} M(s, t)$. Also note that for this problem an M-evaluation has fairly modest cost (linear in the number of compare-and-swap gates and likely measured in milliseconds), but the size of T is exponential (2^n, where n is the length of the input sequence).

A subtle difference between the resilient piping problem and the re-stated sorting networks problem is that for the latter it is really important that we find a maximum of g and we might be able to tell whether or not we have, whereas for the former we would not recognize a maximum even if we found one, so we're really looking for as high a value of g as we can find. Nonetheless, any method we can apply to the generic, black-box formulation of the resilient piping problem we can also apply to the sorting networks problem. Problems like these are the object of study in the field of co-optimization (Service and Tauritz, 2008a; Popovici et al., 2010; Popovici and Winston, 2015).

Supervised Learning Formalisms and Mapping to Co-optimization. We now turn our attention to the latter two problems, which come from the field of supervised machine learning (Abu-Mostafa et al., 2012). For the photo labeling problem, let X be the set of all possible photos (up to a certain size limit). Let $Y = \{yes, no\}$ be the set of labels, denoting presence or absence of balloons. Let $f : X \to Y$ denote the correct assignment of labels to photos, where correctness can always be judged by humans. This is often referred to as the target function to be learned. We are provided with a set of n labeled examples $\mathrm{D} = (x_i, y_i = f(x_i))_{i=1..n} \subset X \times Y$ and the goal is to find the full f, or, more precisely, to find a computer program that implements f. Such a program is often referred to as a classifier or predictor, the examples D are also referred to as data points or simply as data or a data set, and the problem solving goal is also referred to as (binary) classification.

We can map this problem to a test-based co-optimization problem as follows. Let S be the set of all computer programs that take as input a photo $x \in X$ and output a label $y \in Y$. For any $s \in S$ and any $x \in X$ we denote by $s(x)$ the output of program s for input photo x. Note $|S|$ is larger than $|Y^X|$, since multiple programs can implement the same functionality. Let $T = X$ (i.e., photos constitute tests) and define $M : S \times T \to \mathbb{R}$ via $M(s, t) = \delta(s(t), f(t))$, where δ is the Kronecker delta function or some other function measuring the agreement between $s(t)$ and $f(t)$. Then the photo labeling goal is to find a program $s \in S$ such that $M(s, t) = 1$ for all $t \in T$. This is essentially the same goal as

for sorting network design and can be restated as maxisum, which is useful since for this particular application we do not actually expect to find a perfect program. Equivalently and more commonly, M is defined as $1 - \delta(s(t), f(t))$ and called error, which is then to be minimized.

The problem of predicting secondary protein structure is similar: X is the set of all possible primary-structure amino-acid sequences, Y is the set of all possible secondary-structure sequences and f represents the real-world mapping of primary to secondary structure sequences. When mapping to co-optimization, S is the set of all computer programs that take as input a primary-structure amino-acid sequence and output a secondary-structure sequence, $T = X$ and $M(s, t)$ is defined to measure agreement between $s(t)$, the secondary-structure sequence output by program s for amino-acid sequence t, and the true secondary-structure sequence given by $f(t)$; agreement could be binary (1 if sequences are identical, 0 otherwise) or it could give partial credit for those positions that do match between the two sequences. The problem-solving goal, especially for the second choice of M, is once again maxisum.

Comparison. While these formalisms show us how we can express the problem-solving goals of typical supervised learning tasks in a similar fashion to those of co-optimization, they also expose subtle but important differences in the information available to any algorithm attempting to solve such problems. One such difference is the access we have to the function M. For generic co-optimization problems like resilient piping and sorting network design, we can evaluate M for any $\langle s, t\rangle$, thus the main concern is with the cost of each such evaluation and, consequently, with the number of evaluations that can be afforded and how good a potential solution (as judged via g) we can find given such an evaluation budget. For supervised learning problems like photo labelling and protein structure prediction, we can only get at the value of M for pairs $\langle s, x_i\rangle_{i\in 1..n}$ corresponding to the data set D (which is provided as input to the algorithm for free) and the main concern is how good a potential solution we can find given only those n specific labeled examples. Note how n does not impose a hard limit on the number of $\langle s, x_i\rangle$ pairs for which we can compute M, but see more on cost in Section 2.3.1.

Having made this difference explicit, we can now see that the related field of supervised active learning (Settles, 2012) concerns itself with problems whose information availability lies on a spectrum between the above two extremes.

2.1.4 Active Learning Problems

For instance, consider the following slight variation of the photo labeling problem. The goal is the same as before, but instead of a set of labeled photos we have a much larger set of unlabeled photos (obtained, for instance, via internet crawling). With the advent of the Amazon Mechanical Turk (Amazon, 2005) obtaining a label for a photo can be relatively cheap and prompt, though it is not completely free, and the time required to get it is not computational in nature, but rather depends on human availability (Maji, 2011). Both to save money and to speed up the process, it would be beneficial if we could ask for a label only for those images most useful for the learning task. This is referred to as a pool-based active learning problem. Formally, we are given X the set of all possible photos, Y the set

Symbol	Description
S	Space of potential solutions.
T	Space of tests.
$V \subseteq \mathbb{R}$	Space of metric values.
$M : S \times T \to V$	Interaction metric; $M(s, t)$ is the result of evaluating potential solution s with test t; it can be completely black box or based on labels in supervised learning.
$g : S \to \mathbb{R}$	Quality function that is the target of optimization; $g(s)$ gives quality of potential solution $s \in S$; examples include taking the average, sum or minimum of M over T.
X	Space of inputs, from which data is sampled.
Y	Space of outputs (labels for inputs).
$f : X \to Y$	Target function from Y^X, labeling inputs; it defines M when viewing supervised learning as co-optimization.
$\mathrm{D} \subset X \times Y$ $\mathrm{D} = (x_i, f(x_i))_{i=1..n}$	Data set of size n from which f must be inferred.
$\mathcal{H}$	In supervised learning, set of hypotheses considered (also called a model). In co-optimization, set of potential solutions actually being searched. Its mapping to Y^X or, respectively, S is typically not one-to-one.

Table 1: Notation summary for Section 2.1 (Problems) and Section 2.2 (Algorithms).

of labels, a strict subset $X' = (x_i)_{i=1..n}$ of X and access to f in black-box fashion with the restriction that we can only query f for $x_i \in X'$.

If mapping this to co-optimization, the restriction on access to f means in turn that we can only get the value of M for pairs $\langle s, x_j\rangle$ for those $x_j \in X'$ for which we have obtained a label. However, while in generic co-optimization one metric evaluation is the basic unit of cost, in active learning if we perform multiple $M(s, x_j)$ evaluations with different s but the same x_j we incur the cost of obtaining the label $f(x_j)$ only once overall and the costs for $s(x_j)$ and $\delta(s(x_j), f(x_j))$ multiple times, once for each s.

This is true for two additional flavours of active learning. One further variation of the above problem consists of the case when we are not provided with a set $X' \subset X$, but with a stream of elements from X: each element is available only for a limited time, during which the algorithm can ask for a label for that element, thus it must decide quickly whether or not to do so—and we cannot ask for all labels. This setup is of interest for domains such as part-of-speech tagging (Dagan and Engelson, 1995).

Let us also revisit the protein structure problem and relax the requirement that we use only data previously generated by lab experiments. Rather, we are allowed to perform further experiments for any existing amino-acid chain to determine its secondary structure, though for any such experiment we incur costs both in money and in time required to perform the experiment. Note that for this setting f is truly black-box and we have no restrictions as to for which examples $x \in X$ we can obtain the label $f(x)$—examples

can be generated de-novo. Consequently, if we model this as co-optimization, we have no restrictions in terms of which pairs $\langle s, x\rangle$ we can compute M for. Thus, this problem is the most similar to the co-optimization problems of resilient piping design and sorting network design. The difference is that for those problems it was M directly that was given in black-box fashion, whereas for the de-novo active learning problem we have that M is defined through f and it is f that we have black-box access to. The impact of this difference will be discussed in Section 3.

For all active learning problems there may be a budget constraining how many labels the algorithm can obtain.

2.2 Algorithms

We now turn to discussing the various types of algorithms that have historically been used to approach supervised learning and co-optimization problems. Even state-of-the art algorithms tend to fall under these broad types or employ as components the core approaches described here and hybrids thereof; when they don't, this framework should make it easy to express how they differ. Our categorization is mainly driven by the way algorithms make use of the information available to them. Sections 2.2.1 and 2.2.2 will read familiar to the machine learning researcher; we review some basic notions so that: 1) whenever possible, we can directly describe them using co-optimization terminology; and 2) we can refer back and contrast with them the co-optimization-specific notions we introduce later in Section 2.2.3 and thus build towards the vocabulary mapping in Table 3.

2.2.1 Supervised Learning Algorithms

Recall from the formalization of supervised learning in Section 2.1.2 that we are looking for an element $s \in S$ that minimizes $g(s) = \sum_{t\in T} M(s,t) = \sum_{t\in T}(1 - \delta(s(t), f(t)))$, but we only have access to the value of $f(t)$ for $t \in (x_i)_{i=1..n}$ from the data D. Thus, a first issue to be addressed is that we *cannot* actually compute $g(s)$ for any s. Supervised learning algorithms deal with this by working with restrictions of the sum inside g to subsets of the data. State-of-the-art methods employ meta-approaches such as regularization or cross-validation (discussed at the end of this section), but at their core they incorporate and expand upon the following procedure. Data is randomly partitioned into a training set and a test set. An $s \in S$ is identified that minimizes g^{tr}, the variant of g obtained by summing $M(s,t)$ only over the t in the training set, also called in-sample error or training error. It is hoped that $g^{tr}(s)$ will correlate with the actual value of $g(s)$, though of course the former is an optimistically-biased estimator of the latter. For that reason, for the best s *only*, an unbiased estimator is computed by summing $M(s,t)$ only over the t in the test set. Overfitting or over-training is said to have occurred if an s was chosen that had low training error, but high test set error. The term generalization error is used to refer either to the actual value of $g(s)$ or to the out-of-sample error defined as the sum of $M(s,t)$ only over the $t \in T \setminus (x_i)_{i=1..n}$ (the symbol '$\setminus$' stands for set difference).

In minimizing g^{tr}, the next issue concerns the set S. For supervised learning problems, the true space of all potential solutions S is the set of all computer programs implementing a function from X to Y. In practice though, supervised learning algorithms only consider a subset $\mathcal{H} \subset S$ of programs, but one with a structure that facilitates finding a program that implements f or a good approximation of f. The elements of $\mathcal{H}$ are referred to as hypotheses. For instance, $\mathcal{H}$ could be the set of neural networks of a particular size and shape, or the set of decision trees of a certain depth, etc. Such a set $\mathcal{H}$ is typically called a model.[5] There may be multiple elements of $\mathcal{H}$ that implement the same function from X to Y and there may be functions from X to Y that no element of $\mathcal{H}$ implements (i.e., the mapping from $\mathcal{H}$ to Y^X is not one-to-one). The choice of $\mathcal{H}$ (and an algorithm to search it) has a strong bearing on the expected generalization error and carries with it a so-called bias-variance tradeoff (Geman et al., 1992), where bias and variance represent two different sources of said error that are difficult to minimize simultaneously. Bias is the average, over possible data sets, of the distance between the target function and the best hypothesis found by the algorithm for that data set. The variance measures how much this best hypothesis varies with the data set. Larger, more *complex* sets $\mathcal{H}$ tend to have lower bias but higher variance and be prone to high generalization error via overfitting. Smaller and simpler sets $\mathcal{H}$ tend to have lower variance but be prone to high generalization error due to high bias (underfitting).

For many supervised learning approaches, the choice of $\mathcal{H}$ requires that the data is either directly given as or can be transformed into tabular form, i.e., the set X can be written as $X^{\mathrm{f}} = X_1^{\mathrm{f}} \times X_2^{\mathrm{f}} \times ... \times X_k^{\mathrm{f}}$ where the sets X_i^{f} are numeric, ordinal or categorical in nature (the superscript 'f' stands for 'feature'). These sets represent so-called variables or features or (in statistics) covariates. In certain domains the elements of X in fact have richer structure, which cannot be represented in tabular form. When mapping such an X into an X^{f} there will inevitably be information loss; additionally, the choice of mapping could have an impact on the difficulty of the problem.

At the core of even sophisticated machine learning methods there are two main ways in which the structure of $\mathcal{H}$ helps locate an element of $\mathcal{H}$ that minimizes g^{tr} and they differ in the way they make use of the training data: constructive versus exploratory.[6]

Constructive approaches rely on a mapping of X to X^{f} and define $\mathcal{H}$ in such a way that the training data can be used to directly construct a *single* element $s \in \mathcal{H}$ with a good g^{tr}-value (low training error). From a co-optimization standpoint, this constructive process used during training does not involve any $M(s,t)$ evaluations. For instance, for decision trees (Rokach and Maimon, 2014) this is done by repeatedly partitioning the examples based on one feature X_i^{f} at a time, by using the relationship between the feature's values for the training data points and the respective labels of those data points; a new node is added to the tree with each partition until a maximum depth is reached. For support vector machines (SVMs, Cristianini and Shawe-Taylor, 2000), $\mathcal{H}$ is a set of polynomials which, along with some constraints, makes the problem of minimizing training error into a quadratic programming problem.

Exploratory approaches work by computing the training error g^{tr} for *multiple* elements of $\mathcal{H}$ in order to find one with a good value. In co-optimization lingo, they perform a series of $M(s,t)$ evaluations for various $s \in \mathcal{H}$ and various t in the training set portion of $(x_i)_{i=1..n}$ in order to decide which s to pick. This search process can be strongly or loosely guided.

5. In some works, the word 'model' is used to mean a single hypothesis. To avoid confusion, in this paper 'model' only refers to a set of hypotheses $\mathcal{H}$.
6. These have also been referred by De Jong (2006) as bottom-up and, respectively, top-down.

One way to provide strong guidance to the search is to map X to X^{f} and define the space $\mathcal{H}$ so that the error g^{tr} has a closed form with easily computable derivative that allows gradient descent techniques to be used for minimization. Neural networks are an example of this (Rumelhart et al., 1986): the X_i^{f} are mapped to the input nodes of the network, $\mathcal{H}$ is the set of all possible weight combinations for the edges and the error function is linear in these weights. Forward propagation constitutes performing one M-evaluation.[7] Back-propagation of the error to update the weights via gradient descent generates a new network $s' \in \mathcal{H}$. Using a single example for each back-propagation step generates a series of evaluations of the type $M(s_1, t^1), M(s_2, t^2), M(s_3, t^3), \ldots$. Using multiple examples and aggregating the error before propagating it back leads to a series of evaluations of the type $M(s_1, t_1^1), M(s_1, t_2^1), \ldots, M(s_1, t_i^1), M(s_2, t_1^2), M(s_2, t_2^2), \ldots, M(s_2, t_j^2),\ M(s_3, t_1^3), \ldots$. In either case, the s with best g^{tr}-value (often last in the series) is output. Choosing the order in which to use the examples is typically done stochastically and orderings resulting from different random seeds lead to different exploratory paths through $\mathcal{H}$.

An example of a loosely-guided exploratory approach is the application of genetic programming (Koza et al., 2006) to supervised learning. Genetic programming is a set of techniques from the field of evolutionary computation (De Jong, 2006) that draw inspiration from the principle of evolution by natural selection to search for computer programs achieving some task. Programs can take various shapes, such as rule-sets in the field of learning classifier systems (Lanzi et al., 2003), expression trees in symbolic regression (Icke and Bongard, 2013), even neural networks with variable structure in neuro-evolution (Floreano et al., 2008), etc. The set X, representing inputs to the programs, need not necessarily take tabular X^{f} form and the space $\mathcal{H}$ of programs is not constrained to lead to a closed-form error function. Instead, $\mathcal{H}$ is endowed with a topology such that neighboring points (programs) in $\mathcal{H}$ have close error values. This topology is induced by so-called genetic operators that generate new programs in $\mathcal{H}$ from previous programs, in a manner inspired by how mutation and crossover work on DNA. A program $s \in \mathcal{H}$ is evaluated by computing $M(s, t)$ for some (usually all) t from the training-set portion of $(x_i)_{i=1..n}$ and aggregating these measurements into one value, often referred to as fitness. Programs with good fitness (low error) are selected and operators are applied to them to produce a population of new programs; the process then repeats for multiple iterations called generations. These techniques are slower at climbing any given local optima than pure-gradient methods—since they essentially have to indirectly estimate the gradient by performing $M(s, t)$ evaluations—but better at avoiding getting stuck in one. Note also that they are general optimization techniques applicable to other tasks besides supervised learning and therefore not tuned to it.

Meta-approaches. Both exploratory and constructive approaches need to be concerned about over-/under-fitting. One heuristic is to try out sets $\mathcal{H}$ of different complexities: for example, put a hard constraint on the degree of the polynomials or the depth of the trees considered, then investigate what happens when varying the value of the constraint. Another heuristic is to use "soft" constraints, i.e., allow an all-encompassing $\mathcal{H}$, but include in the training error g^{tr} a penalty term for the complexity of individual hypotheses in that $\mathcal{H}$; examples include: allowing trees of high maximum depth, but including a penalty term that grows with the depth; or allowing polynomials of high degree, but including a penalty term that is low when most weights are close to 0 and high otherwise. This latter heuristic is generally referred to as regularization; in genetic programming, penalizing tree depth is called parsimony pressure. Different levels must be tried out for the relative contribution of the penalty term versus the main error term.

7. Feeding example t through network s computes $s(t)$ and determining disagreement between the network's output and the example's label computes $M(s, t) = 1 - \delta(s(t), f(t))$.

In the above, the expression 'try out' refers to the fact that the overall algorithm applies multiple basic methods (either exploratory or constructive), or applies a parameterized basic method multiple times with different parameter values, then selects one of the variants for producing the final output—a process known as model selection. To perform the selection, (semi)unbiased empirical estimates are computed for the output of each variant by performing additional $M(s, t)$ evaluations with tests not used during the training of that variant—a process known as (cross-)validation (Refaeilzadeh et al., 2009; Geisser, 1975; Stone, 1974). Thus, during this process, what was originally designated as a training set gets further split (possibly multiple times for cross-validation) into an *actual* training set and a validation set. The term holdout set has also been used in the literature, but with inconsistent meaning, referring sometimes to a validation set and sometimes to a true test set. The test set is still to be used only once to get an unbiased estimate for the quality of the final output. Such uses of estimates for output selection and performance evaluation are relevant for co-optimization algorithms as well, as discussed in sections 2.2.3 and 2.3.2.

2.2.2 Active Learning Algorithms

Typical approaches to active learning treat the problem as an extension of supervised learning. As such, they use a query strategy to decide for which points $x \in X$ to obtain a label, then apply one of the supervised learning methods described above on the labeled data accumulated so far, then re-iterate these two steps. Some query strategies are generic and could be used with any supervised learning approach, whereas others are specific to the supervised learning algorithm used (Settles, 2012). Whether the core supervised learning approach is constructive or exploratory, *the query strategy of active learning algorithms can be thought of as exploring the set X*. The resemblance of active learning to co-optimization suggests there may be approaches that are more direct than the standard iterative one. We will revisit this at the end of Section 3.

2.2.3 Co-optimization Algorithms

Because in co-optimization it is the metric M that is given in black-box fashion (as opposed to an f that further defines M), it is not obvious if or how one could find a good element of S through a constructive approach. Therefore, co-optimization algorithms (Service and Tauritz, 2008a) are exploratory by necessity and involve adaptations of stochastic search and optimization heuristics (Luke, 2013) like simulated annealing (SA), hill-climbing (HC) and evolutionary algorithms (EAs)—the genetic programming described in Section 2.2.1 being an example of the latter.

Note that in this case the search space S does not necessarily contain computer programs, but possibly other kinds of objects or structures (e.g., subsets of valves in the case of resilient piping design or, often, simply vectors of parameter values characterizing a structure or design). Recall that supervised learning algorithms actually search a hypothesis set $\mathcal{H}$ that does not have a one-to-one mapping with Y^X; similarly, in co-optimization, the

way the space S is represented in the computer often means that what's actually being explored by the algorithm is a set other than S and whose mapping to S is not one-to-one. To limit notation, we'll denote this set by the same symbol $\mathcal{H}$. In the field of evolutionary computation (De Jong, 2006), the choice of $\mathcal{H}$ is called a representation and the mapping from $\mathcal{H}$ to S is called the genotype-to-phenotype mapping. One still needs to consider the tradeoff between the ability of $\mathcal{H}$ to represent many/all the elements of S and the ease of searching $\mathcal{H}$ (which may decrease as the size of $\mathcal{H}$ increases). This can be considered a counterpart concept to supervised learning's bias-variance tradeoff.

As in supervised learning, an important issue is the fact that we might not be able to compute the actual value of the quality function g that we are to optimize, because $g(s)$ is defined as the sum of $M(s,t)$ for *all* $t \in T$ and T is very large and/or M is very expensive.

In contrast to supervised learning, there are no restrictions as to which elements $t \in T$ we can use in $M(s,t)$ evaluations. Consequently, *co-optimization algorithms explore not just the space S but also the space T, as well as the cross-product of the two.* Thus, a co-optimization algorithm can be formalized as having two components: 1) the exploration mechanism,[8] deciding which interaction between a potential solution $s \in S$ and a test $t \in T$ should be evaluated next via the metric M, given the sequence of interactions previously evaluated and their M-values, also referred to as the history[9]; and 2) the output mechanism,[10] deciding, based on such a history, which potential solution s to output. This formalism can be used to describe exploratory approaches to supervised learning as well; and most meta approaches perform some kind of output selection even when incorporating constructive components.

Exploration mechanisms. A basic but sometimes successful approach is to use any of the previously-mentioned stochastic search heuristics for exploring S, by evaluating any considered $s \in S$ with a random sample of tests $t \in T$ and using as fitness an aggregation of the resulting M-values, a placeholder for the true quality of s. This sample may contain a portion that is fixed throughout the entire algorithm, a portion that is regenerated for each iteration and a portion that is regenerated for each s.

Another popular approach is to apply the evolutionary metaphor to both spaces. The resulting *coevolutionary* algorithms (Popovici et al., 2010) maintain a population of potential solutions $s \in S$ and a population of tests $t \in T$. Elements in the former population are paired with elements in the latter and the resulting interactions are evaluated via the metric M. Various interaction methods can be used to determine who is paired (interacts) with whom (Popovici, 2006). For the maxiavg/maxisum solution concepts, fitness is typically assigned to each element s in the S-population by averaging $M(s,t)$ for all the interactions that particular s was involved in; such fitness represents a subjective quality, internal to the algorithm. Much research has gone into how to assign fitness to the elements of T, leading to two main paradigms: 1) if high values of M are good for the elements of S, then reward tests in T for low values of M (i.e., for being *difficult* tests)—a paradigm traditionally

8. This has also been called just an *algorithm* (Wolpert and Macready, 2005; Service and Tauritz, 2008b) or a *search heuristic* (Service and Tauritz, 2008b; Service, 2009a,b; Popovici and De Jong, 2009; Popovici et al., 2011).
9. This has also been called a *sample* (Wolpert and Macready, 2005) or simply a *sequence* (Service and Tauritz, 2008b).
10. This has also been called a *champion-selection rule* (Wolpert and Macready, 2005), a *champion selection function* (Service and Tauritz, 2008b), a *candidate selection function* (Service and Tauritz, 2008b; Service, 2009a,b) or *output selection* (Popovici and De Jong, 2009; Popovici et al., 2011).

referred to as *competitive* coevolution (Angeline and Pollack, 1993; Popovici et al., 2010); and 2) reward elements of T for their ability to differentiate between elements of S, that is for being *informative* tests (Bucci, 2007). In addition to the tests in the current population, archives of hard/informative tests can be kept and used in evaluations (Rosin and Belew, 1997; de Jong, 2005). Co-optimization problems emerged as a formalization of the kinds of problems coevolutionary algorithms were being applied to.

Both approaches (random sampling of tests versus coevolution) typically use the same *number* of tests to evaluate all potential solutions considered; these potential solutions are called *partially evaluated*, as said number of tests is usually much smaller than $|T|$.

Output mechanisms. A typical approach is *greedy*, which outputs the partially-evaluated potential solution with the best average of M-values over the tests it was evaluated with. It is also common to maintain archives of promising potential solutions (as identified by the greedy method) and then do another round of M-evaluation for them, with either freshly sampled or similarly archived tests, and pick the output based on the resulting *empirical* g-estimates (Panait and Luke, 2002); this is similar to the process of validation for model selection used in supervised learning. Newer work (Wolpert and Macready, 2005) has introduced optimal output mechanisms whose definition makes use of *theoretical* (Bayes) estimates of g consisting of aggregations (e.g., averages) over all possible values that yet-unperformed metric evaluations could result in. Interestingly—and unlike greedy—such output mechanisms even consider outputting *completely-unevaluated* potential solutions, i.e., ones for which the algorithm hasn't yet performed any M-evaluations, if their Bayes estimates are better. We will discuss these in more detail in sections 2.3.2 and 3.

Estimates of g, whether empirical or theoretical, can also be used to judge the quality of the outputted potential solution. In particular, using freshly sampled tests to obtain an unbiased empirical estimate (Chong et al., 2008) is similar to the way a test set is used in supervised learning; such an estimate represents an objective quality and is external to the algorithm itself. As with over/under-training in supervised learning, this is necessary because a disconnect can occur between internal and external quality measurements (Popovici and De Jong, 2005; de Jong, 2007).

2.3 Cost and Performance

In computer science the terms cost and performance are sometimes used interchangeably, especially when algorithms are acceptable only if they produce correct output and differentiating between them is done by measuring their execution speed. In supervised learning and co-optimization the notion of correctness is replaced by as-good-as-possible, thus we have to differentiate between two terms. We use the word 'performance' to refer to quantities having to do with the quality of potential solutions outputted by the algorithm. We use the word 'cost' to refer to time or money. Performance and cost are typically inter-related, since the performance of an algorithm may vary with the cost expended. We start by formalizing the various types of cost encountered in co-optimization and supervised learning, and hint at possible tradeoffs between different types. Then we review ways of assessing performance, which may itself incur costs.

The study in Section 3 relies upon and provides examples for several of the notions of cost and performance introduced here.

2.3.1 Cost

In general, costs can take several high-level forms: 1) computational time, that is the time required by a computer to perform a calculation; it can vary from one hardware to another, but may be expressible in terms of basic operations via the big-O notation of traditional computational complexity—this tends to be more difficult to achieve when the calculation is in fact a complex simulation; 2) real-world time, for instance time required to run a physical or chemical experiment, or time required to get a human to provide a label; and 3) money, for instance to pay for a lab, materials and people.

Algorithms for supervised learning and co-optimization problems can incur these kinds of costs while carrying out various activities. Some are directly driven by the problem specification: obtaining labels for active learning and evaluating $M(s,t)$ for co-optimization; all algorithms attacking these problems will necessarily incur the associated costs. Other cost-incurring activities are dependent on the specific choice of algorithm and include: exploration decisions, construction, output selection. Last but not least, in supervised learning and co-optimization evaluating the performance of any algorithm often incurs additional costs. Table 2 provides an overview in concise form.

Labeling costs are relevant only for *active learning* and are discussed extensively in Section 2.1.4, so we only re-summarize them on the top of Table 2. The only additional note is that while for all the problem examples in Section 2.1.4 labeling costs were in the form of real-world time or money, they can occasionally be measured in computational time: for instance, trying to find a quick-to-compute approximation for a very-slow-to-compute simulation (Sacks et al., 1989), such as one involving computational fluid dynamics or a weather model.

Metric-evaluation costs are part of the problem specification in *co-optimization*, as detailed in Section 2.1.1; their type and magnitude can vary, but even when computational and small it is a concern due to the very large sizes of S and T (often combinatorial or exponential) requiring a lot of M-evaluations to find good potential solutions.

Metric-evaluation costs are also relevant in *supervised learning*. They are a key concern for all exploratory algorithms, since exploration decisions are made based on the outcome of such evaluations. Additionally, incurring $M(s,t)$ evaluation costs may also be necessary for two other algorithmic activities discussed later in this section, namely output selection and performance assessment, which are relevant even for constructive approaches to supervised learning. The cost of evaluating $M(s,t)$ is always computational time and has two components: 1) cost of computing $s(t)$, that is the computational time required to run example t through program s (e.g., forward-propagation through a neural network, or top-down propagation through a decision tree, or evaluation of a polynomial for a support vector machine); and 2) cost of computing the agreement $\delta(s(t), f(t))$. The latter cost is typically small, e.g., constant when simply comparing photo labels or linear in the number of amino-acids when comparing protein structures. The cost of computing $s(t)$ varies with the size of individual hypotheses, e.g., number of edges in a neural network, levels in a decision tree (and trees in a random forest, Ho, 1995) or terms in a polynomial; it has grown in recent years for models such as deep learning's convolutional neural networks with millions of connections (LeCun et al., 2015). Using models with a low cost of computing $s(t)$ is desirable because this cost is also always incurred once a solution is deployed in the real world.

Table 2: Summary of the various types of costs encountered in test-based co-optimization and supervised learning.

		Supervised learning				Test-based Co-optimization
		Passive	Active			
			Pool-based	Stream-based	De-novo	
Labeling costs		Free ("gift" from someone else)	Pay for label Type: usually money or time, occasionally computational. Magnitude: varies widely across domains			Not applicable (N/A), there are no labels
Label availability	*How many ...*	Restricted by problem specification	Restricted mainly by labeling budget, then by pb. spec.	Restricted by labeling budget		
	... and which examples we can get labels for:	Restricted			Unrestricted	
Metric-evaluation (M(s,t)) costs		Type: always computational, two components *s(t)* and *δ(s(t), f(t))*; exclusive of cost of obtaining label *f(t)* Incurred during training by exploratory algorithms Incurred during testing and validation by all algorithms Magnitude: *s(t)* varies with hypothesis size, *δ* with label size Cost of s(t) always incurred in the real world by deployed solution				Main cost of interest, all algorithms incur it Type: usually computational, sometimes real-world time and/or money Magnitude: varies widely, a concern even if small due to extremely large *S* and *T*
		May also be incurred as part of ouput-selection and performance-assessment				
		Restricted by metric-evaluation budget				
M(s,t) availability	*How many ...*					
	... and which interactions <s,t>:	Restricted to *t* in training data	Restricted to *t* that we have chosen to obtain a label for (see label availability)			Unrestricted
Exploration-decision costs	*... to decide which interactions to evaluate next:*	Magnitude: negligible when picking the output of core training components; multiple of metric-evaluation costs when performing (cross-)validation for model selection				Always incurred, as all approaches are exploratory Magnitude: negligible compared to metric-evaluation costs for traditional approaches (EAs, SA, HCs), not so for newer ones (CMA-ES, EDAs, Bayes-optimal)
		Magnitude: for backprop, at most as much as metric-evaluation costs; for others, see co-opt. box to the right				
	... to decide which examples to label next:	N/A	Incurred by query strategy			N/A
Construction costs		Incurred by constructive approaches Magnitude: big-O complexity as function of data size				N/A
		Incurred many times if full retraining is required				
Output-selection costs		Magnitude: negligible for greedy methods; multiple of metric-evaluation costs when using empirical estimates; negligible to prohibitive for theoretical Bayes estimates				
		Incurred to determine which of the considered potential solutions to output as the algorithm's best guess for a solution				
Performance-evaluation costs		Magnitude: metric-evaluation costs over a single-use test set				Magnitude: multiple of metric-evaluation costs when using empirical estimates; negligible to prohibitive for theoretical Bayes estimates
		Incurred to estimate the actual, "true" quality of the outputted potential solution				

Type: usually computational, unless we collect new data or perform new experiments

Type: always computational

For *active learning*, note that the cost of obtaining the label $f(t)$ is not included in the cost of $M(s,t)$, because a label can be used for multiple computations of $M(s,t)$ for the same t but different s.

Exploration-decision costs are incurred to decide which problem-specification costs to incur next (e.g., which evaluations $M(s,t)$ an exploratory approach would perform, or, in the case of active learning, which labels a query strategy would request). They are always computational.

Back-propagation in neural networks for supervised learning is an example of an exploration decision incurring a cost: it produces a new network s for which to then evaluate $M(s,t)$ via forward propagation. The cost of one backward propagation has similar magnitude to that of one forward propagation and both vary with the size of the network, as described earlier. If multiple rounds of forward propagation are performed for each backward propagation, then, overall, exploration-decision costs are accordingly smaller than metric-evaluation costs.

Evolutionary approaches, whether used for *supervised learning* or *co-optimization*, usually incur exploration-decision costs while performing selection and breeding to produce a new generation of potential solutions, which will then be involved in $M(s,t)$ evaluations. Similarly, simulated annealing and hill-climbers incur exploration costs when generating and sampling from a local neighborhood. Historically, these costs have been considered much lower than $M(s,t)$ costs. But newer stochastic black-box optimization heuristics tend to have non-negligible exploration costs, such as: quadratic matrix computations for covariance-matrix adaptation evolutionary strategies (CMA-ES, Hansen and Ostermeier, 2001), estimating distributions for estimation of distribution algorithms (EDAs, Larranaga and Lozano, 2002), recursive optimization for optimal exploration mechanisms (Popovici and Winston, 2015; Popovici, 2017).

In *active learning*, exploration-decision costs are also incurred via the query strategy used to decide the next $x \in X$ for which to obtain a label. Such costs vary widely with the strategy (Settles, 2012) and have to be weighted against the costs of obtaining labels, since reducing the latter is the main goal of sophisticated query strategies .

Construction costs, as the name suggests, are incurred by constructive approaches to supervised learning, whether passive or active; examples include the construction of a decision tree and solving quadratic programming equations for support vector machines. They are computational costs, usually measured in basic operations and expressed via big-O complexity as a function of properties of the problem, such as size of data. They are of particular concern for the increasingly more common "big data", as well as in active learning, where these costs are incurred multiple times if full retraining is necessary each time new labels are obtained.

Output-selection costs are incurred in order to decide which potential solution to output out of the possibly multiple ones considered. This activity is closely-related to performance assessment, since the performance of an algorithm always relies in some manner on the quality of potential solutions it outputs. As discussed in Section 2.2, one distinguishing feature—and potential difficulty—of co-optimization and supervised learning problems is the inability to compute the actual quality of any potential solution for the problem instance at hand. Consequently, assessing quality of potential solutions must be done by means of estimation, which carries a cost.

In co-optimization, simple output selection methods just side-step the issue, by using the greedy heuristic of outputting the potential solution with the best average over only the tests seen so far, a highly-biased empirical estimate of g. Keeping track of this potential solution is trivial, so output selection costs have historically been considered negligible compared to metric-evaluation costs. As described in Section 2.2.3, there are also approaches that acknowledge the issue. Some compute new empirical estimates of g for a handful of potential solutions identified via the greedy method, then use these estimates (which could be unbiased or at least less biased) to select which one to output; this requires performing additional $M(s,t)$ evaluations and incurring the respective metric-evaluation costs. Others use theoretical g-estimates whose computation does not involve performing additional metric evaluations, but rather aggregating (e.g., averaging) over all possible values that such evaluations could result in. These aggregates can be computationally-prohibitive; but output selection only relies on comparisons between them, the outcome of which can sometimes be determined without actually computing the aggregates themselves (Popovici and De Jong, 2009; Popovici et al., 2011; Popovici, 2017), in which case the costs are minimal.

Unless we collect new data or perform new experiments for the purpose of additional $M(s,t)$ evaluations, output selection costs are only computational. This is the case for supervised learning as well. Here too, output selection costs can be negligible if the algorithm simply outputs the outcome of the training process for a particular method, for example, the decision tree constructed, or the final set of weights for a neural network (which is often the greedy best-so-far, given the gradient-based nature of exploration via back-propagation). More commonly though, a model is selected first via (cross-)validation, as described in Section 2.2.1. To compute the empirical estimates involved in validation, $M(s,t)$ costs are incurred by performing additional metric evaluations for the outputs of the models to be selected amongst.

2.3.2 Performance

By performance of an algorithm we mean the quality of the algorithm's outputted potential solutions with respect to the problem solving goal. But because in supervised learning and often in co-optimization as well exact quality is not actually computable, assessing algorithmic performance is an uncertainty-laced, more difficult task in these fields compared to traditional function optimization.

Perhaps the simplest performance question one can answer—and one that is quite important to answer in practice—is: on the problem instance at hand, what is the quality of the potential solution that constitutes our algorithm's final choice as *the solution* to be deployed in the real-world?

This is a quality-estimation question, so it can be answered the same way as when part of the output selection process (see previous section). But subtle care must be taken. Computing unbiased empirical estimates (by performing additional $M(s,t)$ evaluations and incurring the associated costs) must be done with fresh tests in co-optimization (Chong et al., 2008) or, in supervised learning, with labeled data points (a test set) not used for any other purpose.[11] Moreover, the resulting estimate *should not* be used for any further

11. Note the different usage of the word 'test' in the two parts of this sentence.

tweaking of the chosen potential solution, or else it ceases to be an unbiased estimate; in other words, a true test set is always single-use—but see the work of Dwork et al. (2015) for a novel technique where the bias resulting from reuse is small and quantifiable as a function of the amount of reuse.

We may also wish to know how performance of an algorithm varies as a function of given resources/cost expended, so we can determine how we may be able to boost performance. Ideally, such characterizations would take into account *all costs* listed in the previous section. In reality, different approaches have focused on different aspects of the performance-cost dependence.

In passive supervised learning, the main concern is with performance as a function of resources, in the form of size of available data. However, the cost of acquiring data is not counted as a cost incurred by the learning algorithm. Indirectly, this dependency on data size does describe a relationship between performance and algorithmic cost, because making use of more data takes more time, whether it is done in constructive or exploratory fashion. Because of this, a whole subfield of supervised learning concerns itself with instance selection, that is determining a subset of the labeled examples that is small, yet using it would result in similar output quality as using the whole set (Reinartz, 2002). In co-optimization, the main concern is with performance as a function of metric-evaluation costs. In active learning, the main concern is with performance as a function of labeling costs. Unlike passive supervised learning, the latter two fields *do* concern themselves with the dependency of performance on the cost of acquiring data, whether the data consists of values of M or values of f.

Algorithms often have one or more parameters whose values must be set. Thus another important issue is how performance varies with changes in parameters. In supervised learning, a common example of this comes from binary classification in the form of receiver operating characteristic (ROC) curves (Spackman, 1989) which plot how the error for one class changes as a function of the error for the other class while varying the value of the algorithm parameter investigated.

Being able to compare the performance obtained for two different values of one parameter of an algorithm is akin to being able to compare two different algorithms. While empirical quality estimates allow us to do that and to determine if one one algorithm is better than another for the one problem instance at hand, the question that immediately arises is whether some algorithms are better than others in a general sense (i.e., across all, or a large class of problem instances). This is important because designing or tuning a new algorithm for each problem instance is a time-consuming manual activity that might not be feasible in practice. For instance, imagine a factory that must schedule jobs on a set of machines susceptible to various breakdowns (Jensen, 2001) and must produce a new schedule every day for the jobs of that day. One cannot re-tune with such high frequency.

To answer the above question, one must make precise the meaning of 'better' and of 'general'. Theoretical Bayes estimates are intrinsically about performance on multiple problem instances. By definition, they are aggregations (e.g., expectations) over all possible hypotheses f or metrics M, assuming a certain probability distribution over such functions. Wolpert (1992, 1996) first introduced this view of algorithmic performance and argued that, to the extent that the goal of supervised learning is to generalize to *unseen* examples, the meaning of 'better' should really be based on out-of-sample error (off-training set (OTS) error)

Supervised Learning	**Co-optimization**
labeled example data point data set data	 test interaction event
data size	budget
training set	history
test set	fresh tests
hypotheses classifiers predictors	potential solutions population
model variables features covariates	representation search space
bias-variance tradeoff	genotype-to-phenotype mapping
unknown target	black-box metric
forward propagation labeling	evaluation
IID error generalization error out-of-sample, off-training set (OTS) error	quality function external, objective quality
in-sample error validation error	fitness internal, subjective quality
overfitting (over-training) underfitting	internal-external quality disconnect
regularization	parsimony pressure
gradient descent back-propagation query strategy instance selection	exploration mechanism genetic operators (e.g., mutation, crossover) interaction methods generation
model selection (cross-)validation	output mechanism
supervised learning (binary) classification active learning (pool-based, stream, de novo)	solution concept (e.g., maxiavg, maxisum, maximin) test-based co-optimization problem co-search problem
no free lunch (NFL)	

Table 3: Terminology map. Terms in respective cells for the two fields are not synonymous, but represent concepts that are similar or related (e.g., back-propagation can be thought of as being a type of exploration mechanism).

rather than on the overall error g (IID error, where IID stands for independently identically distributed). For the OTS notion, he showed that there is no free lunch (NFL), meaning that for any given data set: 1) if all hypotheses consistent with that data set are equally likely (i.e., uniformly distributed), then all algorithms have the exact same aggregated performance; and 2) for any two algorithms, there are as many distributions over hypotheses for which one algorithm is aggregately-better than the other for that data set, as there are distributions where the opposite relation between algorithms holds. If by 'better' we mean better actual g value (i.e., over all tests in T), then there is free lunch, but of a rather unsatisfying type: the only aggregate-performance differences between algorithms come from their in-sample error. And while overall error is non-increasing as a function of the data set size n, out-of-sample error can actually increase with n (Wolpert, 1992). All of this holds regardless of the amount of computation expended by the algorithms.

Wolpert and Macready (2005) adapted and applied this approach to co-optimization. Since in co-optimization there is no given data set, the meaning of 'better' *must* rely on the actual value of g over all tests in T and the "fair ground" on which we compare algorithms is not a data set but the budget of $M(s,t)$ evaluations they are allowed to perform. Under this setup, there is free lunch (i.e., some algorithms are aggregately-better than others) *even if* all metrics M are equally likely.[12]

Active learning has not been studied to a similar extent from a free lunch perspective. Cursory treatment by Wolpert (1996) claims negative implications of no free lunch theorems apply just as they do for passive supervised learning. The details of this are unclear though, as there are several different NFL theorems for passive supervised learning and they are conditioned on a given data set (or its size) and are generally concerned with error outside this data set, whereas in active learning there is no data set given upfront, generating data is a key part of the algorithms. The claim is also intriguing given active learning's similarities with co-optimization and the existence of free lunch in that field. Investigating this issue is subject for future work and the treatment in Section 3 should provide a base towards that: it uses the common ground built in Section 2 to get a deeper understanding of the commonalities and differences between the two fields from a free lunch perspective.

3. Free Lunch Study

Having laid out the various concerns and approaches to measuring performance and cost, we now use them to present a proof-of-concept method transfer from co-optimization to supervised learning. Specifically, we focus on the free lunch view of performance versus cost, where quality is measured by means of theoretical Bayes estimates and metric-evaluation costs are constrained by a budget. We show how optimal *exploratory* algorithms can be designed within this framework, but also emphasize how they might incur significant costs in some of the other categories listed in Section 2.3.1, such as exploration-decision costs and output-selection costs, thus arguing for the need of a more encompassing view on costs to be developed.

12. Wolpert and Macready (2005) originally showed this for the worst-case (maximin) solution concept, but similar results have later been shown for other solution concepts (Service and Tauritz, 2008b; Service, 2009a,b), including the maxisum that most resembles supervised learning (Popovici and De Jong, 2009; Popovici, 2017).

We structure the presentation in 3 incremental steps with respect to metric-evaluation costs: in Section 3.1 we describe the small-scale case of only one or two M-evaluations in total at most one of which is remaining in the budget; in Section 3.2 we progress to histories of arbitrarily-many already-evaluated interactions, but still at most one remaining to be evaluated; finally, in Section 3.3 we discuss the most general case of arbitrarily-large spent budgets and arbitrarily-large remaining budgets. In each section we first review the co-optimization perspective and results pertaining to free lunch and optimality, then derive and contrast their counterparts for supervised learning. We focus specifically on binary classification, but many of the ideas presented would apply in a multi-class situation. Throughout, we differentiate between the nature of free lunches for output mechanisms versus exploration mechanisms. We keep the presentation as informal as possible and support it with small but concrete examples; precise mathematical definitions of all concepts involved and proofs of the results can be found in the accompanying Online Appendix A.

3.1 Small Overall Budget

In this section we look at a spent budget of 0 or 1 evaluations and a remaining budget of 1.

3.1.1 Co-optimization

Consider a domain with $S = \{a, b, c\}$, $T = \{t_1, t_2\}$, $V = \{0, 1\}$ and $M : S \times T \to V$, meaning there are 3 potential solutions, 2 tests and binary outcomes for interactions between solutions and tests. To ease comparison with supervised learning, suppose the values in V represent some kind of penalty, such that lower values are better and we're trying to find the potential solution minimizing the sum of M over the tests (i.e., a minisum solution concept rather than maxisum). The total number of interactions is $|S| \cdot |T| = 3 \cdot 2 = 6$ and therefore the total number of possible metrics M for this domain is $|V|^{|S|\cdot|T|} = 2^6 = 64$. These are shown in columns 1–7 of Table 4, one per row, numbered from 0 to 63.

Output mechanisms concern themselves with which potential solution to output given the history of interactions evaluated so far and their M-values; this history reflects the budget spent so far; the remaining budget is not a concern. If the current history is an empty sequence H = [] (i.e., we haven't yet performed any evaluations, spent budget is 0) and we have no other information, then all 64 metrics are possible and, we assume, equally likely (more on this in Section 3.4). Columns 8–10 in the table show, for each metric, the actual value of g (i.e., the sum of M over all tests) for each of the 3 potential solutions. At the bottom, the expected value of g, averaged over all metrics, shows that in the absence of any data or information all 3 potential solutions are equally promising, with an expected g-value of 1. Due to the equal likelihood of all metrics, this can also be thought of as follows: the expected M-value for a potential solution on a test it hasn't yet seen is the average of the values in V, which is 0.5; summing that over two tests gives $2 \cdot 0.5 = 1$.

Consider now the situation where an algorithm has evaluated the interaction $\langle a, t_1 \rangle$ and observed value $M(a, t_1) = 1$, so now the history is H = $[\langle \langle a, t_1 \rangle, 1 \rangle]$ (spent budget is 1). What should the algorithm output as the most promising potential solution? As we can see from the columns 11–13 in Table 4, metrics from 0 to 31 are no longer possible, since they have $M(a, t_1) = 0$. Averaging g only over the remaining metrics (which we say are *consistent* with the history) yields a different expected value for potential solution a (1.5)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58
	Interactions and their evaluations via the metric M							H=[]			H=[[⟨s,t⟩,1)]																		Exploration choices for history H=[[⟨s,t⟩,1)], case s=a, t=t1																																							
											Case s=a, t=t1, s' = b or c			Case s=b, t=t1, s' = a or c			Case s=c, t=t1, s' = a or b			Case s=a, t=t2, s' = b or c			Case s=b, t=t2, s' = a or c			Case s=c, t=t2, s' = a or b			Choice 1: Test s=a with t'=t1≠t=t2. Let s''=s. Possible outcomes						Choice 2: Pick s'≠s=a and test with t=t1. Let s''≠s,s' Possible outcomes:												Choice 2: Pick s'≠s=a and test with t'=t2≠t=t1. Let s''≠s,s' Possible outcomes:											
																													M(s,t')=0			M(s,t')=1			M(s',t')=0						M(s',t')=1						M(s',t')=0						M(s',t')=1					
																													H=[[⟨a,t1⟩,1), ⟨b,t1⟩,0)]			H=[[⟨a,t1⟩,1), ⟨a,t2⟩,1)]			Case s'=b			Case s'=c			H=[[⟨a,t1⟩,1), ⟨b,t2⟩,1)]			H=[[⟨a,t1⟩,1), ⟨c,t2⟩,1)]			Case s'=b			Case s'=c			H=[[⟨a,t1⟩,1), ⟨b,t2⟩,0)]			H=[[⟨a,t1⟩,1), ⟨c,t2⟩,0)]			Case s'=b			Case s'=c		
	⟨a,t1⟩	⟨a,t2⟩	⟨b,t1⟩	⟨b,t2⟩	⟨c,t1⟩	⟨c,t2⟩	g(a)	g(b)	g(c)		g(a)	g(b)	g(c)	g(a)	g(b)	g(c)	g(a)	g(b)	g(c)	g(a)	g(b)	g(c)	g(a)	g(b)	g(c)	g(a)	g(b)	g(c)	g(a)	g(b)	g(c)	g(a)	g(b)	g(c)		g(a)	g(b)	g(c)	g(a)	g(b)	g(c)		g(a)	g(b)	g(c)	g(a)	g(b)	g(c)		g(a)	g(b)	g(c)	g(a)	g(b)	g(c)		g(a)	g(b)	g(c)	g(a)	g(b)	g(c)
0	0	0	0	0	0	0	0	0	0	0																																																												
1	0	0	0	0	0	1	1	0	0	1																																																									
2	0	0	0	0	1	0	1	0	0	2																																																								
3	0	0	0	0	1	1	2	0	1	0																																																								
4	0	0	0	1	0	0	1	0	1	1																																																							
5	0	0	0	1	0	1	2	0	1	2																																																						
6	0	0	0	1	1	0	2	0	2	0																																																					
7	0	0	0	1	1	1	3	0	2	1																																																				
8	0	0	1	0	0	0	1	1	0	0																																																			
9	0	0	1	0	0	1	2	1	0	1																																																		
10	0	0	1	0	1	0	2	1	0	2																																																	
11	0	0	1	0	1	1	3	1	1	0																																																
12	0	0	1	1	0	0	2	1	1	1																																															
13	0	0	1	1	0	1	3	1	1	2																																														
14	0	0	1	1	1	0	3	1	2	0																																													
15	0	0	1	1	1	1	4	1	2	1																																													
16	0	1	0	0	0	0	1	1	0	2																																													
17	0	1	0	0	0	1	2	1	1	0																																												
18	0	1	0	0	1	0	2	1	1	1																																											
19	0	1	0	0	1	1	3	1	1	2																																										
20	0	1	0	1	0	0	2	1	2	0																																										
21	0	1	0	1	0	1	3	1	2	1																																									
22	0	1	0	1	1	0	3	1	3	0																																								
23	0	1	0	1	1	1	4	1	3	1																																								
24	0	1	1	0	0	0	1	2	0	0																																								
25	0	1	1	0	0	1	2	2	0	1																																							
26	0	1	1	0	1	0	2	2	0	2																																						
27	0	1	1	0	1	1	3	2	0	3																																					
28	0	1	1	1	0	0	2	2	1	0																																				
29	0	1	1	1	0	1	3	2	1	1																																				
30	0	1	1	1	1	0	3	2	2	0																																				
31	0	1	1	1	1	1	4	2	2	1																																				
32	1	0	0	0	0	0	1	1	0	2																																				
33	1	0	0	0	0	1	2	1	1	0																																				
34	1	0	0	0	1	0	2	1	1	1																																				
35	1	0	0	0	1	1	3	1	1	2																																				
36	1	0	0	1	0	0	2	1	2	0																																				
37	1	0	0	1	0	1	3	1	2	1																																				
38	1	0	0	1	1	0	3	1	3	0																																				
39	1	0	0	1	1	1	4	1	3	1																																				
40	1	0	1	0	0	0	1	2	0	0																																				
41	1	0	1	0	0	1	2	2	0	1																																				
42	1	0	1	0	1	0	2	2	0	2																																				
43	1	0	1	0	1	1	3	2	0	3																																				
44	1	0	1	1	0	0	2	2	1	0																																				
45	1	0	1	1	0	1	3	2	1	1																																				
46	1	0	1	1	1	0	3	2	2	0																																				
47	1	0	1	1	1	1	4	2	2	1																																				
48	1	1	0	0	0	0	1	1	0	2																																				
49	1	1	0	0	0	1	2	1	1	0																																				
50	1	1	0	0	1	0	2	1	1	1																																				
51	1	1	0	0	1	1	3	1	1	2																																				
52	1	1	0	1	0	0	2	1	2	0																																				
53	1	1	0	1	0	1	3	1	2	1																																				
54	1	1	0	1	1	0	3	1	3	0																																				
55	1	1	0	1	1	1	4	1	3	1																																				
56	1	1	1	0	0	0	1	2	0	0																																				
57	1	1	1	0	0	1	2	2	0	1																																				
58	1	1	1	0	1	0	2	2	0	2																																				
59	1	1	1	0	1	1	3	2	0	3																																				
60	1	1	1	1	0	0	2	2	1	0																																				
61	1	1	1	1	0	1	3	2	1	1																																				
62	1	1	1	1	1	0	3	2	2	0																																				
63	1	1	1	1	1	1	4	2	2	1																																				
Expected g for given potential solution and H											1.5	1	1	1	1.5	1	1	1	1.5	1.5	1	1	1	1.5	1	1	1	1.5	1	1	1	2	1	1	1.5	0.5	1	1.5	1	0.5	1.5	1.5	1	1.5	1	0.5	1.5	0.5	1	1.5	1	0.5	1.5	1.5	1	1.5	1	1.5
Expected g above, averaged over cases											s			s			s			s'											s			s'						s			s'						s			s'						s			s'		
											1.5									1											1			1						1.5			0.5						1.5			0.5						1.5			1		
Expected g for best output given H (minimum error)											1																			1			1			0.5						1						0.5						1					
Expected g for given exploration decision, averaged over possible outcomes											1																			1			1			0.75						1						0.75						1					

Table 4: Examples of co-optimization free lunch in domain with $S = \{a, b, c\}$, $T = \{t_1, t_2\}$, $V = \{0, 1\}$. Output mechanism free lunch (columns 11–28): for history $\mathrm{H} = [\langle\langle s, t\rangle, 1\rangle]$ with $s \in \{a, b, c\}$ and $t \in \{t_1, t_2\}$, outputting a new potential solution $s' \neq s$ yields strictly better expected performance (1) than outputting s (1.5). Exploration mechanism free lunch (right half of table): for history $\mathrm{H} = [\langle\langle s, t\rangle, 1\rangle]$, instantiated with $s = a$ and $t = t_1$, testing a new potential solution $s' \neq s$ with either the same test t or a new test $t' \neq t$ yields strictly better expected performance (0.75) than further testing s (1). Font style indicates which values were averaged to compute an expectation, green cell shading indicates where the minimum value is coming from and orange cell shading highlights the expected performance for each exploration choice.

than for b or c (1). Given these estimates, it follows that outputting one of b or c is the strictly best choice from the perspective of the algorithm's aggregated performance. This is an example of existence of *output mechanism free lunch* in co-optimization. Note this does not mean that outputting b or c is the best thing to do for each and every metric; for instance, if M happens to be M_{47}, outputting a would be the better choice.

Often, an algorithm might not be able to pick a specific potential solution and/or a specific test, rather it picks at random from a set of previously unexplored possibilities. Consider for instance a history $\mathrm{H} = [\langle\langle s, t\rangle, 1\rangle]$, where s is any of a, b or c and t is either t_1 or t_2, but we do not know which. There are 6 possible cases, depicted in columns 11–28 of Table 4. In each of these cases, the expected g-value of s is 1.5, so overall the expectation is also 1.5 and the expected g-value of any $s' \neq s$ is 1, thus outputting s' is the better choice. These values can also be arrived at with the kind of reasoning described a few paragraphs back: for s, the expected g-value is the sum of the actual M-value observed for t, which is 1, plus the expected value of M on the remaining test, 0.5; for $s' \neq s$, since it hasn't seen any tests yet, the expected g-value is the same as before: we simply multiply the expected M-value on a given test, 0.5, by the total number of tests, 2.

Exploration mechanisms concern themselves with which interaction to evaluate next given the history; this decision may be influenced by the remaining budget of M-evaluations. Suppose that the algorithm having produced history $\mathrm{H} = [\langle\langle s, t\rangle, 1\rangle]$ (spent budget of 1) is now allowed one additional evaluation (remaining budget of 1). It is the job of the exploration mechanism to decide which of the remaining 5 interactions to run through the metric M. And to determine the performance expected to result from a given interaction choice (exploration decision), we must aggregate over the potential outcomes of evaluating that interaction (0 or 1) the expected value of g for the best output for the ensuing lengthened history for that outcome. The exploration decision can be thought of as having two parts: choosing which potential solution to evaluate and choosing which test to evaluate it with. If s is chosen, then there is only one test it hasn't seen yet, $T \setminus \{t\}$. If an $s' \neq s$ is chosen, then there is a choice between evaluating s' with t or with the other test, $t' \neq t$. These choices are shown in columns 29–58 of Table 4, instantiated only with with $s = a$ and $t = t_1$, due to space constraints; the other instantiations yield the same kind of results. We can see that once an interaction is chosen and its M-value observed, the number of consistent metrics is further cut in half: there are as many metrics producing an M-value of 1 as of 0. Thus aggregating over outcomes is simply by averaging. And within each outcome, for each potential solution we average over the consistent metrics the g-value for that solution, then choose the potential solution with the best average. We see that the choice of evaluating an $s' \neq s$, regardless of whether with test t or t', leads to a better expected g-value (0.75) than evaluating s again with the remaining test t' (expected g-value 1). This is an example of existence of *exploration mechanism free lunch* in co-optimization.

3.1.2 Supervised Learning

Consider now a small binary classification domain with inputs $X = \{t_1, t_2, t_3\}$ and outputs $Y = \{y, n\}$ (short-hand for *yes*, *no*). There are a total of $|Y|^{|X|} = 2^3 = 8$ possible functions $f : X \to Y$. These are shown on the top left side of Table 5 (columns 1–4), one per row, numbered from 0 to 7. The problem is to guess f given some data D in the form

of pairs from $X \times Y$. Columns 5–28 in Table 5 show how this problem can be mapped to a co-optimization problem. We've already denoted the elements of X via the letter t because from a co-optimization perspective they serve as tests. For simplicity, let the space of potential solutions S have a one-to-one mapping with Y^X—though in practice this will often not be the case, as discussed in Section 2.1.3. So let $S = \{s_0, ..., s_7\}$ where s_i corresponds to f_i. Typically, an algorithm attempting to find an $s \in S$ that approximates f would know there is some mapping between S and Y^X, but may not be able to pick $s \in S$ guaranteeing specific outputs for given inputs (e.g., we don't know how to directly set the weights of a neural network to guarantee a certain error for a certain input). We have $T = X = \{t_1, t_2, t_3\}$ and $M : S \times T \to V = \{0, 1\}$, $M(s, t) = 1 - \delta(s(t), f(t))$ where δ is the Kronecker delta function. There are $|S| \cdot |T| = 8 \cdot 3 = 24$ possible interactions. In generic co-optimization, there would be $|V|^{|S| \cdot |T|} = 2^{24}$ possible metrics M, but since here M is fully determined by f, there are only 8 possible metrics M, fully-specified, one per row, in columns 5–28 of Table 5.

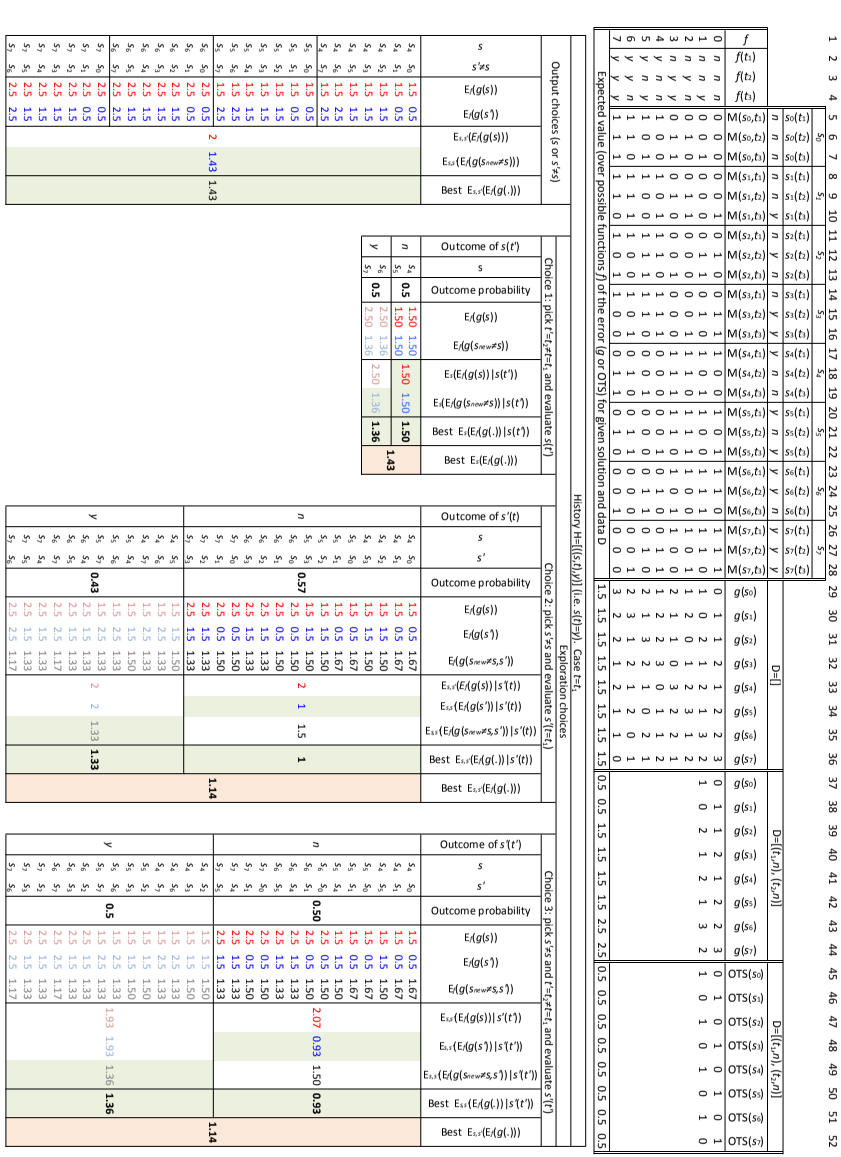
One way to think about it is that supervised learning with an unrestricted set of functions f maps to co-optimization with a restricted set of metrics M; the specifics of this restriction lead to differences in the design of optimal algorithms that we point out throughout the entire Section 3. Note also that in supervised learning there is a dependence between the number of tests and the number of theoretically-possible distinct potential solutions, as $|Y^X| = |Y|^{|X|}$, whereas that's typically not the case in co-optimization (for instance, in the piping network problem, the set S of possible valve placements would be the same whether tests consist of pairs of co-occurring damages or triples thereof, the latter of which are more numerous).

Output mechanisms. If we have no data (D = {}), then all these metrics (functions) are possible and, we assume, equally likely. We discuss this assumption in more detail in Section 3.4. Columns 29–36 in the table show, for each metric, the actual value of g (i.e., the sum of errors over all tests) for each of the 8 potential solutions. At the bottom, the expected value of g, averaged over all metrics, shows that in the absence of information all potential solutions are equally promising, with an expected g-value of 1.5.

Consider being given 2 data points, namely D = $\{(t_1, n), (t_2, n)\}$. Columns 37–44 in Table 5 show that only 2 out of the 8 functions f (metrics M) are still possible (consistent with the data D). Averaging the expected value of g over only these functions shows that potential solutions s_0 and s_1 look more promising than the rest. So any algorithm that would output one of them, by whatever means, would be better overall than an algorithm outputting one of the other six potential solutions. This constitutes *free lunch with respect to g*. By contrast, the *no free lunch* that Wolpert (1992) showed was *for off-training-set (OTS) error*. This is exemplified in columns 45–52 of Table 5, where OTS(s_i) is simply equal to the error over the remaining test t_3, which is $M(s_i, t_3)$. Averaging this over the two possible functions shows that all potential solutions have the same expected error outside of the sample D.

Nonetheless, one question is: how might we exploit the free lunch with respect to g? Answering this for the equal-likelihood case can provide a base from which to then study non-uniform distributions over problems, where there should be free lunch with respect to OTS as well. We can approach this question in a manner similar to co-optimization by considering only algorithms that use an exploratory approach, where they can sample

Table 5: Examples of binary classification free lunch in domain with $X = \{t_1, t_2, t_3\}$, $Y = \{y, n\}$. Font style indicates which values and probabilities were aggregated, green cell shading indicates where the minimum value is coming from and orange cell shading highlights the expected performance for each exploration choice.



$s \in S$ and $t \in T$ that are present in the data and pay for each $M(s,t)$ evaluation. While traditional exploratory approaches to supervised learning always perform *full evaluation*, i.e., for each s considered they evaluate $M(s,t)$ for all t in the data, there is no constraint to do so. In fact, as we shall see later in this section, full evaluation can be suboptimal.

Consider the history $\mathrm{H} = [\langle\langle s,t\rangle, y\rangle]$, meaning the algorithm randomly selected an $s \in S$ and a $t \in \{t_1, t_2\}$, evaluated $s(t)$ and observed value y (spent budget is 1). The output mechanism has two choices: output the partially-evaluated s or output a randomly selected, completely-unevaluated potential solution $s' \neq s$. These choices are shown in the bottom left part of Table 5 (instantiated only with with $t = t_1$, due to space constraints; the other instantiation yields the same kind of results). Since $s(t) = y$, s can only be one of s_4, s_5, s_6, s_7, with respective expected g-values of 1.5, 1.5, 2.5, 2.5. For each possible s, there are seven possibilities for $s' \neq s$, of which four (s_0, s_1, s_2, s_3) have expected g less than or equal to 1.5 and three $(\{s_4, s_5, s_6, s_7\} \setminus \{s\})$ have expected g greater than or equal to 1.5. Averaging over all situations yields an overall expectation of 2 for $g(s)$ and ≈ 1.43 for $g(s')$, thus outputting s' is the strictly better choice and we have *output mechanism free lunch*.

Exploration mechanisms. Suppose now that the algorithm having produced history $\mathrm{H} = [\langle\langle s,t\rangle, y\rangle]$ (spent budget of 1) is allowed one additional M-evaluation (remaining budget of 1). It is the job of the exploration mechanism to decide which interaction to run through the metric M. The rest of Table 5 outlines the three exploration choices: 1) evaluate s again with the remaining test present in the data (in this case, $t' = t_2$); 2) pick a random $s' \neq s$ and evaluate it with the test already seen by s (that is $t = t_1$); and 3) pick a random $s' \neq s$ and evaluate it with the other test ($t' = t_2$). And to determine the expected performance for a given exploration choice, we must: consider each of the two potential outcomes, y or n, that could be observed when evaluating the chosen interaction; determine the best (minimum) g-value estimate for the history resulting from that outcome; then aggregate the estimates for the two outcomes. This aggregate is a *weighted* average, since the two outcomes might not be equally likely; for instance, if we choose to evaluate $s'(t = t_1)$, there are 16 cases with outcome n and only 12 with outcome y, so the respective probabilities are $16/28 \approx 0.57$ and $12/28 \approx 0.43$. The calculations reveal that evaluating an interaction involving s', whether with t or t', yields a strictly better expected g-value (1.14) than further evaluating s (1.43). This constitutes *exploration mechanism free lunch*.

It is insightful to look more closely at these calculations so as to understand how the estimates for the various potential solutions involved change when a new interaction is evaluated. Recall that after having evaluated $s(t)$ and observing y the expected $g(s)$ is 2 and the expected $g(s')$ for any completely-unevaluated potential solution $s' \neq s$ is 1.43. The outcome of evaluating $s(t')$ changes the estimate for s: if we observe $s(t') = n$, the new estimate goes down to 1.5 (we've just observed a match between s and f, so s looks more promising now), whereas if we observe $s(t') = y$, then the estimate goes up to 2.5. This is not surprising, since by evaluating $s(t')$ we are, after all, learning more about s. What is interesting is that the estimate of g for a completely-unevaluated potential solution also changes: either to 1.5 if $s(t') = n$ or to 1.36 if $s(t') = y$. This is pointedly different from generic co-optimization. Referring back to Table 4, for history $\mathrm{H} = [\langle\langle a, t_1\rangle, 1\rangle]$ the expected g-value of b or c is 1 and it does not change when we evaluate $M(a, t_2)$ no matter what outcome we observe. Similar impacts can be noted in Table 5 for the other two exploration choices: evaluating $s'(t)$ for $s' \neq s$ changes the expected value for a random

remaining completely-unevaluated potential solution, but not for s, which has already seen t; evaluating $s'(t')$ for $s' \neq s, t' \neq t$ changes both expectations. In contrast, for co-optimization solution concepts like maxisum and maximin the outcome of evaluations involving one potential solution never changes the expectation for any other potential solutions (and this is independent of which test is used in the evaluation).

This difference can be explained through the prism of the possible metrics M, which in the case of supervised learning are much fewer than $|V|^{|S|\cdot|T|}$. Note for instance how in Table 5 we have no metric M assigning 0 to all interactions, nor one assigning 1 to all interactions. In fact, each of the metrics induced by possible functions f has the property that half the interactions get assigned a 0 and half get assigned a 1; and this is also true at the test level: for any test $t \in T$, there are $|S|/2$ potential solutions $s \in S$ such that $M(s,t) = 0$ and just as many for which $M(s,t) = 1$. Thus for any given test t, the expected value of M for a random potential solution that hasn't seen t yet depends on how many other potential solutions have seen t and what the outcomes were. We provide a generic characterization of this in the following section.

3.2 Large Spent Budget, Small Remaining Budget

When the budget spent so far consists of more than a couple of M-evaluations, the structure of the resulting histories can become more complex in terms of which potential solutions have interacted with which tests. We extend the treatment from the small examples above to these more generic cases; when discussing exploration mechanisms we still keep the remaining budget at just one M-evaluation.

3.2.1 Co-optimization

The issue of free lunch and optimality for the maxisum solution concept of co-optimization has first been investigated by Popovici and De Jong (2009), for output mechanisms only. As shown there and alluded to in the simple example of Section 3.1.1, given a history of evaluated interactions H, the expected value of g for a potential solution s is obtained by adding two components: 1) the sum of actual $M(s,t)$ values over those tests t seen by s as part of H; and 2) the sum of *expected* $M(s,t)$ values over tests t not seen by s. Due to the assumption of equally-likely metrics, the later is simply the product of the number of unseen tests and the average of V. To express this formally, assume T and V are finite, $|T| = m$, $T = \{t_1, \ldots, t_m\}$, $|V| = p$, $V = \{v_1, \ldots, v_p\}$. Given a history H and a potential solution s, let $0 \leq k \leq m$ denote the number of *distinct* interactions in H involving s: $\langle\langle s, t_{i_1}\rangle, v_{j_1}\rangle, \ldots \langle\langle s, t_{i_k}\rangle, v_{j_k}\rangle$, where $i_1, \ldots, i_k \in 1..m$, all distinct, and $j_1, \ldots, j_k \in 1..p$, $v_{j_\ell} = M(s, t_{i_\ell})$. Let the *current sum* of s given H, $g_{\mathrm{H}}(s) = v_{j_1} + \cdots + v_{j_k}$. Then the expected g-value of s given H is $\mathbb{E}(g(s)|\,\mathrm{H}) = g_{\mathrm{H}}(s) + (m-k)\cdot \mathrm{avg}(V)$. For brevity, we sometimes leave H implicit and simply write $\mathbb{E}(g(s))$. Notably, this value does not depend on any interactions involving potential solutions other than s. It also does not depend on which specific distinct k tests s has interacted with as part of H, only how many and what the outcomes of those interactions were. For a potential solution that is completely unevaluated (i.e., $k = 0$), the expectation is simply $m \cdot avg(V)$.

Output mechanisms are optimal if for any given H they output the potential solution s with the best $\mathbb{E}(g(s)|\,\mathrm{H})$ value; this may be a partially-evaluated potential solution ($k > 0$)

or a completely-unevaluated one. The computational cost of this, alluded to in Section 2.3.1, is small[13] : updating $\mathbb{E}(g(s))$ after each new M-evaluation involving s is trivial, as is determining the potential solution with the best $\mathbb{E}(g(\cdot))$-value; the cost of optimal output-selection is at most $O(|\,\mathrm{H}\,|)$. But the feasibility of implementation is dependent on knowing $\mathrm{avg}(V)$. Note that we do not need to know the actual value of m, since the the term $m \cdot \mathrm{avg}(V)$ is common to all estimates and thus doesn't impact the outcome of comparisons between them.

Exploration mechanisms have the nature of choices available to them influenced by the above-mentioned lack of dependencies, as follows. To assess an interaction choice, for each possible outcome in V we assess the resulting history and determine the potential solution with the best g-value estimate for that history, using the $\mathbb{E}(g(s)|\,\mathrm{H})$ formula in the previous paragraph; then we aggregate these values over the possible outcomes; under the assumption of all possible and equally-probable metrics, all outcomes for a given interaction are also equally likely, so the aggregation is via uniform averaging. These mathematical derivations reveal that for any interaction $\langle s,t\rangle$ that is new with respect to history H, the expected performance after evaluating $\langle s,t\rangle$ is independent of the specific t used, as long as s has not previously seen it (Popovici, 2017). Thus the only actual choice to be made is which potential solution to evaluate; the test can be picked randomly amongst those yet unseen by that solution (can be one seen by other potential solutions). For a history containing measured interactions involving α distinct potential solutions, the number of choices is $\alpha+1$: one of α partially-evaluated potential solutions or a random completely-unevaluated one. And the computational cost of assessing one choice is the sum of p optimal output mechanism costs (as described in the previous paragraph), one for each possible outcome in V. This is considerably more expensive than the optimal output mechanism, though explicitly implementing it could still be tractable for small values of p and α. Furthermore, such explicit implementation might not be necessary in some situations: for domains where V is symmetric around 0, if the history H does not contain any fully-evaluated potential solutions (ones that have seen all m tests), then the best choice is mathematically-provable to be evaluating a new interaction for a still-testable potential solution with the best $\mathbb{E}(g(\cdot))$ (Popovici, 2017)—and the ensuing exploration-decision cost is as small as described above for the optimal output mechanism.

3.2.2 Supervised Learning

Let us now return to the case of binary classification. Given a history of evaluated interactions H, the expected value of g for a potential solution s is still obtained by adding the same two components as in co-optimization: the sum of $M(s,t)$ values known from the history ($g_{\mathrm{H}}(s)$) and the sum of expected $M(s,t)$ values. The difference is that for the second component, the expected value of $M(s,t)$ varies with t.

To see this, consider the example in Table 6. For a binary classification domain with $|X| = m$, assuming a one-to-one relationship between S and Y^X, the total number of potential solutions is 2^m. For any test and any metric M, half of these potential solutions will produce an M-value of 1, while the other half will produce an M-value of 0. As part of the history H, test t has interacted with potential solutions s and s', both times resulting

13. For other co-optimization solution concepts, like maximin, output-selection cost can be much larger.

	M(,.)									
	t	t'	t''							
s	1			E($g(s)$)	=	M(s,t)	+	E(t')	+	E(t'')	+	(m-3)*0.5
s'	1	0		E($g(s')$)	=	M(s',t)	+	M(s',t')	+	E(t'')	+	(m-3)*0.5
s''			1	E($g(s'')$)	=	E(t)	+	E(t')	+	M(s'',t'')	+	(m-3)*0.5
s_{new}				E($g(s_{new})$)	=	E(t)	+	E(t')	+	E(t'')	+	(m-3)*0.5
# 1s seen (β_1)	2	0	1									
# pot. sols. seen (β)	2	1	1									
E(.) = $\frac{2^{m-1}-\beta 1}{2^m-\beta}$	$\frac{2^{m-1}-2}{2^m-2}$	$\frac{2^{m-1}}{2^m-1}$	$\frac{2^{m-1}-1}{2^m-1}$									

Table 6: Examples of computing expected g-values for potential solutions for a given history. Binary classification domain with $|X| = m > 4$, $|Y| = 2$ and $3 \leq |D| < m$, with $\mathrm{D} \supseteq \{(t,n),(t',n),(t'',y)\}$ and history $\mathrm{H} = [\langle\langle s,t\rangle,y\rangle, \langle\langle s',t\rangle,y\rangle, \langle\langle s',t'\rangle,n\rangle, \langle\langle s'',t''\rangle,n\rangle]$. $\mathbb{E}(\cdot)$ is shorthand notation for $\mathbb{E}(\cdot|\,\mathrm{D},\mathrm{H})$, whether applied to tests or to g-values for potential solutions.

in value 1. Thus of the remaining 2^m-2 potential solutions that have not been evaluated with t yet, there are only $2^{m-1}-2$ left that can produce a value of 1, while the other 2^{m-1} give a value of 0. So the expected value of $M(\cdot,t)$ for any potential solution other than s and s' is $\mathbb{E}(t|\,\mathrm{D},\mathrm{H}) = \mathbb{E}(t) = \frac{(2^{m-1}-2)\cdot 1+2^{m-1}\cdot 0}{2^m-2} = \frac{2^{m-1}-2}{2^m-2}$. In general, if as part of H a test has interacted with β other potential solutions, of which β_1 produced an outcome of 1, then the probability of observing 1 upon evaluating another potential solution with this test is $\frac{2^{m-1}-\beta_1}{2^m-\beta}$ and the probability of observing 0 is $\frac{2^{m-1}-(\beta-\beta_1)}{2^m-\beta}$; so the expected value for evaluating a new interaction with that test is $\frac{2^{m-1}-\beta_1}{2^m-\beta}\cdot 1 + \frac{2^{m-1}-(\beta-\beta_1)}{2^m-\beta}\cdot 0 = \frac{2^{m-1}-\beta_1}{2^m-\beta}$. $\mathbb{E}(t')$ and $\mathbb{E}(t'')$ in Table 6 were also derived using this formula. The expected value for any test not seen by any potential solution is still $\mathrm{avg}(V) = 0.5$, which can also be derived as $\frac{2^{m-1}-0}{2^m-0}$. This is regardless of whether such a test is part of the data D or not. Then the g-value estimates for potential solutions are $\mathbb{E}(g(s)|\,\mathrm{D},\mathrm{H}) = \mathbb{E}(g(s)) = M(s,t)+\mathbb{E}(t')+\mathbb{E}(t'')+(m-3)\cdot 0.5$ and respectively $\mathbb{E}(g(s'))$, $\mathbb{E}(g(s''))$ and $\mathbb{E}(g(s_{new}))$ shown in Table 6. More thorough mathematical derivations of these quantities can be found in Online Appendix A.2.

Output mechanisms are optimal if they output the potential solution with minimum $\mathbb{E}(g(\cdot))$. To implement that, the estimates are inexpensive to compute, albeit more involved than for co-optimization: for each new $M(s,t)$ evaluation, first update the expected value for $\mathbb{E}(t)$, then update $\mathbb{E}(g(s))$ as well as $\mathbb{E}(g(\cdot))$ for any other potential solutions that haven't yet seen t (including a representative completely-unevaluated one) then re-determine the potential solution with the best g-value estimate, for a total output-selection cost of $O(|\,\mathrm{H}\,|)$. However, these computations do require knowing the value of m and care must be taken to avoid potential computer precision errors for large m.

Exploration mechanisms are exemplified in Table 7, which shows how Table 5's performance estimates for the various exploration choices can be re-derived using a more compact computation approach: for a given choice, for each of the two possible outcomes, consider the resulting history and compute $\mathbb{E}(g(\cdot))$ for all potential solutions involved, using

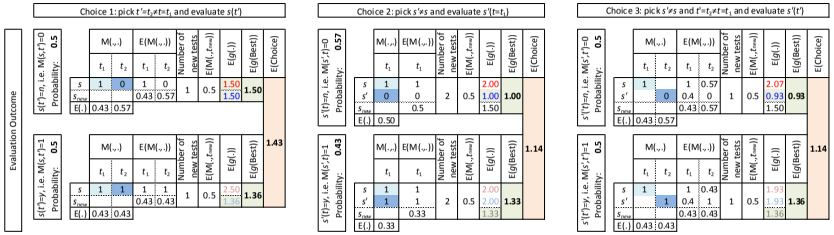


Table 7: Examples of computing expected g-values for potential solutions after various exploration choices and outcomes. Binary classification domain with $X = \{t_1, t_2, t_3\}$, $Y = \{y, n\}$, data $\mathrm{D} = \{(t_1, n), (t_2, n)\}$ and history $\mathrm{H} = [\langle\langle s, t_1\rangle, y\rangle]$. Font style and green & orange cell shading link to the respective cells in Table 5. Light blue shading corresponds to the current history and dark blue shading shows the next interaction chosen for evaluation and its M-outcome. $\mathbb{E}(\cdot)$ is shorthand notation for $\mathbb{E}(\cdot \,|\, \mathrm{D}, \mathrm{H})$. For a given s and t, the expected value $\mathbb{E}(M(s,t))$ is equal either to the actual value of $M(s,t)$, if it has been evaluated, or to $\mathbb{E}(t)$ otherwise.

formulas like the ones in Table 6; then take the best (minimum) estimate; then use the probabilities of the two outcomes to compute a weighted average of the best estimates.

For example, consider the middle panel of Table 7, where $m = 3$ and we choose to evaluate a new potential solution s' with the test $t = t_1$, which has previously interacted only with s (so $\beta = 1$) and produced a value $M(s,t) = 1$ (so $\beta_1 = 1$). The probability of outcome 1 for $M(s', t_1)$ is $\frac{2^2-1}{2^3-1} = \frac{3}{7} \approx 0.43$ and the probability of outcome 0 is $\frac{2^2-0}{2^3-1} = \frac{4}{7} \approx 0.57$. For outcome 1 the best g-value estimate is 1.33 for s_{new} and for outcome 0 it is 1 for s', resulting in an expected performance of $1.14 = 1.33 \cdot 0.43 + 1 \cdot 0.57$ for the $\langle s', t_1\rangle$ exploration choice.

For this particular example, the expected performance for evaluating a new potential solution s' was the same whether using the test t_1 already seen by s or the new test t_2. This is not the case in general, as exemplified in Table 8 for a slightly larger domain and history. When choosing to evaluate a new potential solution s'', we get different expected performance whether we use the test t previously seen by s and s', the test t' previously seen by s' only or a new test t'' not previously seen by any potential solution. Similarly, if choosing to further evaluate s, we get different expected performance whether we use the test t' previously seen by s' or a new test t''.

Thus for binary classification the number of relevant choices available for the exploration mechanism for a given history is considerably larger than in co-optimization. If a history H contains η distinct interactions involving α distinct potential solutions and γ distinct tests ($\eta \leq \alpha \cdot \gamma$), then there could be as many as $(\alpha+1) \cdot (\gamma+1) - \eta$ relevant choices, as opposed to at most $\alpha + 1$ in co-optimization. The computational cost of assessing one choice is the sum of two optimal output mechanism costs, one for each of the possible outcomes $\{y, n\}$. Consequently, the total exploration-decision cost is significantly larger than the optimal output mechanism cost, though still polynomial with a naive explicit implementation of $\mathrm{O}(|\,\mathrm{H}\,|^3)$, which could be tractable for moderate-size histories.

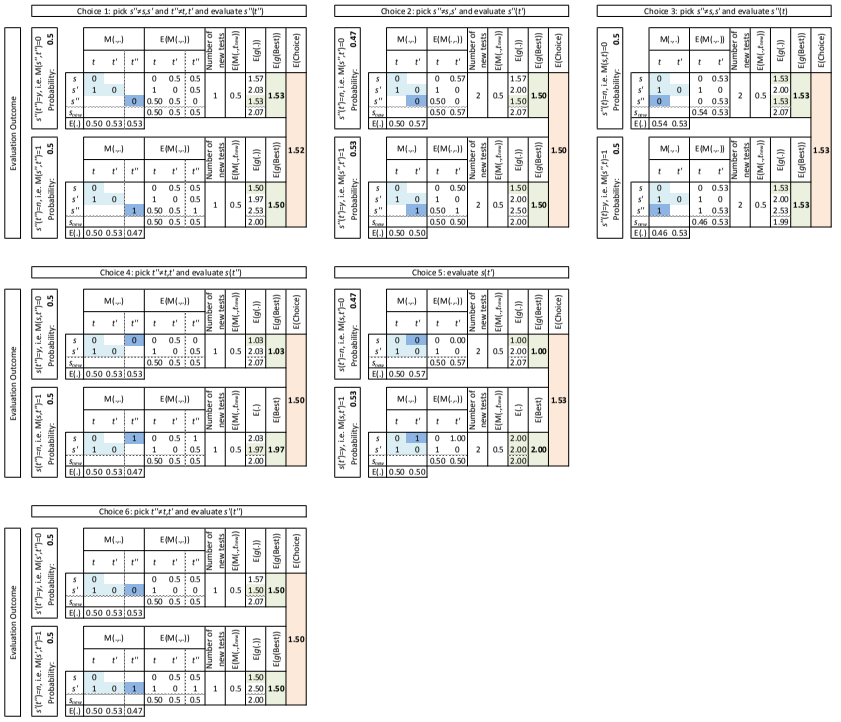


Table 8: Examples of how the choice of test impacts the expected resulting performance. Binary classification domain with $|X| = m = 4$, $Y = \{y, n\}$, data $\mathrm{D} = \{(t, n), (t', n), (t'', y)\}$ and history $\mathrm{H} = [\langle\langle s, t\rangle, n\rangle, \langle\langle s', t\rangle, y\rangle, \langle\langle s', t'\rangle, n\rangle]$. Bold font shows which values and probabilities were aggregated. Green cell shading indicates where the best (minimum) value is coming from, orange cell shading highlights the expected performance for each exploration choice, light blue shading corresponds to the current history and dark blue shading shows the next interaction chosen for evaluation and its M-outcome. $\mathbb{E}(\cdot)$ is shorthand notation for $\mathbb{E}(\cdot \,|\, \mathrm{D}, \mathrm{H})$. For a given s and t, the expected value $\mathbb{E}(M(s,t))$ is equal either to the actual value of $M(s,t)$, if it has been evaluated, or to $\mathbb{E}(t)$ otherwise. $\mathbb{E}(g(s))$ is equal to the sum of the $\mathbb{E}(M(s,t))$ over tests seen by any potential solution plus the number of new tests times $\mathbb{E}(M(\cdot, t_{new})) = \mathbb{E}(t_{new}) = 0.5$.

Since math-based reductions of exploration-decision costs have been proven for co-optimization, we did a preliminary analysis to see if the same were possible for binary classification. We were able to prove a couple of ways to analytically prune the set of relevant exploration choices while maintaining optimality. Specifically, if t_{new} is a test present in the data but previously unseen by any potential solution, then out of the exploration choices $\langle s, t_{new}\rangle, s \in S$ we only need to assess the expected performance for $\langle s^{best}, t_{new}\rangle$, where s^{best} is the potential solution with the best expectation $\mathbb{E}(g(\cdot))$ for the current H (Theorem 1 in Appendix A.3). Moreover, if s^{best} is a completely-unevaluated potential solution, then out of all choices of type $\langle s^{best}, t\rangle, t \in T$ we only need to assess $\langle s^{best}, t^*\rangle$ for a t^* easily-identifiable as maximizing the quantity $\mathbb{E}(t) \cdot \big(1 - \mathbb{E}(t)\big) \cdot \frac{2^{m-\beta}}{2^{m-\beta}-1}$, where β is as in Section 3.2.2, the number of potential solutions seen by t (Theorem 2 in Appendix A.3).

While helpful, these results allow us to prune at most $\alpha + \gamma$ exploration choices, which in the worst case (if $\alpha \cdot \gamma \gg \eta$) could still leave us with $\mathrm{O}(|\,\mathrm{H}\,|^2)$ interaction choices to be assessed, each costing $\mathrm{O}(|\,\mathrm{H}\,|)$. Further research is needed to determine if additional analytical pruning can be performed. Moreover, these results only concern the interaction choice optimizing the expected g-value for the best output right after evaluating that interaction, i.e., a remaining budget of $n = 1$. Which interaction choice is optimal could vary with n and we briefly discuss this case in the next section; the pruning results above were obtained by first generalizing to the formulas below and then instantiating them with $n = 1$.

3.3 Large Remaining Budget

Since output mechanisms are not impacted by remaining budgets and we have already studied large spent budgets in Section 3.2, in this section we focus on exploration mechanisms. Their optimality and free lunch for large remaining budgets was previously investigated in co-optimization for the maxisum (Popovici, 2017) and maximin (Popovici and Winston, 2015) solutions concepts. Following the reasoning from those works, exemplified for $n = 1$ in the previous section, the expected performance for an optimal exploratory algorithm for binary classification, given data D, current history H and remaining budget n, denoted by $\Phi_{\mathrm{a}}^{\min}(\mathrm{D}, \mathrm{H}, n)$, can be recursively expressed as:

$$\Phi_{\mathrm{a}}^{\min}(\mathrm{D}, \mathrm{H}, 0) = \min_{s \in S} \mathbb{E}(g(s)|\,\mathrm{D}, \mathrm{H}),$$

$$\begin{aligned}\Phi_{\mathrm{a}}^{\min}(\mathrm{D}, \mathrm{H}, n) &= \min_{\langle s,t\rangle \in \mathcal{E}'(\mathrm{D},\mathrm{H})} \mathop{\mathbb{E}}_{v \in \{0,1\}} \Phi_{\mathrm{a}}^{\min}(\mathrm{D}, \mathrm{H} \oplus \langle\langle s, t\rangle, v\rangle, n-1) \\ &= \min_{\langle s,t\rangle \in \mathcal{E}'(\mathrm{D},\mathrm{H})} \sum_{v \in \{0,1\}} P(M(s,t) = v|\,\mathrm{D}, \mathrm{H}) \cdot \Phi_{\mathrm{a}}^{\min}(\mathrm{D}, \mathrm{H} \oplus \langle\langle s, t\rangle, v\rangle, n-1)\end{aligned}$$

where

- $\mathcal{E}'(\mathrm{D}, \mathrm{H})$ represents the set of all new interactions that could be evaluated next;
- the probabilities $P(M(s,t) = v|\,\mathrm{D}, \mathrm{H})$ of outcomes 0 and 1 are as described at the beginning of Section 3.2.2;
- $\oplus$ denotes adding a measured interaction to a history; and
- the subscript 'a' refers to 'algorithms'.

In words, the optimal performance for a given history and budget is dependent on the optimal performance for histories lengthened by one and budgets decreased by one; it is obtained by optimizing over interaction choices the expected value, aggregated over possible outcomes for the interaction, of the optimal performance for the resulting lengthened history and reduced budget.

Going back to the expression of $\mathbb{E}(g(s))$ from Table 6, note that it can be re-written as $M(s,t) + \mathbb{E}(t') + \mathbb{E}(t'') + (|D| - 3) \cdot 0.5) + (m - |D|) \cdot 0.5$. The last term is the expected out-of-sample (OTS) error and is the same for all potential solutions: this is the no free lunch of Wolpert (1992). The sum of the other terms represents the *expected* in-sample error. Note this differs from traditional exploratory approaches to supervised learning, where each considered potential solution is evaluated with all tests in the data and therefore the actual in-sample error is known. Such an approach can still be used if the budget n is large in relation to the size of the data D, but given the existence of exploration mechanism free lunch, this could conceivably be a suboptimal approach.

To show that this can indeed happen, we apply the above recursive formula to a small example. Consider a domain with 3 inputs ($|X| = m = 3$) and a situation where we are given 2 data points ($|\,\mathrm{D}\,| = 2$) and a budget of 2 M-evaluations. Whenever output needs to be produced, we use the optimal output mechanism, which chooses the potential solution with minimum $\mathbb{E}(g(\cdot))$ given the history at that point. The optimal exploration mechanism takes into account the outcome of the first interaction in order to determine what the second interaction should be; the respective aggregate performance, given by the above formula, is ≈ 1.071. Full evaluation simply evaluates a potential solution with the two available tests and there are 4 equally-likely combinations of outcomes (11, 01, 10, 00); averaging the respective minimum $\mathbb{E}(g(\cdot))$ values (1.357, 1.5, 1.5, 0.5) yields an aggregate performance of ≈ 1.214, which is strictly worse.

However, implementing the optimal exploration mechanism appears daunting for large budgets n, as it seemingly involves solving a very time-consuming recursive optimization problem. Interestingly, the co-optimization studies of optimal exploration mechanisms for maxisum (Popovici, 2017) and maximin (Popovici and Winston, 2015) showed that for some situations an analytical solution to this recursive optimization can be proven, making the optimal algorithm straightforward to implement, with low exploration-decision and output-selection costs of $\mathrm{O}(|\,\mathrm{H}\,|)$.

The preliminary analysis we performed for binary classification so far only yielded the pruning results for $n = 1$ presented in the previous section, which could still leave us with more than one choice, requiring recursing down to $n = 0$. This is not an issue for such a small n, but for large n the exponential cost of recursive optimization can be infeasible. The additional complexity introduced in binary classification by the interdependence between different potential solutions makes it unclear at the moment whether a full analytical solution for the recursive optimization (or at least some amount of pruning) can be proven for large remaining budgets—and thus whether the costs of the optimal exploration mechanism are feasible or not. This serves to highlight the importance of acknowledging *all* types of costs, as optimizing with respect to one type (e.g., metric-evaluation costs) can incur another (e.g., exploration-decision costs).

3.4 Discussion

The above analysis made two core assumptions: 1) that all functions in Y^X are equally likely; and 2) that there is a of one-to-one mapping between S and Y^X. Additionally, algorithms were constrained to use only a limited budget of M-evaluations; such algorithms might not know the actual in-sample error (error on the data D) for all/any of the potential solutions considered and thus must decide what to output given only such incomplete information.

For this setup, given a certain budget n of M-evaluations, an optimal algorithm is one that has the smallest average over all functions in Y^X of the error on the data D (in-sample error) for the potential solution outputted after using up the budget; and there is free lunch, in that some algorithms produce strictly smaller such average values than others. Unfortunately, this is not useful for generalization, since under the above assumptions, all potential solutions have the same average over functions of OTS-error (error outside the data D): $(m - |\,\mathrm{D}\,|) \cdot 0.5$.

In practice though, we often have reason to believe that the first assumption is not true, rather that there is a non-uniform P on Y^X. But if we do not know anything more specific about P besides its existence, this in itself is not sufficient reason for dismissing OTS no free lunch, as there are as many distributions over metrics for which one algorithm is aggregately-better than the other as there are distributions where the opposite relation holds (Wolpert, 1992). On the other hand, if the knowledge we have about P and the way we choose S allow us to express theoretical Bayes estimates, then we can perform the same type of analysis as presented here to derive and investigate algorithms that are aggregately-optimal from a generalization perspective as well.

Here are several ways in which this could become useful. First, it could be used in passive supervised learning to provide a new approach for dealing with data sets so large that it is too costly to perform evaluation with their entire contents for every potential solution considered. Rather than performing instance selection (Reinartz, 2002) and then performing full evaluation using the selected subset of examples, computational resources could be used more effectively by selecting at a more granular level, namely that of interactions $\langle s, t\rangle$ rather than that of examples (tests) t.

Second, it could be extended for usage in active learning, as a direct exploratory approach instead of the current methods which perform retraining after new labels are observed. To achieve this, the analysis will have to be adapted to take into account labeling costs; whether a new interaction $\langle s, t\rangle$ will incur both an M-evaluation cost and a labeling cost or only the former depends on whether or not the test t has been previously seen by other potential solutions and thus already labeled. This could be formalized as having two different budgets, one for labeling and one for M-evaluations, or, if both costs are in fact measured in the same "currency", have a single budget and allow the algorithm to determine the best way to allocate it to the two activities.

Last but not least, recall from the discussion on page 11 that some exploratory approaches do not rely on the error function having a closed form with easily computable derivative and therefore they also do not require that the data be in (or transformed into) tabular form. The optimal algorithms described in Section 3 have this desirable property and thus would be applicable to supervised learning for more complex data structures.

4. Conclusions

Metaphorically speaking, this paper laid out a deciphered Rosetta Stone for the languages of supervised learning and co-optimization, then used it to translate free lunch research from the latter field to the former. While practical optimal algorithms based on free lunch are still a ways away, the hope of this presentation is that the different perspective provided by co-optimization could spark ideas for new types of heuristics for supervised learning (promising directions were described in Section 3.4) and also that this translation will become only one of many, the beginning of cross-pollination of ideas between the fields.

Several paths for future research have already been pointed out: reducing the time complexity of optimal exploration mechanisms in the base setup comprising uniformity assumptions, expanding to non-uniform distributions and adapting to the multiple types of costs incurred in active learning. Additionally, there are yet more fields concerned with similar problems and constructing similar mappings for them could lead to further advances; the most obvious ones are reinforcement learning, design of experiments and statistical learning in general.

Acknowledgments

This material is based upon work supported by the Office of Naval Research under Contract No. N00014-12-C-0131. The author would also like to thank Siegfried Martens and the anonymous reviewers for reading earlier versions of the manuscript and providing valuable and constructive feedback.

References

Yaser S. Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. *Learning From Data*. AMLBook, 2012.

Amazon. Amazon Mechanical Turk Documentation, 2005. URL `https://aws.amazon.com/documentation/mturk/`. Accessed on 2016-03-08.

Peter J. Angeline and Jordan B. Pollack. Competitive environments evolve better solutions for complex tasks. In *International Conference on Genetic Algorithms*, pages 264–270. Morgan Kaufmann Publishers Inc., 1993.

Anthony Bucci. *Emergent Geometric Organization and Informative Dimensions in Co-evolutionary Algorithms*. PhD thesis, Michtom School of Computer Science, Brandeis University, USA, 2007.

Anthony Bucci, Jordan B Pollack, and Edwin De Jong. Automated extraction of problem structure. In *Genetic and Evolutionary Computation Conference*, pages 501–512. Springer, 2004.

Jianlin Cheng, Allison N. Tegge, and Pierre Baldi. Machine learning methods for protein structure prediction. *IEEE Reviews in Biomedical Engineering*, 1:41–49, 2008.

Siang Yew Chong, Peter Tino, and Xin Yao. Measuring generalization performance in coevolutionary learning. *IEEE Transactions on Evolutionary Computation*, 12(4):479–505, 2008.

Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, 2000.

Ido Dagan and Sean P. Engelson. Committee-based sampling for training probabilistic classifiers. In *International Conference on Machine Learning*, pages 150–157, 1995.

Edwin D. de Jong. The MaxSolve algorithm for coevolution. In *Genetic and Evolutionary Computation Conference*, pages 483–489. ACM Press, 2005.

Edwin D. de Jong. Objective fitness correlation. In *Genetic and Evolutionary Computation Conference*, pages 440–447. ACM Press, 2007.

Kenneth A. De Jong. *Evolutionary Computation: A Unified Approach*. MIT press, 2006.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. The reusable holdout: preserving validity in adaptive data analysis. *Science*, 349 (6248):636–638, 2015.

Dario Floreano, Peter Dürr, and Claudio Mattiussi. Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1(1):47–62, 2008.

Seymour Geisser. The predictive sample reuse method with applications. *Journal of the American Statistical Association*, 70(350):320–328, 1975.

Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58, 1992.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

W. Daniel Hillis. Co-evolving parasites improve simulated evolution in an optimization procedure. *Physica D*, 42:228–234, 1990.

Tin Kam Ho. Random decision forests. In *Third International Conference on Document Analysis and Recognition*, volume 1, pages 278–282. IEEE, 1995.

Ilknur Icke and Josh C. Bongard. Improving genetic programming based symbolic regression using deterministic machine learning. In *Congress on Evolutionary Computation*, pages 1763–1770. IEEE, 2013.

Mikkel T. Jensen. Finding worst-case flexible schedules using coevolution. In *Genetic and Evolutionary Computation Conference*, pages 1144–1151. Morgan Kaufmann, 2001.

John R Koza, Martin A. Keane, Matthew J. Streeter, William Mydlowec, Jessen Yu, and Guido Lanza. *Genetic Programming IV: Routine Human-Competitive Machine Intelligence*, volume 5 of *Genetic Programming*. Springer Science & Business Media, 2006.

Pier L. Lanzi, Wolfgang Stolzmann, and Stewart W. Wilson. *Learning Classifier Systems: From Foundations to Applications*. Springer, 2003.

Pedro Larranaga and Jose A. Lozano. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, volume 2 of *Genetic Algorithms and Evolutionary Computation*. Springer Science & Business Media, 2002.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553): 436–444, 2015.

Sean Luke. *Essentials of Metaheuristics*. Lulu, second edition, 2013. URL `http://cs.gmu.edu/~sean/book/metaheuristics/`. Available for free.

Subhransu Maji. Large scale image annotations on amazon mechanical turk. *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2011-79*, 2011.

Liviu Panait and Sean Luke. A comparison of two competitive fitness functions. In W. B. Langdon et al, editor, *Genetic and Evolutionary Computation Conference*, pages 503–511. Morgan Kaufmann, 2002.

Elena Popovici. *An Analysis of Two-Population Coevolutionary Computation*. PhD thesis, George Mason University, USA, 2006.

Elena Popovici. Co-optimization free lunches: Tractability of optimal black-box algorithms for maximizing expected utility. *Evolutionary Computation (accepted for publication March 21st)*, 2017.

Elena Popovici and Kenneth A. De Jong. Relationships between internal and external metrics in co-evolution. In *Congress on Evolutionary Computation*, pages 2800–2807. IEEE Press, 2005.

Elena Popovici and Kenneth A. De Jong. Monotonicity versus performance in co-optimization. In *Foundations of Genetic Algorithms*, pages 151–170. ACM Press, 2009.

Elena Popovici and Ezra Winston. A framework for co-optimization algorithm performance and its application to worst-case optimization. *Theoretical Computer Science*, 567:46 – 73, 2015.

Elena Popovici, Oliver Bandte, and Paolo Gaudiano. New approaches for the automated performance evaluation and design of control systems: a firemain example. In *Automation and Controls Symposium*. American Society of Naval Engineers, 2007.

Elena Popovici, Anthony Bucci, R. Paul Wiegand, and Edwin D. de Jong. Coevolutionary principles. In Grzegorz Rozenberg, Thomas Back, and Joost N. Kok, editors, *Handbook of Natural Computing*, Theoretical Computer Science, pages 987–1033. Springer-Verlag, 2010.

Elena Popovici, Ezra Winston, and Anthony Bucci. On the practicality of optimal output mechanisms for co-optimization algorithms. In *Foundations of Genetic Algorithms*, pages 43–60. ACM Press, 2011.

Mitchell A. Potter. *The Design and Analysis of a Computational Model of Cooperative Coevolution*. PhD thesis, George Mason University, USA, 1997.

Payam Refaeilzadeh, Lei Tang, and Huan Liu. Cross-validation. In *Encyclopedia of database systems*, pages 532–538. Springer, 2009.

Thomas Reinartz. A unifying view on instance selection. *Data Mining and Knowledge Discovery*, 6(2):191–210, 2002.

Lior Rokach and Oded Maimon. *Data Mining with Decision Trees: Theory and Applications*. World Scientific, 2014.

Christopher D Rosin and Richard K Belew. New methods for competitive coevolution. *Evolutionary Computation*, 5(1):1–29, 1997.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

Jerome Sacks, William J. Welch, Toby J. Mitchell, and Henry P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4(4):409–423, 1989.

Travis C. Service. Unbiased coevolutionary solution concepts. In *Foundations of Genetic Algorithms*, pages 121–130. ACM Press, 2009a.

Travis C. Service. Free lunches in pareto coevolution. In *Genetic and Evolutionary Computation Conference*, pages 1721–1728. ACM Press, 2009b.

Travis C. Service and Daniel R. Tauritz. Co-optimization algorithms. In *Genetic and Evolutionary Computation Conference*, pages 387–388. ACM, 2008a.

Travis C. Service and Daniel R. Tauritz. A no-free-lunch framework for coevolution. In *Genetic and Evolutionary Computation Conference*, pages 371–378. ACM, 2008b.

Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.

Kent A. Spackman. Signal detection theory: Valuable tools for evaluating inductive learning. In *Sixth International Workshop on Machine Learning*, pages 160–163. Morgan Kaufmann Publishers Inc., 1989. ISBN 1-55860-036-1.

Mervyn Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*, 36(2):111–147, 1974.

David H. Wolpert. On the connection between in-sample testing and generalization error. *Complex Systems*, 6(1):47, 1992.

David H. Wolpert. The lack of a priori distinctions between learning algorithms. *Neural Computation*, 8(7):1341–1390, 1996.

David H. Wolpert and William G. Macready. Coevolutionary free lunches. *IEEE Transactions on Evolutionary Computation*, 9(6):721–735, December 2005.

Journal of Machine Learning Research 18 (2017) 1-5 Submitted 10/16; Revised 4/17; Published 4/17

GFA: Exploratory Analysis of Multiple Data Sources with Group Factor Analysis

Eemeli Leppäaho eemeli.leppaaho@aalto.fi
Muhammad Ammad-ud-din muhammad.ammad-ud-din@aalto.fi
Samuel Kaski samuel.kaski@aalto.fi
Helsinki Institute for Information Technology HIIT
Department of Computer Science
Aalto University
P.O.Box 15400, FI-00076 Aalto, FINLAND

Editor: Mohammad Emtiyaz Khan

Abstract

The R package GFA provides a full pipeline for factor analysis of multiple data sources that are represented as matrices with co-occurring samples. It allows learning dependencies between subsets of the data sources, decomposed into latent factors. The package also implements sparse priors for the factorization, providing interpretable biclusters of the multi-source data.

Keywords: Bayesian latent variable modelling, biclustering, data integration, factor analysis, multi-view learning

1. Introduction

The need for intelligent data integration and analysis methods is getting increasingly common. We often encounter connected measurements from multiple data sources, possessing complex shared variation and structured noise. Modelling the relationships within this kind of a data collection, and using them for prediction, is a challenging problem. Two existing R packages, CCAGFA (Klami et al., 2015) and CMF (Klami et al., 2014), provide good tools for this[1], but remain limited in their use for two reasons: the packages do not provide a complete data analysis pipeline, and they assume the structure present within data sources to be dense, that is, present in all the samples and the features. This assumption is often unrealistic, and we provide package GFA to allow sparse factorization of multi-source data, along with tools from data preprocessing to model interpretation.

Extending from the classical Bayesian matrix factorization work, *group factor analysis* (GFA) (Virtanen et al., 2012; Klami et al., 2015) assumes that data matrix $\mathbf{Y}^{(m)} \in \mathbb{R}^{N \times D_m}$ can be approximated with $\mathbf{X}\mathbf{W}^{(m)\top}$, where the latent variable matrix $\mathbf{X} \in \mathbb{R}^{N \times K}$ describes the observed data in a lower dimensional space, and the projection matrix $\mathbf{W}^{(m)} \in \mathbb{R}^{D_m \times K}$ gives a mapping between the latent space and the data space of the mth source. A key aspect of the model definition is a group sparse prior for the $\mathbf{W}$ matrices: the association of a data matrix to component k is based on evidence in the data. Allowing only a subset of the components to explain a data matrix results in the posterior identifying relationships between any subset of the data sources. The spike-and-slab sparsity priors used to achieve this are described by Bunte et al. (2016). Alternative priors aiming for similar goals have been presented by Zhao et al. (2016)[2], though not containing any tools for preprocessing or model interpretation.

1. Available at `https://cran.r-project.org/package=CCAGFA` and `https://cran.r-project.org/package=CMF` .

2. The GFA Package

GFA implements a Gibbs sampling scheme for inferring the posterior of group factor analysis, and it is available for R at `https://cran.r-project.org/package=GFA`. It has been designed to contain the key elements needed for the data analysis task of factorization of multiple data sources.

2.1 Model Options and Initialization

The package includes a variety of model options for the user to choose from, depending on the application requirement. The default priors assume latent factors that are dense along the sample space, and group sparsity along the feature space (Klami et al., 2015). Using them, the posterior given data `Y` can be inferred by:

```
opts <- getDefaultOpts()
res <- gfa(Y,opts)
```

Another main use of the package is biclustering of multiple data sources, which is done by using element-wise sparse priors for the matrices $\mathbf{X}$ and $\mathbf{W}$. This approach has been presented in Bunte et al. (2016), and is usable via `getDefaultOpts(bicluster=TRUE)`.

The input data `Y` given to function `gfa()` is a list of matrices (that is, data sources) that each have an equal number of rows. Alternatively, if there are several data sources with the same columns, it is possible to input a list with two elements: the sources with paired rows, and the sources with paired columns. Other important options include which parts of the data collection share the scale of each component or the strength of the residual noise (by default each data source for both), and whether the convergence of the sampling chain should be estimated (based on the Geweke diagnostic of its reconstruction). A full description of the possible options is given in the documentation of the function `getDefaultOpts()`.

2.2 Missing Value Prediction

GFA can be used for prediction by simply including the missing matrix elements as `NA`s. The model parameters are sampled based on the observed data only, and the missing values are predicted with the function `reconstruction()`. If the data samples are acquired in a sequential manner, it is alternatively possible to i) infer the models parameters on the first batch, and ii) using fixed projections, quickly predict missing data sources of the second batch based on the observed data sources. Both the approaches use learned relationships between the data sources (shared components) to predict from one to other, and provide predictive distributions along with point estimates.

2. Available at `https://github.com/judyboon/BASS`.

©2017 Eemeli Leppäaho, Muhammad Ammad-ud-din and Samuel Kaski.

License: CC-BY 4.0, see `https://creativecommons.org/licenses/by/4.0/`. Attribution requirements are provided at `http://jmlr.org/papers/v18/16-509.html`.

JMLR 18(39):1-5, 2017

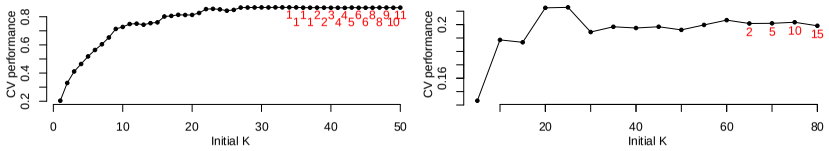


Figure 1: Illustrating model complexity selection with GFA on synthetic data (left) and GDSC data set (right), showing the cross-validated predictive performance (Spearman correlation), and the number of empty components (red numbers) as a function of initial model complexity K. High initial K allows the components to produce a flexible noise model, as opposed to low initial K.

3. Usage

In addition to inferring model parameters and predicting missing values, the data analysis pipeline typically needs additional steps presented here.

3.1 Preprocessing

The input data can be normalized in different ways with `normalizeData(Y)`. As GFA aims to explain maximal variance with a compact set of parameters, and has no separate assumptions for the bias (mean) of features, centering the data is advised. Furthermore, the higher the variance of features (or data sources), the larger their effect to the model parameters. If the user considers this to be desired behavior, no scaling should be done. On the other hand, if the features (or data sources) are deemed approximately equally important, they should be scaled accordingly.

3.2 Model Complexity Selection

The priors of GFA are able to infer the model complexity. This is achieved by initializing the model with a high number of components K (default $\min(N, \sum_m D_m)/2$) and allowing the sparsity prior to prune out excessive components (explaining only little variance). As complexity selection is a difficult task in general, we recommend using any prior information of the data, if available, when selecting the initial K. Here it should be noted that the components can model (structured) noise in addition to the signal of interest, inducing a flexible noise model. If this is not desired, `informativeNoisePrior()` can be used to specify how large proportion of variance is expected to be explained with the components; in effect causing weaker (structured noise) components to be explained by the simple residual noise. The package issues a warning if the automatic model complexity selection does not seem to work as intended, for example if no components are pruned.

We demonstrate model complexity selection with two examples in Figure 1: a synthetic data set, and a cancer cell line data set by Genomics of Drug Sensitivity in Cancer project (GDSC) (Yang et al., 2013). The complexity of the synthetic data set is overestimated slightly, resulting in accurate predictions as long as initial K is above the true $K = 32$. With the GDSC data set, where the true model complexity is unknown, initial K at 20 or 25 gives the best predictions, but the full model is not far behind.

3.3 Interpreting the Model

The inferred factorization explains dependencies in the data collection through K components. The factorization can be visualized using function `visualizeComponents()`, which shows the component activities in the data sources. Shared components imply predictability from a data source to another. The function can also show the full parameter matrices $\mathbf{X}$ and $\mathbf{W}$, as well as their effect, and the effect of individual components, in the data space.

3.4 Robustness Analysis

As in factor analysis in general, the factorization GFA aims to infer is unidentifiable, resulting in an extremely multi-modal posterior.[3] Thus consideration is needed if a user wishes to analyze multiple factorizations (in this case sampling chains) jointly, as they are not likely to be identical. To the best of our knowledge, there are no sampling methods that would in practice converge to the true multi-modal posterior with high-dimensional data. To this end, we provide function `robustComponents()` which uses a correlation-based procedure to analyze which components occur frequently in different GFA sampling chains. This approach can be used to obtain robust factorizations for noisy data collections, where the components explain only small parts of the variation, resulting in uncertain estimates.

3.5 Examples

The package manual contains simplified examples demonstrating function usage. Besides this, we provide the following three demonstrations:

- demo(GFApipeline): Simple illustration of the GFA pipeline.
- demo(GFAexample): Elaborate illustration of the GFA model on simulated data.
- demo(GFAdream): Replication of Bunte et al. (2016) results (requiring data download, which is instructed).

4. Discussion

GFA allows inferring the relationships between multiple co-occurring data matrices by reconstructing them with group sparse priors for the projection matrices. It finds a decomposition with components that are specific to the data sources, shared between them all, or shared between any subset of them. This kind of sparsity is beneficial for exploratory data analysis and integration. Our GFA package covers essential tools ranging from preprocessing to modelling assumptions and from robustness analysis to interpreting the model.

Acknowledgments

This work was financially supported by the Academy of Finland (Finnish Center of Excellence in Computational Inference Research COIN; grants 295503 and 292337 to MA and SK). We acknowledge the computational resources provided by Aalto Science-IT project.

3. Unimodal posterior could be achieved in special cases, given limiting identifiability constraints.

References

Kerstin Bunte, Eemeli Leppäaho, Inka Saarinen, and Samuel Kaski. Sparse group factor analysis for biclustering of multiple data sources. *Bioinformatics*, 32(16):2457–2463, 2016. Preprint: `http://arxiv.org/abs/1512.08808`.

Arto Klami, Guillaume Bouchard, and Abhishek Tripathi. Group-sparse embeddings in collective matrix factorization. In *International Conference on Learning Representations*, 2014.

Arto Klami, Seppo Virtanen, Eemeli Leppäaho, and Samuel Kaski. Group factor analysis. *IEEE Transactions on Neural Networks and Learning Systems*, 26(9):2136–2147, 2015.

Seppo Virtanen, Arto Klami, Suleiman A. Khan, and Samuel Kaski. Bayesian group factor analysis. In Neil Lawrence and Mark Girolami, editors, *Fifteenth International Conference on Artificial Intelligence and Statistics*, volume 22 of *JMLR W&CP*, pages 1269–1277. JMLR, 2012.

Wanjuan Yang, Jorge Soares, Patricia Greninger, Elena J Edelman, Howard Lightfoot, Simon Forbes, Nidhi Bindal, Dave Beare, James A Smith, I Richard Thompson, et al. Genomics of drug sensitivity in cancer (GDSC): a resource for therapeutic biomarker discovery in cancer cells. *Nucleic Acids Research*, 41(D1):D955–D961, 2013.

Shiwen Zhao, Chuan Gao, Sayan Mukherjee, and Barbara E Engelhardt. Bayesian group factor analysis with structured sparsity. *Journal of Machine Learning Research*, 17(196): 1–47, 2016.

Journal of Machine Learning Research 18 (2017) 1-6 Submitted 10/16; Revised 2/17; Published 4/17

GPflow: A Gaussian Process Library using TensorFlow

Alexander G. de G. Matthews AM554@CAM.AC.UK
Department of Engineering
University of Cambridge
Cambridge, UK

Mark van der Wilk MV310@CAM.AC.UK
Department of Engineering
University of Cambridge
Cambridge, UK

Tom Nickson TRON@ROBOTS.OX.AC.UK
Department of Engineering Science
University of Oxford
Oxford, UK

Keisuke Fujii FUJII@ME.KYOTO-U.AC.JP
Department of Mechanical Engineering and Science
Graduate School of Engineering, Kyoto University, Japan

Alexis Boukouvalas ALEXIS.BOUKOUVALAS@MANCHESTER.AC.UK
Division of Informatics
Manchester University
Oxford Road, Manchester, UK

Pablo León-Villagrá PABLO.LEON@ED.AC.UK
School of Informatics
University of Edinburgh
10 Crichton Street, Edinburgh, UK

Zoubin Ghahramani ZOUBIN@ENG.CAM.AC.UK
Department of Engineering
University of Cambridge
Cambridge, UK

James Hensman JAMES.HENSMAN@LANCASTER.AC.UK
CHICAS, Faculty of Health and Medicine
Lancaster University
Lancaster, UK

Editor: Alexandre Gramfort

Abstract

GPflow is a Gaussian process library that uses TensorFlow for its core computations and Python for its front end.[1] The distinguishing features of GPflow are that it uses variational inference as the primary approximation method, provides concise code through the use of automatic differentiation, has been engineered with a particular emphasis on software testing and is able to exploit GPU hardware.

1. GPflow and TensorFlow are available as open source software under the Apache 2.0 license.

©2017 Alexander G. de G. Matthews, Mark van der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas, Pablo León-Villagrá, Zoubin Ghahramani, and James Hensman.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-537.html

1. Existing Gaussian Process Libraries

Gaussian processes (GPs) are versatile Bayesian nonparametric models using a prior on functions (Rasmussen and Williams, 2006). There are now many publicly available GP libraries ranging in scale from personal projects to major community tools. We do not here give detailed consideration to all libraries, regrettably neglecting, for instance, those that only support Gaussian likelihoods (Ambikasaran et al., 2014) or do not make provision for scalability (Pedregosa et al., 2011). The influential GPML toolbox (Rasmussen and Nickisch, 2010) uses *MATLAB*. It has been widely forked. A key reference for our particular contribution is the GPy library (GPy, since 2012), which is written primarily using Python and Numeric Python (NumPy). GPy has an intuitive object-oriented interface. Another relevant GP library is GPstuff (Vanhatalo et al., 2013) which is also a *MATLAB* library.

2. Objectives for a new library

GPflow is motivated by a set of goals. The software is designed to be fast, particularly at scale. Where approximation inference is necessary we want it to be accurate. We aim to support a variety of kernel and likelihood functions. Another goal is that the implementations are well tested. The software should be made easy to use by an intuitive user interface. Finally it should be easy to extend the software. We argue that there is a way to better meet these simultaneous objectives than existing packages and that it is realised in GPflow.

3. Key features for meeting the objectives

To best meet the key goals of our library, we were led to a project that had all of the following distinguishing features:

(i) The use of variational inference as the primary approximation method to meet the twin challenges of non-conjugacy and scale (Matthews, 2016).

(ii) Relatively concise code which uses automatic differentiation to remove the burden of gradient implementations.

(iii) The ability to leverage GPU hardware for fast computation.

(iv) A clean object-oriented Python front end.

(v) A dedication to testing and open source software principles.

Table 1 gives a summary of which GP libraries possess the distinguishing features we have highlighted. The GPflow interface and Python architecture are heavily influenced by GPy. An important difference between GPflow and GPy is that GPflow uses TensorFlow for its core computations rather than numeric Python. This difference significantly affects the general requirements of the architecture. The GPU functionality in GPy is currently limited to CUDA code for the GPLVM (Dai et al., 2014). By contrast the GPflow implementation is targeted at a broad variety of GPU capability.

Library	Sparse variational inference	Automatic differentiation	GPU demonstrated	OO Python front end	Test coverage
GPML	✓	✗	✗	✗	N\R
GPstuff	Partial	✗	✗	✗	N\R
GPy	✓	✗	GPLVM	✓	49%
GPflow	✓	✓	SVI	✓	99%

Table 1: A summary of the features possessed by existing GP libraries at the time of writing. OO stands for object-oriented. In the GPU column GPLVM denotes GP latent variable model and SVI is Stochastic variational inference. N\R denotes not reported.

Having established a desirable set of key design features, the question arises as how best to engineer a GP library to achieve them. A central insight here is that many of the features we highlight are well supported in neural network libraries. Of the available libraries we use TensorFlow (Abadi et al.), as discussed in the next section.

4. Contributing GP Requirements to TensorFlow

In TensorFlow (Abadi et al.) a computation is described as a directed graph where the nodes represent *Operations* (*Ops* for short) and the edges represent *Tensors*. As a directed edge, a Tensor represents the flow of some data between computations. Ops are recognisable mathematical functions such as addition, multiplication etc. *Kernels* (in an unfortunate collision in terminology with the GP literature) are implementations of a given Op on a specific device such as a CPU or GPU. Like almost all modern neural network software, TensorFlow comes with the ability to automatically compute the gradient of an objective function with respect to some parameters. The TensorFlow open source implementation comes with a wealth of GPU kernels for the majority of Ops.

Although we gained significantly from using TensorFlow within GPflow, there were some capabilities that were not yet present in the software which were required for our purposes. We therefore added this functionality to TensorFlow. GP software needs the ability to solve systems of linear equations using common linear algebra algorithms. The differentiation of computational graphs that used the Cholesky decomposition required a new Op, which we contributed. The main part of the code was a C++ implementation of the blocked Cholesky algorithm proposed by Murray (2016). We also contributed code that enabled GPU solving of matrix triangular systems, which in some cases is the bottleneck for approximate inference.

5. Details of GPflow

GPflow supports exact inference where possible, as well as a variety of approximation methods. One source of intractability is non-Gaussian likelihoods, so it is helpful to categorize the available likelihood functionality on this basis. Another major source of intractability is the adverse scaling of GP methods with the number of data points. To this end we support 'variationally sparse' methods which ensure that the approximation is scalable and close

	Gaussian likelihood	Non-Gaussian likelihood (variational)	Non-Gaussian likelihood (MCMC)
Full covariance	GPR	VGP	GPMC
Variational sparsity	SGPR	SVGP	SGPMC

Table 2: A table showing the inference classes in GPflow. Relevant references are VGP (Opper and Archambeau, 2009), SGPR (Titsias, 2009), SVGP (Hensman et al., 2013, 2015b) and SGPMC (Hensman et al., 2015a).

in a Kullback-Leibler sense to the posterior (Matthews et al., 2016). Whether or not a given inference method uses variational sparsity is another useful way to categorize it. The inference options, which are implemented as classes in GPflow, are summarized in Table 2. Note that all the MCMC based inference methods support a Bayesian prior on the hyper-parameters, whereas all other methods assume a point estimate. Our main MCMC method is Hamiltonian Monte Carlo (Neal, 2010).

We now discuss some architectural considerations. The whole Python component of GPflow is intrinsically objected-oriented. The code for the various inference methods in Table 2 is structured in a class hierarchy, where common code is pulled out into a shared base class. The object-oriented paradigm, whilst very natural for the Python layer of the code, has a different emphasis to that of a computational graph which is arguably closer to a functional concept. The largely functional computational graph, and a object-oriented interface need to live cleanly together in GPflow. This is achieved through the Param class that allows parameters to be both properties of an object that can be manipulated as such and Variables in a TensorFlow graph that can be optimized.

A number of steps have been taken to ensure project quality and usability. All GPflow source code is openly available on GitHub at `http://github.com/GPflow/GPflow`. The web page uses continuous integration to run an automated test suite. The test code coverage for GPflow is higher than similar packages where the code coverage statistics are published, achieving a level of 99% (Table 1). A user manual can be found at `http://gpflow.readthedocs.io`.

6. Timed Experiments

As a scenario, we studied training a multiclass GP classifier on MNIST using stochastic variational inference (Hensman et al., 2015b,a).[2] We compared against GPy. None of the other libraries discussed support this algorithm. Functionally, the algorithms are nearly identical in GPflow and GPy. We did a series of trials measuring the time each package took to perform 50 iterations of the algorithm. The trials included a set of CPU experiments, where we varied the number of threads available to the two packages. For GPflow, we also measured the effect of adding a GPU on top of the maximal number of CPU threads considered. GPy does not presently have a GPU implementation of this algorithm. We

2. Code for these timing experiments can be found at `http://github.com/gpflow/GPflowBenchmarks`

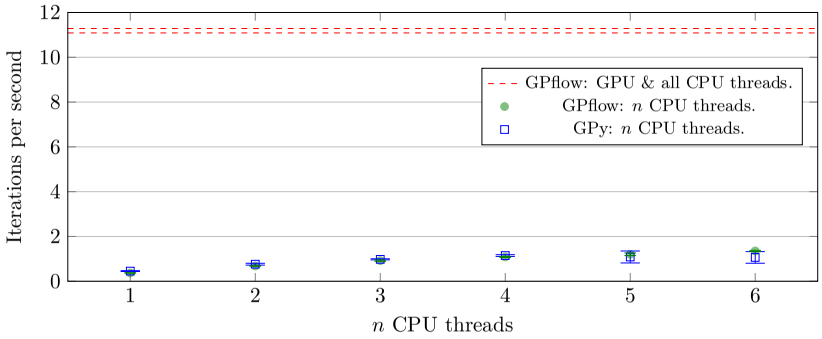


Figure 1: A comparison of iterations of stochastic variational inference per second on the MNIST data set for GPflow and GPy. Error bars shown represent one standard deviation computed from five repeats of the experiment.

used a Linux workstation Intel Core I7-4930K CPU clocked at 3.40GHz an NVIDIA GM200 Geforce GTX Titan X GPU.

The results of the timing experiments are shown in Figure 1. For the CPU experiments, the speeds for GPflow and GPy are similar. It can be seen that the increase in speed from adding a GPU is considerable. These gains could make a significant difference to the work flow of a researcher on this topic. Based on the measurements we have made, training using GPflow with 6 CPU threads would take approximately 41 hours or just under 2 days. Adding a GPU would currently reduce the training time to about 5 hours.

Acknowledgments

We acknowledge contributions from Valentine Svensson, Dan Marthaler, David J. Harris, Rasmus Munk Larsen and Eugene Brevdo. We acknowledge EPSRC grants EP/I036575/1, EP/N014162/1 and EP/N510129/1. James Hensman was supported by an MRC fellowship.

References

Martín Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems.

Sivaram Ambikasaran, Daniel Foreman-Mackey, Leslie Greengard, David W. Hogg, and Michael O'Neil. Fast direct methods for Gaussian processes and the analysis of NASA Kepler mission data. *arXiv preprint 1403.6015*, 2014.

Zhenwen Dai, Andreas Damianou, James Hensman, and Neil Lawrence. Gaussian process models with parallelization and GPU acceleration. *arXiv preprint 1410.4984*, 2014.

GPy. GPy: A Gaussian process framework in Python. `http://github.com/SheffieldML/GPy`, since 2012.

James Hensman, Nicolo Fusi, and Neil D Lawrence. Gaussian processes for big data. In *UAI*, July 2013.

James Hensman, Alexander G. de G. Matthews, Maurizio Filippone, and Zoubin Ghahramani. MCMC for variationally sparse Gaussian processes. In *NIPS*, December 2015a.

James Hensman, Alexander G. de G. Matthews, and Zoubin Ghahramani. Scalable variational Gaussian process classification. In *AISTATS*, May 2015b.

Alexander G. de G. Matthews. *Scalable Gaussian Process Inference using Variational Methods*. PhD thesis, University of Cambridge, 2016.

Alexander G. de G. Matthews, James Hensman, Richard E. Turner, and Zoubin Ghahramani. On Sparse variational methods and the Kullback-Leibler divergence between stochastic processes. In *AISTATS*, May 2016.

Iain Murray. Differentiation of the Cholesky decomposition. *arXiv preprint 1602.07527*, February 2016.

Radford M. Neal. MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*, pages 113–162, 2010.

Manfred Opper and Cédric Archambeau. The variational Gaussian approximation revisited. *Neural Computation*, 21(3):786–792, 2009.

Fabian Pedregosa et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Carl E. Rasmussen and Hannes Nickisch. Gaussian Processes for machine learning (GPML) toolbox. *Journal of Machine Learning Research*, 11:3011–3015, 11 2010.

Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

Michalis K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *AISTATS*, April 2009.

Jarno Vanhatalo, Jaakko Riihimäki, Jouni Hartikainen, Pasi Jylänki, Ville Tolvanen, and Aki Vehtari. GPstuff: Bayesian modeling with Gaussian processes. *Journal of Machine Learning Research*, 14(1):1175–1179, 2013.

Journal of Machine Learning Research 18 (2017) 1-49 Submitted 3/16; Revised 1/17; Published 5/17

COEVOLVE: A Joint Point Process Model for Information Diffusion and Network Evolution*◇

Mehrdad Farajtabar MEHRDAD@GATECH.EDU
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA

Yichen Wang YICHEN.WANG@GATECH.EDU
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA

Manuel Gomez-Rodriguez MANUELGR@MPI-SWS.ORG
MPI for Software Systems
Paul-Ehrlich-Strasse, 67663 Kaiserslautern
Germany

Shuang Li SLI370@GATECH.EDU
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332, USA

Hongyuan Zha ZHA@CC.GATECH.EDU
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA

Le Song LSONG@CC.GATECH.EDU
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332, USA

Editor: Edo Airoldi

Abstract

Information diffusion in online social networks is affected by the underlying network topology, but it also has the power to change it. Online users are constantly creating new links when exposed to new information sources, and in turn these links are alternating the way information spreads. However, these two highly intertwined stochastic processes, information diffusion and network evolution, have been predominantly studied *separately*, ignoring their co-evolutionary dynamics.

We propose a temporal point process model, COEVOLVE, for such joint dynamics, allowing the intensity of one process to be modulated by that of the other. This model allows us to efficiently simulate interleaved diffusion and network events, and generate traces obeying common diffusion and network patterns observed in real-world networks such as Twitter. Furthermore, we also develop a convex optimization framework to learn the parameters of the model from historical diffusion and network evolution traces. We experimented with both synthetic data and data gathered from Twitter, and show that our model provides a good fit to the data as well as more accurate predictions than alternatives.

Keywords: social networks, information diffusion, network structure, co-evolutionary dynamics, point processes, Hawkes process, survival analysis

. *Preliminary version of this work appeared in (Farajtabar et al., 2015b).

. ◇ Implementation codes are available at https://github.com/farajtabar/Coevolution

©2017 Mehrdad Farajtabar, Yichen Wang, Manuel Gomez-Rodriguez, Shuang Li, Hongyuan Zha, Le Song.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-132.html.

1. Introduction

Online social networks, such as Twitter or Weibo, have become large information networks where people share, discuss and search for information of personal interest as well as breaking news (Kwak et al., 2010). In this context, users often forward to their *followers* information they are exposed to via their *followees*, triggering the emergence of information *cascades* that travel through the network (Cheng et al., 2014), and constantly create new links to information sources, triggering changes in the network itself over time. Importantly, recent empirical studies with Twitter data have shown that both information diffusion and network evolution are coupled and network changes are often triggered by information diffusion (Antoniades and Dovrolis, 2015; Weng et al., 2013; Myers and Leskovec, 2014).

While there have been many recent works on modeling information diffusion (Gomez-Rodriguez et al., 2010; Du et al., 2013; Cheng et al., 2014; Farajtabar et al., 2015a) and network evolution (Chakrabarti et al., 2004; Leskovec et al., 2008, 2010), most of them treat these two stochastic processes independently and separately, ignoring the influence one may have on the other over time. More notably, Weng et al. (2013) were the first to show experimental evidence that information diffusion influences network evolution in microblogging sites both at system-wide and individual levels. In particular, they studied *Yahoo! Meme*, a social micro-blogging site similar to Twitter, which was active between 2009 and 2012, and showed that information diffusion causes about 24 % of the new links, and that the likelihood of a new link from a user to another increases with the number of posts by the second user seen by the first one. Also, Antoniades and Dovrolis (2015) studied the temporal characteristics of retweet-driven connections within the Twitter network and realized that the number of retweets is an important factor to infer such connections. They showed that links created due to information diffusion account for 42 % of new links, and are rather infrequent, compared to tweets and retweets, but they are responsible for a large percentage of the new links in Twitter.

However, there are a few limitations in the above-mentioned studies. First, they only characterize the effect that information diffusion has on the network dynamics, or the vice versa, but not the bidirectional influence. Second, previous studies are mostly empirical and only make binary predictions on link creation events without precise timing. For example, the work of (Weng et al., 2013; Antoniades and Dovrolis, 2015) predict whether a new link will be created based on the number of retweets; and, Myers and Leskovec (2014) predict whether a burst of new links will occur based on the number of retweets and users' similarity. Thus, to better understand information diffusion and network evolution, there is an urgent need for joint probabilistic models of the two processes, which are largely inexistent to date.

In this paper, we propose a probabilistic generative model, COEVOLVE, for the joint dynamics of information diffusion and network evolution on Twitter-like networks. Our

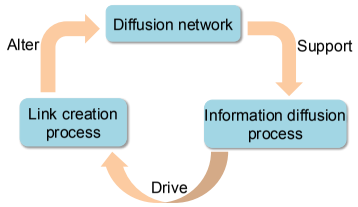


Figure 1: Illustration of how information diffusion and network structure processes interact

model is able to learn parameters from real world data, and predict the precise timing of both diffusion and new link events. The proposed model is based on the framework of temporal point processes, which explicitly characterizes the continuous time interval between events, and it consists of two interwoven and interdependent components, as shown in Figure 1:

I. **Information diffusion process.** We design an "identity revealing" multivariate Hawkes process (Farajtabar et al., 2014) to capture the mutual excitation behavior of retweeting events, where the intensity of such events in a user is boosted by previous events from her time-varying set of followees. Although Hawkes processes have been used for information diffusion before (Farajtabar et al., 2016; Blundell et al., 2012; Iwata et al., 2013; Zhou et al., 2013a; Farajtabar et al., 2014; Linderman and Adams, 2014; Valera and Gomez-Rodriguez, 2015), the key innovation of our approach is to explicitly model the excitation due to a particular source node, hence revealing the identity of the source. Such design reflects the reality that information sources are explicitly acknowledged, and it also allows a particular information source to acquire new links in a rate according to her "informativeness".

II. **Network evolution process.** We model link creation as an "information driven" survival process, and couple the intensity of this process with retweeting events. Although survival processes have been used for link creation before (Hunter et al., 2011; Vu et al., 2011), the key innovation in our model is to incorporate retweeting events as the driving force for such processes. Since our model has captured the source identity of each retweeting event, new links will be targeted toward information sources, with an intensity proportional to their degree of excitation and each source's influence.

Our model is designed in such a way that it allows the two processes, information diffusion and network evolution, unfold simultaneously in the same time scale and exercise bidirectional influence on each other, allowing sophisticated coevolutionary dynamics to be generated, as illustrated in Figure 2.

Importantly, the flexibility of our model does not prevent us from efficiently simulating diffusion and link events from the model and learning its parameters from real world data:

- **Efficient simulation.** We design a scalable sampling procedure that exploits the sparsity of the generated networks. Its complexity is $O(nd \log m)$, where n is the number of events, m is the number of users and d is the maximum number of followees per user.

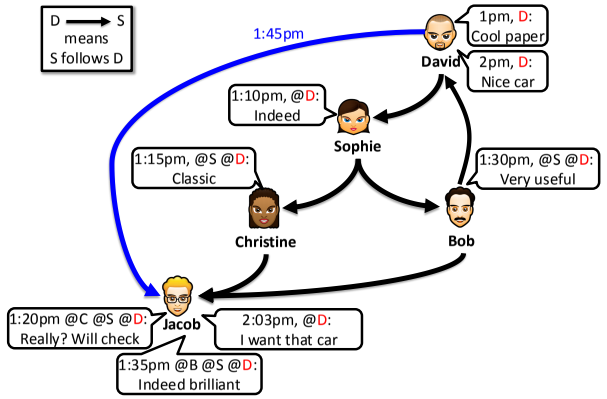


Figure 2: Illustration of information diffusion and network structure co-evolution.
Information diffusion → network structure: David's tweet at 1:00 pm about a paper is retweeted by Sophie and Christine respectively at 1:10 pm and 1:15 pm to reach out to Jacob. Jacob retweets about this paper at 1:20 pm and 1:35 pm and then finds David a good source of information and decides to follow him directly at 1:45 pm.
Information diffusion → network structure: As a result new path of information from David to Jacob (and his downstream followers) is created. Consequently, a subsequent tweet by David about a car at 2:00 pm directly reaches out to Jacob without need to Sophie and Christine retweet.

- **Convex parameters learning.** We show that the model parameters that maximize the joint likelihood of observed diffusion and link creation events can be efficiently found via convex optimization.

Then, we experiment with our model and show that it can produce coevolutionary dynamics of information diffusion and network evolution, and generate retweet and link events that obey common information diffusion patterns (*e.g.*, cascade structure, size and depth), static network patterns (*e.g.*, node degree) and temporal network patterns (*e.g.*, shrinking diameter) described in related literature (Leskovec et al., 2005, 2010; Goel et al., 2012). Finally, we show that, by modeling the coevolutionary dynamics, our model provides significantly more accurate link and diffusion event predictions than alternatives in large scale Twitter data set (Antoniades and Dovrolis, 2015).

The remainder of this article is organized as follows. We first proceed by building sufficient background on the temporal point processes framework in Section 2. Then, we introduce our joint model of information diffusion and network structure co-evolution in Section 3. Sections 4 and 5 are devoted to answer two essential questions: how can we generate data from the model? and how can we efficiently learn the model parameters from historical event data? Any generative model should be able to answer the above questions.

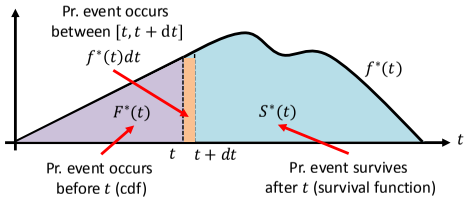


Figure 3: Illustration of three inter-related quantities in point processes framework: conditional density function, conditional cumulative density function, and survival function.

In Sections 6, 7, and 8 we perform empirical investigation of the properties of the model, we evaluate the accuracy of the parameter estimation in synthetic data, and we evaluate the performance of the proposed model in the real-world data set, respectively. Section 9 reviews the related work, and finally, the paper is concluded in Section 10. Proofs, more detailed contents and experimental results, and some extensions are left to the appendices.

2. Background on Temporal Point Processes

A temporal point process is a random process whose realization consists of a list of discrete events localized in time, $\{t_i\}$ with $t_i \in \mathbb{R}^+$ and $i \in \mathbb{Z}^+$. Many different types of data produced in online social networks can be represented as temporal point processes, such as the times of retweets and link creations. A temporal point process can be equivalently represented as a counting process, $N(t)$, which records the number of events before time t. Let the history $\mathcal{H}(t)$ be the list of times of events $\{t_1, t_2, \ldots, t_n\}$ up to but not including time t. Then, the number of observed events in a small time window $[t, t + dt)$ of length dt is

$$dN(t) = \sum_{t_i \in \mathcal{H}(t)} \delta(t - t_i)\, dt, \tag{1}$$

and hence $N(t) = \int_0^t dN(s)$, where $\delta(t)$ is a Dirac delta function. More generally, given a function $f(t)$, we can define the convolution with respect to $dN(t)$ as

$$f(t) \star dN(t) := \int_0^t f(t - \tau)\, dN(\tau) = \sum\nolimits_{t_i \in \mathcal{H}(t)} f(t - t_i). \tag{2}$$

The point process representation of temporal data is fundamentally different from the discrete time representation typically used in social network analysis. It directly models the time interval between events as random variables, avoids the need to pick a time window to aggregate events, and allows temporal events to be modeled in a fine grained fashion. Moreover, it has a remarkably rich theoretical support (Aalen et al., 2008).

An important way to characterize temporal point processes is via the conditional intensity function—a stochastic model for the time of the next event given all the times of previous events. Formally, the conditional intensity function $\lambda^*(t)$ (intensity, for short) is

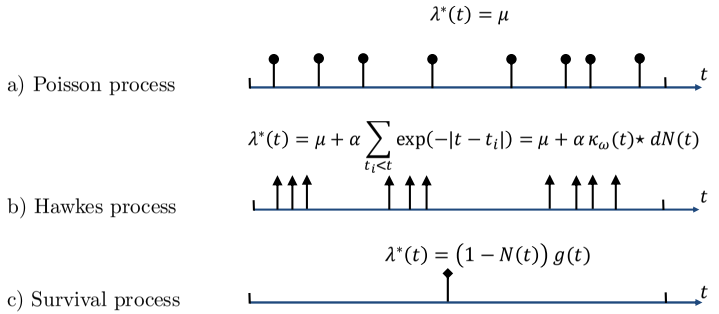


Figure 4: Three types of point processes with a typical realization

the conditional probability of observing an event in a small window $[t, t + dt)$ given the history $\mathcal{H}(t)$, *i.e.*,

$$\lambda^*(t)dt := \mathbb{P}\left\{\text{event in } [t, t + dt) | \mathcal{H}(t)\right\} = \mathbb{E}[dN(t) | \mathcal{H}(t)], \tag{3}$$

where one typically assumes that only one event can happen in a small window of size dt and thus $dN(t) \in \{0, 1\}$. Then, given the observation until time t and a time $t' \geqslant t$, we can also characterize the conditional probability that no event happens until t' as

$$S^*(t') = \exp\left(-\int_t^{t'} \lambda^*(\tau)\, d\tau\right), \tag{4}$$

the (conditional) probability density function that an event occurs at time t' as

$$f^*(t') = \lambda^*(t')\, S^*(t'), \tag{5}$$

and the (conditional) cumulative density function, which accounts for the probability that an event happens before time t':

$$F^*(t') = 1 - S^*(t') = \int_t^{t'} f^*(\tau)\, d\tau. \tag{6}$$

Figure 3 illustrates these quantities. Moreover, we can express the log-likelihood of a list of events $\{t_1, t_2, \ldots, t_n\}$ in an observation window $[0, T)$ as

$$\mathfrak{L} = \sum_{i=1}^{n} \log \lambda^*(t_i) - \int_0^T \lambda^*(\tau)\, d\tau, \quad T \geqslant t_n. \tag{7}$$

This simple log-likelihood will later enable us to learn the parameters of our model from observed data. Finally, the functional form of the intensity $\lambda^*(t)$ is often designed to capture the phenomena of interests. Some useful functional forms we will use are (Aalen et al., 2008):

(i) **Poisson process.** The intensity is assumed to be independent of the history $\mathcal{H}(t)$, but it can be a nonnegative time-varying function, *i.e.*,

$$\lambda^*(t) = g(t) \geqslant 0. \tag{8}$$

(ii) **Hawkes process.** The intensity is history dependent and models a mutual excitation between events, *i.e.*,

$$\lambda^*(t) = \mu + \alpha\kappa_\omega(t) \star dN(t) = \mu + \alpha \sum_{t_i \in \mathcal{H}(t)} \kappa_\omega(t - t_i), \tag{9}$$

where,

$$\kappa_\omega(t) := \exp(-\omega t)\mathbb{I}[t \geqslant 0] \tag{10}$$

is an exponential triggering kernel and $\mu \geqslant 0$ is a baseline intensity independent of the history. Here, the occurrence of each historical event increases the intensity by a certain amount determined by the kernel and the weight $\alpha \geqslant 0$, making the intensity history dependent and a stochastic process by itself. In our work, we focus on the exponential kernel, however, other functional forms, such as log-logistic function, are possible, and the general properties of our model do not depend on this particular choice.

(iii) **Survival process.** There is only one event for an instantiation of the process, *i.e.*,

$$\lambda^*(t) = (1 - N(t))g(t), \tag{11}$$

where $g(t) \geqslant 0$ and the term $(1 - N(t))$ makes sure $\lambda^*(t)$ is 0 if an event already happened before t.

Figure 4 illustrates these processes. Interested reader should refer to Aalen et al. (2008) for more details on the framework of temporal point processes.

Point process models have been applied to many tasks in networks such as fake news mitigation (Farajtabar et al., 2017), recommendation systems (Hosseini et al., 2017), outlier detection (Li et al., 2017), activity shaping in social networks (Farajtabar et al., 2014), verifying crowd-generated data (Tabibian et al., 2017), sequence modeling using deep recurrent neural networks (Xiao et al., 2017), and campaigning in networks (Farajtabar et al., 2016).

3. Generative Model of Information Diffusion and Network Evolution

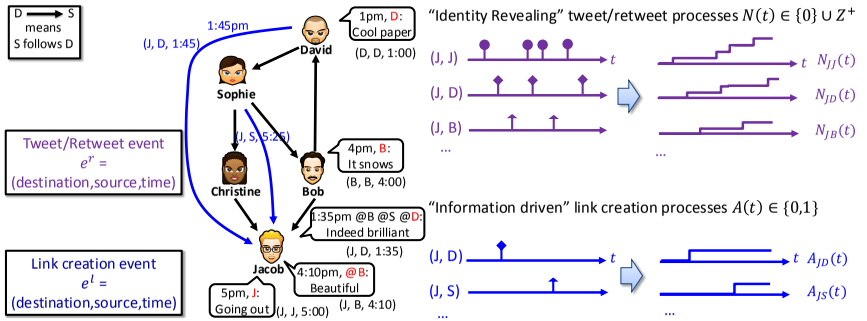
In this section, we use the above background on temporal point processes to formulate Coevolve, our probabilistic model for the joint dynamics of information diffusion and network evolution.

3.1 Event Representation

We model the generation of two types of events: tweet/retweet events, e^r, and link creation events, e^l. Instead of just the time t, we record each event as a triplet, as illustrated in Figure 5(a):

$$e^r \quad \text{or} \quad e^l \quad := \quad (\underset{\text{destination}}{u}, \overset{\text{source}}{s}, \underset{\text{time}}{t}). \tag{12}$$

For retweet event, the triplet means that the destination node u retweets at time t a tweet originally posted by source node s. Recording the source node s reflects the real world scenario that information sources are explicitly acknowledged. Note that the occurrence of event e^r does *not* mean that u is directly retweeting from or is connected



(a) Event representation (b) Point and counting processes

Figure 5: Events of point processes and their associated counting processes for link creation and information diffusion; (a) Trace of events generated by a tweet from David followed by new links Jacob creates to follow David and Sophie; (b) The associated points in time and the counting process realizations.

to s. This event can happen when u is retweeting a message by another node u' where the original information source s is acknowledged. Node u will pass on the same source acknowledgement to its followers (*e.g.*, "I agree @a @b @c @s"). Original tweets posted by node u are allowed in this notation. In this case, the event will simply be $e^r = (u, u, t)$. Given a list of retweet events up to but not including time t, the history $\mathcal{H}^r_{us}(t)$ of retweets by u due to source s is

$$\mathcal{H}^r_{us}(t) = \{e^r_i = (u_i, s_i, t_i) | u_i = u \text{ and } s_i = s\}. \tag{13}$$

The entire history of retweet events is denoted as

$$\mathcal{H}^r(t) := \cup_{u,s \in [m]} \mathcal{H}^r_{us}(t). \tag{14}$$

For link creation event, the triplet means that destination node u creates at time t a link to source node s, *i.e.*, from time t on, node u starts following node s. To ease the exposition, we restrict ourselves to the case where links cannot be deleted and thus each (directed) link is created only once. However, our model can be easily augmented to consider multiple link creations and deletions per node pair, as discussed in Section E. We denote the link creation history as $\mathcal{H}^l(t)$.

3.2 Joint Model with Two Interwoven Components

Given m users, we use two sets of counting processes to record the generated events, one for information diffusion and another for network evolution. More specifically,

I. Retweet events are recorded using a matrix $\boldsymbol{N}(t)$ of size $m \times m$ for each fixed time point t. The (u, s)-th entry in the matrix, $N_{us}(t) \in \{0\} \cup \mathbb{Z}^+$, counts the number of retweets of u due to source s up to time t. These counting processes are "identity revealing", since they keep track of the source node that triggers each retweet. The

matrix $\boldsymbol{N}(t)$ is typically less sparse than $\boldsymbol{A}(t)$, since $N_{us}(t)$ can be nonzero even when node u does not directly follow s. We also let $d\boldsymbol{N}(t) := (\, dN_{us}(t)\,)_{u,s\in[m]}$.

II. Link events are recorded using an adjacency matrix $\boldsymbol{A}(t)$ of size $m \times m$ for each fixed time point t. The (u, s)-th entry in the matrix, $A_{us}(t) \in \{0, 1\}$, indicates whether u is directly following s. Therefore, $A_{us}(t) = 1$ means the directed link has been created before t. For simplicity of exposition, we do not allow self-links. The matrix $\boldsymbol{A}(t)$ is typically sparse, but the number of nonzero entries can change over time. We also define $d\boldsymbol{A}(t) := (\, dA_{us}(t)\,)_{u,s\in[m]}$.

Then, the interwoven information diffusion and network evolution processes can be characterized using their respective intensities

$$\mathbb{E}[d\boldsymbol{N}(t) \,|\, \mathcal{H}^r(t) \cup \mathcal{H}^l(t)] = \boldsymbol{\Gamma}^*(t)\, dt \tag{15}$$

$$\mathbb{E}[d\boldsymbol{A}(t) \,|\, \mathcal{H}^r(t) \cup \mathcal{H}^l(t)] = \boldsymbol{\Lambda}^*(t)\, dt, \tag{16}$$

where,

$$\boldsymbol{\Gamma}^*(t) = (\, \gamma^*_{us}(t)\,)_{u,s\in[m]} \tag{17}$$

$$\boldsymbol{\Lambda}^*(t) = (\, \lambda^*_{us}(t)\,)_{u,s\in[m]}. \tag{18}$$

The sign * means that the intensity matrices will depend on the joint history, $\mathcal{H}^r(t) \cup \mathcal{H}^l(t)$, and hence their evolution will be coupled. By this coupling, we make: (i) the counting processes for link creation to be "information driven" and (ii) the evolution of the linking structure to change the information diffusion process. In the next two sections, we will specify the details of these two intensity matrices.

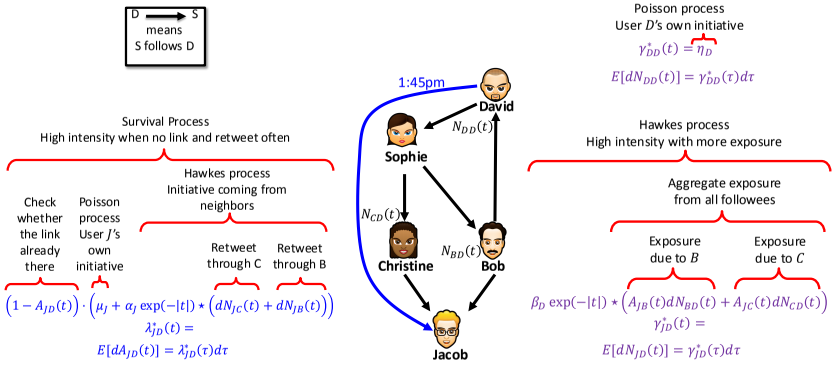
3.3 Information Diffusion Process

We model the intensity, $\boldsymbol{\Gamma}^*(t)$, for retweeting events using multivariate Hawkes process:

$$\gamma^*_{us}(t) = \mathbb{I}[u = s]\, \eta_u + \mathbb{I}[u \neq s]\, \beta_s \sum_{v\in\mathcal{F}_u(t)} \kappa_{\omega_1}(t) \star (A_{uv}(t)\; dN_{vs}(t))\,, \tag{19}$$

where $\mathbb{I}[\cdot]$ is the indicator function and $\mathcal{F}_u(t) := \{v \in [m] : A_{uv}(t) = 1\}$ is the current set of followees of u. The term $\eta_u \geqslant 0$ is the intensity of original tweets by a user u on his own initiative, becoming the source of a cascade, and the term $\beta_s \sum_{v\in\mathcal{F}_u(t)} \kappa_{\omega_1}(t) \star (A_{uv}(t)\; dN_{vs}(t))$ models the propagation of peer influence over the network, where the triggering kernel $\kappa_{\omega_1}(t)$ models the decay of peer influence over time.

Note that the retweeting intensity matrix $\boldsymbol{\Gamma}^*(t)$ is by itself a stochastic process that depends on the time-varying network topology, the non-zero entries in $\boldsymbol{A}(t)$, whose growth is controlled by the network evolution process in Section 3.4. Hence the model design captures the influence of the network topology and each source's influence, β_s, on the information diffusion process. More specifically, to compute $\gamma^*_{us}(t)$, one first finds the current set $\mathcal{F}_u(t)$ of followees of u, and then aggregates the retweets of these followees that are due to source s. Note that these followees may or may not *directly* follow source s. Then, the more frequently node u is exposed to retweets of tweets originated from source s via her followees, the more likely she will also retweet a tweet originated from source s. Once node u retweets due to source s, the corresponding $N_{us}(t)$ will be incremented, and this in turn will increase the likelihood of triggering retweets due to source s among the followers of u. Thus, the source does *not* simply broadcast the message to nodes directly following her but her influence



(a) Link creation process (b) Social network (c) Information diffusion process

Figure 6: Our hypothetical social network where the information diffusion paths make Jacob follow David: (a) The breakdown of conditional intensity function for the link creation process of Jacob following David $A_{JD}(t)$; (b) The information paths between Jacob and David; (c) The information diffusion processes for David tweeting on his own initiative $N_{DD}(t)$ and Jacob retweeting posts originated from David $N_{JD}(t)$.

propagates through the network even to those nodes that do not directly follow her. Finally, this information diffusion model allows a node to repeatedly generate events in a cascade, and is very different from the independent cascade or linear threshold models (Kempe et al., 2003) which allow at most one event per node per cascade.

3.4 Network Evolution Process

In our model, each user is exposed to information through a time-varying set of neighbors. By doing so, information diffusion affects network evolution, increasing the practical application of our model to real-world network data sets. The particular definition of exposure (*e.g.*, a retweet's neighbor) depends on the type of historical information that is available. Remarkably, the flexibility of our model allows for different types of diffusion events, which we can broadly classify into two categories.

In the first category, events corresponds to the times when an information cascade hits a person, for example, through a retweet from one of her neighbors, but she does not explicitly like or forward the associated post. Here, we model the intensity, $\boldsymbol{\Lambda}^*(t)$, for link creation using a combination of survival and Hawkes process:

$$\lambda^*_{us}(t) = (1 - A_{us}(t)) \left(\mu_u + \alpha_u \sum_{v\in\mathcal{F}_u(t)} \kappa_{\omega_2}(t) \star dN_{vs}(t) \right), \tag{20}$$

where the term $1 - A_{us}(t)$ effectively ensures a link is created only once, and after that, the corresponding intensity is set to zero. The term $\mu_u \geqslant 0$ denotes a baseline intensity, which models when a node u decides to follow a source s spontaneously at her own initiative. The term $\alpha_u \kappa_{\omega_2}(t) \star dN_{vs}(t)$ corresponds to the retweets by node v (a followee of node u) which are originated from source s. The triggering kernel $\kappa_{\omega_2}(t)$ models the decay of interests over time.

In the second category, the person decides to explicitly like or forward the associated post and influencing events correspond to the times when she does so. In this case, we model the intensity, $\mathbf{\Lambda}^*(t)$, for link creation as:

$$\lambda^*_{us}(t) = (1 - A_{us}(t))(\mu_u + \alpha_u \kappa_{\omega_2}(t) \star dN_{us}(t)), \tag{21}$$

where the terms $1 - A_{us}(t)$, $\mu_u \geqslant 0$, and the decaying kernel $\kappa_{\omega_2}(t)$ play the same role as the corresponding ones in Equation (20). The term $\alpha_u \kappa_{\omega_2}(t) \star dN_{us}(t)$ corresponds to the retweets of node u due to tweets originally published by source s. The higher the corresponding retweet intensity, the more likely u will find information by source s useful and will create a *direct* link to s.

In both cases, the link creation intensity $\mathbf{\Lambda}^*(t)$ is also a stochastic process by itself, which depends on the retweet events, be it the retweets by the neighbors of node u or the retweets by node u herself, respectively. Therefore, it captures the influence of retweets on the link creation, and closes the loop of mutual influence between information diffusion and network topology. Figure 6 illustrates these two interdependent intensities.

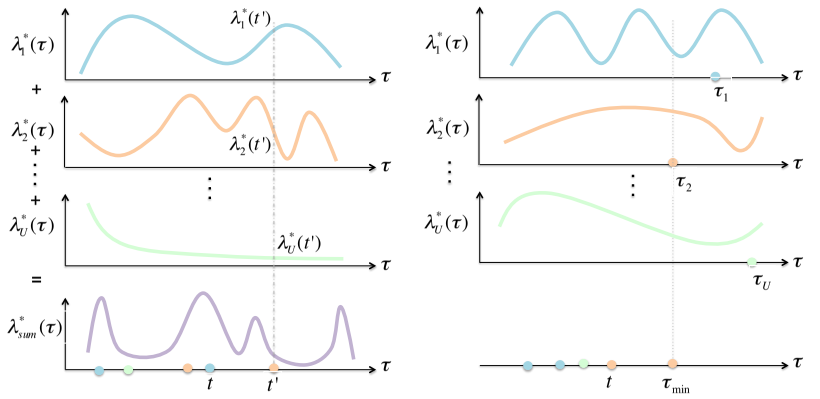
Intuitively, in the latter category, information diffusion events are more prone to trigger new connections, because, they involve the target and source nodes in an explicit interaction, however, they are also less frequent. Therefore, it is mostly suitable to large event data sets, as the ones we generate in our synthetic experiments. In contrast, in the former category, information diffusion events are less likely to inspire new links but found in abundance. Therefore, it is more suitable for smaller data sets, as the ones we use in our real-world experiments. Consequently, in our synthetic experiments we used the latter and in our real-world experiments, we used the former.

More generally, the choice of exposure event should be made based on the type and amount of available historical information. Note that, these are two realizations, among the many others, of the link formation process in the Coevolve framework. Many other extensions can be found in appendix E. In practice, these different forms can be utilized depending on the conditions and constraints of the networks and the data in hand. More importantly, in this section, we make one example on how to tailor the model to fit the application best.

Finally, note that creating a link is more than just adding a path or allowing information sources to take shortcuts during diffusion. The network evolution makes fundamental changes to the diffusion dynamics and stationary distribution of the diffusion process in Section 3.3. As shown in Farajtabar et al. (2014), given a fixed network structure $\boldsymbol{A}$, the expected retweet intensity $\boldsymbol{\mu}_s(t)$ at time t due to source s will depend of the network structure in a nonlinear fashion, *i.e.*,

$$\boldsymbol{\mu}_s(t) := \mathbb{E}[\mathbf{\Gamma}^*_{:s}(t)] = (e^{(\boldsymbol{A}-\omega_1 \boldsymbol{I})t} + \omega_1 (\boldsymbol{A} - \omega_1 \boldsymbol{I})^{-1} (e^{(\boldsymbol{A}-\omega_1 \boldsymbol{I})t} - \boldsymbol{I}))\, \boldsymbol{\eta}_s, \tag{22}$$

where $\boldsymbol{\eta}_s \in \mathbb{R}^m$ has a single nonzero entry with value η_s and $e^{(\boldsymbol{A}-\omega_1 \boldsymbol{I})t}$ is the matrix exponential. When $t \to \infty$, the stationary intensity $\bar{\boldsymbol{\mu}}_s = (\boldsymbol{I} - \boldsymbol{A}/\omega)^{-1} \boldsymbol{\eta}_s$ is also nonlinearly



(a) Ogata's algorithm (b) Proposed algorithm

Figure 7: Ogata's algorithm vs our simulation algorithm in simulating U interdependent point processes characterized by intensity functions $\lambda_1(t), \ldots, \lambda_U(t)$; (a) Illustrating Ogata's algorithm, which first takes a sample from the process with intensity equal to sum of individual intensities and then assigns it to the proper dimension proportionally to its contribution to the sum of intensities; (b) Illustrating our proposed algorithm, which first draws a sample from each dimension independently and then takes the minimum time among them.

related to the network structure. Thus, given two network structures $\boldsymbol{A}(t)$ and $\boldsymbol{A}(t')$ at two points in time, which are different by a few edges, the effect of these edges on the information diffusion is not just an additive relation. Depending on how these newly created edges modify the eigen-structure of the sparse matrix $\boldsymbol{A}(t)$, their effect on the information diffusion dynamics can be very significant.

4. Efficient Simulation of Coevolutionary Dynamics

We could simulate samples (link creations, tweets and retweets) from our model by adapting Ogata's thinning algorithm (Ogata, 1981), originally designed for multidimensional Hawkes processes. However, a naive implementation of Ogata's algorithm would scale poorly, *i.e.*, for each sample, we would need to re-evaluate $\mathbf{\Gamma}^*(t)$ and $\mathbf{\Lambda}^*(t)$. Thus, to draw n sample events, we would need to perform $O(m^2 n^2)$ operations, where m is the number of nodes. Figure 7(a) schematically demonstrates the main steps of Ogata's algorithm. Please refer to Appendix B for further details.

Here, we design a sampling procedure that is especially well-fitted for the structure of our model. The algorithm is based on the following key idea: if we consider each intensity function in $\mathbf{\Gamma}^*(t)$ and $\mathbf{\Lambda}^*(t)$ as a separate point process and draw a sample from each, the minimum among all these samples is a valid sample for the multidimensional point process.

Algorithm 1 Simulation Algorithm for Coevolve

Initialization:
Initialize the priority queue Q
for $\forall\ u, s \in [m]$ **do**
  Sample next link event e^l_{us} from A_{us} (Algorithm 3)
  $Q.insert(e^l_{us})$
  Sample next retweet event e^r_{us} from N_{us} (Algorithm 3)
  $Q.insert(e^r_{us})$
end for
General Subroutine:
$t \leftarrow 0$
while $t < T$ **do**
  $e \leftarrow Q.extract_min()$
  if $e = (u, s, t')$ is a retweet event **then**
    Update the history $\mathcal{H}^r_{us}(t') = \mathcal{H}^r_{us}(t) \cup \{e\}$
    for $\forall\ v$ *s.t.* $u \rightsquigarrow v$ **do**
      Update event intensity: $\gamma_{vs}(t') = \gamma_{vs}(t'^{-}) + \beta$
      Sample retweet event e^r_{vs} from γ_{vs} (Algorithm 3)
      $Q.update_key(e^r_{vs})$
      if NOT $s \rightsquigarrow v$ **then**
        Update link intensity: $\lambda^*_{vs}(t') = \lambda^*_{vs}(t'^{-}) + \alpha$
        Sample link event e^l_{vs} from λ_{vs} (Algorithm 3)
        $Q.update_key(e^l_{vs})$
      end if
    end for
  else
    Update the history $\mathcal{H}^l_{us}(t') = \mathcal{H}^l_{us}(t) \cup \{e\}$
    $\lambda^*_{us}(t) \leftarrow 0\ \ \forall\ t > t'$
  end if
  $t \leftarrow t'$
end while

As the results of this section are general and can be applied to simulate any multi-dimensional point process model we abuse the notation a little bit and represent U (possibly inter-dependent) point processes by U intensity functions $\lambda^*_1, \ldots, \lambda^*_U$. In the specific case of simulating coevolutionary dynamics we have $U = m^2 + m(m-1)$ were the first and second terms are the number information diffusion and link creation processes, respectively. Figure 7 illustrates the way in which both algorithms differ. The new algorithm has the following steps:

1. Initialization: Simulate each dimension separately and find their next sampled event time.

2. Minimization: Take the minimum among all the sampled times and declare it as the next event of the multidimensional process.

Algorithm 2 Efficient Intensity Computation

Global Variabels:
Last time of intensity computation: t
Last value of intensity computation: I
Initialization:
$t \leftarrow 0$
$I \leftarrow \mu$
function $get_intensity(t')$
  $I' \leftarrow (I - \mu)\exp(-\omega(t' - t)) + \mu$
  $t \leftarrow t'$
  $I \leftarrow I'$
  return I
end function

Algorithm 3 1-D next event sampling

Input: Current time: t
Output: Next event time: s
$s \leftarrow t$
$\hat{\lambda} \leftarrow \lambda^*(s)$ (Algorithm 2)
while $s < T$ **do**
  $g \sim Exponential(\hat{\lambda})$
  $s \leftarrow s + g$
  $\bar{\lambda} \leftarrow \lambda^*(s)$ (Algorithm 2)
  Rejection test:
  $d \sim Uniform(0, 1)$
  if $d \times \hat{\lambda} < \bar{\lambda}$ **then**
    return s
  else
    $\hat{\lambda} = \bar{\lambda}$
  end if
end while
return s

3. Update: Recalculate the intensities of the dimensions that are affected by this approved sample and re-sample only their next event. Then go to step 2.

Note that, events are sampled one by one; After each event, the intensities of affected dimensions are recomputed and fixed. This means that, the history is fixed and the dimensions of the Hawkes process become independent (just until the next event). We take next sample. Then, we make the necessary updates in the intensity functions to capture the influence of dimensions on each other. Therefore, each dimension can be thought independent until next sample is drawn because the intensity function is known.

To prove that the new algorithm generates samples from the same distribution as Ogata's algorithm does we need the following Lemma. It justifies step 2 of the above outline.

Lemma 1 *Assume we have U independent non-homogeneous Poisson processes with intensity $\lambda_1^*(\tau), \ldots, \lambda_U^*(\tau)$. Take random variable τ_u equal to the time of process u's first event after time t. Define $\tau_{min} = \min_{1\le u\le U}\{\tau_u\}$ and $u_{min} = \operatorname{argmin}_{1\le u\le U}\{\tau_u\}$. Then,*

(a) τ_{min} is the first event after time t of the Poisson process with intensity $\lambda_{sum}^*(\tau)$. In other words, τ_{min} has the same distribution as the next event (t') in Ogata's algorithm.

(b) u_{min} follows the conditional distribution $\mathbb{P}(u_{min} = u|\tau_{min} = x) = \frac{\lambda_U^*(x)}{\lambda_{sum}^*(x)}$. I.e. the dimension firing the event comes from the same distribution as the one in Ogata's algorithm.

Given the above Lemma, we can now prove that the distribution of the samples generated by the proposed algorithm is identical to the one generated by Ogata's method.

Theorem 2 *The sequence of samples from Ogata's algorithm and our proposed algorithm follow the same distribution.*

It's noteworthy that the dimensions are not independent, but their dependency is considered in the update stage. Until the next update happens they can be considered independent because their intensity function is determined and fixed.

This new algorithm is specially suitable for the structure of our inter-coupled processes. Since social and information networks are typically sparse, every time we sample a new node (or link) event from the model, only a small number of intensity functions in the local neighborhood of the node (or the link), will change. This number is of $O(d)$ where d is the maximum number of followers/followees per node. As a consequence, we can reuse most of the individual samples for the next overall sample. Moreover, we can find which intensity function has the minimum sample time in $O(\log m)$ operations using a heap priority queue. The heap data structure will help maintain the minimum and find it in logarithmic time with respect to the number of elements therein. Therefore, we have reduced an $O(nm)$ factor in the original algorithm to $O(d \log m)$.

Finally, we exploit the properties of the exponential function to update individual intensities for each new sample in $O(1)$. For simplicity consider a Hawkes process with intensity $\lambda^*(t) = \mu + \sum_{t_i\in\mathcal{H}_t} \alpha \exp(-\omega(t-t_i))$. Note that both link creation and information diffusion processes have this structure. Now, let $t_i < t_{i+1}$ be two arbitrary times, we have

$$\lambda^*(t_{i+1}) = (\lambda^*(t_i) - \mu)\exp(-\omega(t_{i+1} - t_i)) + \mu. \tag{23}$$

It can be readily generalized to the multivariate case too. Therefore, we can compute the current intensity without explicitly iterating over all previous events. As a result we can change an $O(n)$ factor in the original algorithm to $O(1)$. Furthermore, the exponential kernel also facilitates finding the upper bound of the intensity since it always lies at the beginning of one of the processes taken into consideration. Algorithm 2 summarizes the procedure to compute intensities with exponential kernels, and Algorithm 3 shows the procedure to sample the next event in each dimension making use of the special property of exponential triggering kernel.

The simulation algorithm is shown in Algorithm 1. By using this algorithm we reduce the complexity from $O(n^2m^2)$ to $O(nd\log m)$, where d is the maximum number of followees per node. That means, our algorithm scales logarithmically with the number of nodes and linearly with the number of edges at any point in time during the simulation. Moreover, events for new links, tweets and retweets are generated in a temporally intertwined and

Algorithm 4 MM-type parameter learning for Coevolve

Input: Set of retweet events $\mathcal{E} = \{e_i^r\}$ and link creation events $\mathcal{A} = \{e_i^l\}$ observed in time window $[0, T)$
Output: Learned parameters $\{\mu_u\}, \{\alpha_u\}, \{\eta_u\}, \{\beta_s\}$
Initialization:
for $u \leftarrow 1$ to m **do**
  Initialize μ_u and α_u randomly
end for
for $u \leftarrow 1$ to m **do**
  $\eta_u = \frac{\sum_{e_i^r\in\mathcal{E}} \mathbb{I}[u=u_i=s_i]}{T}$
end for
for $s \leftarrow 1$ to m **do**
  $\beta_s = \frac{\sum_{e_i^r\in\mathcal{E}} \mathbb{I}[s=s_i\neq u_i]}{\sum_{u\in[m]} \mathbb{I}[u\neq s] \sum_{v\in\mathcal{F}_u(t)} \int_0^T \kappa_{\omega_1}(t) \star (A_{uv}(t)\, dN_{vs}(t))\, dt}$
end for
while not converged **do**
  for $i \leftarrow 1$ to n_l **do**
    $\nu_{i1} = \frac{\mu_{u_i}}{\mu_{u_i} + \alpha_{u_i} \sum_{v\in\mathcal{F}_{u_i}(t_i)} \left(\kappa_{\omega_2}(t) \star dN_{vs}(t)\right)\Big|_{t=t_i}}$
    $\nu_{i2} = \frac{\alpha_{u_i} \sum_{v\in\mathcal{F}_{u_i}(t_i)} \left(\kappa_{\omega_2}(t) \star dN_{vs}(t)\right)\Big|_{t=t_i}}{\mu_{u_i} + \alpha_{u_i} \sum_{v\in\mathcal{F}_{u_i}(t_i)} \left(\kappa_{\omega_2}(t) \star dN_{vs}(t)\right)\Big|_{t=t_i}}$
  end for
  for $u \leftarrow 1$ to m **do**
    $\mu_u = \frac{\sum_{e_i^l\in\mathcal{A}} \mathbb{I}[u=u_i]\, \nu_{i1}}{\sum_{s\in[m]} \int_0^T (1 - A_{us}(t))\, dt}$
    $\alpha_u = \frac{\sum_{e_i^l\in\mathcal{A}} \mathbb{I}[u=u_i] \nu_{i2}}{\sum_{s\in[m]} \int_0^T (1 - A_{us}(t))(\kappa_{\omega_2}(t) \star dN_{us}(t))\, dt}$
  end for
end while

interleaving fashion, since every new retweet event will modify the intensity for link creation and vice versa.

5. Efficient Parameter Estimation from Coevolutionary Events

In this section, we first show that learning the parameters of our proposed model reduces to solving a convex optimization problem and then develop an efficient, parameter-free Minorization-Maximization algorithm to solve such problem.

5.1 Concave Parameter Learning Problem

Given a collection of retweet events $\mathcal{E} = \{e_i^r\}$ and link creation events $\mathcal{A} = \{e_i^l\}$ recorded within a time window $[0, T)$, we can easily estimate the parameters needed in our model using maximum likelihood estimation. To this aim, we compute the joint log-likelihood $\mathfrak{L}$

of these events using Equation (7), *i.e.*,

$$\mathfrak{L}(\{\mu_u\},\{\alpha_u\},\{\eta_u\},\{\beta_s\}) = \underbrace{\sum_{e_i^r\in\mathcal{E}} \log\left(\gamma^*_{u_i s_i}(t_i)\right) - \sum_{u,s\in[m]} \int_0^T \gamma^*_{us}(\tau)\, d\tau}_{\text{tweet / retweet}} + \underbrace{\sum_{e_i^l\in\mathcal{A}} \log\left(\lambda^*_{u_i s_i}(t_i)\right) - \sum_{u,s\in[m]} \int_0^T \lambda^*_{us}(\tau)\, d\tau}_{\text{links}}. \tag{24}$$

For the terms corresponding to retweets, the log term sums only over the actual observed events while the integral term actually sums over all possible combination of destination and source pairs, even if there is no event between a particular pair of destination and source. For such pairs with no observed events, the corresponding counting processes have essentially survived the observation window $[0, T)$, and the term $-\int_0^T \gamma^*_{us}(\tau)d\tau$ simply corresponds to the log survival probability. The terms corresponding to links have a similar structure.

Once we have an expression for the joint log-likelihood of the retweet and link creation events, the parameter learning problem can be then formulated as follows:

$$\begin{array}{ll} \text{minimize}_{\{\mu_u\},\{\alpha_u\},\{\eta_u\},\{\beta_s\}} & -\mathfrak{L}(\{\mu_u\},\{\alpha_u\},\{\eta_u\},\{\beta_s\}) \\ \text{subject to} & \mu_u \geq 0, \quad \alpha_u \geq 0, \quad \eta_u \geq 0, \quad \beta_s \geq 0 \quad \forall\, u, s \in [m]. \end{array} \tag{25}$$

Theorem 3 *The optimization problem defined by Equation* (25) *is jointly convex.*

5.2 Efficient Minorization-Maximization Algorithm

Since the optimization problem is jointly convex with respect to all the parameters, one can simply take any convex optimization method to learn the parameters. However, these methods usually require hyper parameters like step size or initialization, which may significantly influence the convergence. Instead, the structure of our problem allows us to develop an efficient algorithm inspired by previous work (Zhou et al., 2013a,b; Xu et al., 2016), which leverages Minorization Maximization (MM) (Hunter and Lange, 2004) and is parameter free and insensitive to initialization.

Our algorithm utilizes Jensen's inequality to provide a lower bound for the second log-sum term in the log-likelihood given by Equation (24). More specifically, consider a set of arbitrary auxiliary variable ν_{ij}, where $1 \leq i \leq n_l$, $j = 1, 2$ and n_l is the number of link events, *i.e.*, $n_l = |\mathcal{A}|$. Further, assume these variables satisfy

$$\forall\ 1 \leq i \leq n_l : \quad \nu_{i1}, \nu_{i2} \geq 0, \quad \nu_{i1} + \nu_{i2} = 1. \tag{26}$$

The following lemma is preliminary to find a closed form update formula for the parameters.

Lemma 4 *The log-likelihood in Equation* (24) *is lower-bounded as follows:*

$$\begin{aligned} \mathfrak{L} \geq \mathfrak{L}' = & \sum_{e_i^r\in\mathcal{E}} \mathbb{I}[u_i = s_i]\, \log\left(\eta_{u_i}\right) + \sum_{e_i^r\in\mathcal{E}} \mathbb{I}[u_i \neq s_i]\, \log(\beta_{s_i}) \\ & + \sum_{e_i^r\in\mathcal{E}} \mathbb{I}[u_i \neq s_i]\, \log\Big(\sum\nolimits_{v\in\mathcal{F}_{u_i}(t_i)} \left(\kappa_{\omega_1}(t) \star (A_{u_i v}(t)\, dN_{v s_i}(t))\right)\Big|_{t=t_i}\Big) \\ & - \sum_{u,s\in[m]} \eta_u T + \beta_s \sum\nolimits_{v\in\mathcal{F}_u(t)} \int_0^T \kappa_{\omega_1}(t) \star (A_{uv}(t)\, dN_{vs}(t))\, dt \\ & + \sum_{e_i^l\in\mathcal{A}} \nu_{i1} \log(\mu_{u_i}) + \nu_{i2} \log(\alpha_{u_i}) + \nu_{i2} \log\Big(\sum_{v\in\mathcal{F}_{u_i}(t_i)} \left(\kappa_{\omega_2}(t) \star dN_{vs}(t)\right)\Big|_{t=t_i}\Big) \\ & - \sum_{e_i^l\in\mathcal{A}} \nu_{i1} \log(\nu_{i1}) + \nu_{i2} \log(\nu_{i2}) \\ & - \sum_{u,s\in[m]} \mu_u \int_0^T (1 - A_{us}(t))\, dt + \alpha_u \int_0^T (1 - A_{us}(t))(\kappa_{\omega_2}(t) \star dN_{us}(t))\, dt. \end{aligned} \tag{27}$$

Given the above lemma, by taking the gradient of the lower-bound with respect to the parameters, we can find the closed form updates to optimize the lower-bound:

$$\eta_u = \frac{\sum_{e_i^r\in\mathcal{E}} \mathbb{I}[u = u_i = s_i]}{T} \tag{28}$$

$$\beta_s = \frac{\sum_{e_i^r\in\mathcal{E}} \mathbb{I}[s = s_i \neq u_i]}{\sum_{u\in[m]} \mathbb{I}[u \neq s] \sum_{v\in\mathcal{F}_u(t)} \int_0^T \kappa_{\omega_1}(t) \star (A_{uv}(t)\, dN_{vs}(t))\, dt} \tag{29}$$

$$\mu_u = \frac{\sum_{e_i^l\in\mathcal{A}} \mathbb{I}[u = u_i]\, \nu_{i1}}{\sum_{s\in[m]} \int_0^T (1 - A_{us}(t))\, dt} \tag{30}$$

$$\alpha_u = \frac{\sum_{e_i^l\in\mathcal{A}} \mathbb{I}[u = u_i]\, \nu_{i2}}{\sum_{s\in[m]} \int_0^T (1 - A_{us}(t))(\kappa_{\omega_2}(t) \star dN_{us}(t))\, dt}. \tag{31}$$

Finally, although the lower bound is valid for every choice of ν_{ij} satisfying Equation (26), by maximizing the lower bound with respect to the auxiliary variables we can make sure that the lower bound is tight:

$$\begin{array}{lll} \text{maximize}_{\{\nu_{ij}\}} & \mathcal{L}'(\{\mu_u\},\{\alpha_u\},\{\eta_u\},\{\beta_s\},\{\nu_{ij}\}) & \\ \text{subject to} & \nu_{i1} + \nu_{i2} = 1 & \forall\, i : 1 \leq i \leq n_l \\ & \nu_{i0}, \nu_{i1} \geq 0 & \forall\, i : 1 \leq i \leq n_l. \end{array} \tag{32}$$

Fortunately, the above constrained optimization problem can be solved easily via Lagrange multipliers, which leads to closed form updates:

$$\nu_{i1} = \frac{\mu_{u_i}}{\mu_{u_i} + \alpha_{u_i} \sum_{v\in\mathcal{F}_{u_i}(t_i)} \left(\kappa_{\omega_2}(t) \star dN_{vs}(t)\right)\big|_{t=t_i}} \tag{33}$$

$$\nu_{i2} = \frac{\alpha_{u_i} \sum_{v\in\mathcal{F}_{u_i}(t_i)} \left(\kappa_{\omega_2}(t) \star dN_{vs}(t)\right)\big|_{t=t_i}}{\mu_{u_i} + \alpha_{u_i} \sum_{v\in\mathcal{F}_{u_i}(t_i)} \left(\kappa_{\omega_2}(t) \star dN_{vs}(t)\right)\big|_{t=t_i}}. \tag{34}$$

Algorithm 4 summarizes over the learning procedure, which is guaranteed to converge to a global optimum (Hunter and Lange, 2004; Zhou et al., 2013a)

It's notable that, the maximum likelihood is prone to fall into overfitting, therefore, there is a wealth of research on how to add sparsity constraints like low rank or group sparsity regularizers (Zhou et al., 2013a) or via suitable conjugate prior (Linderman and Adams, 2014). The ideas on the next section are applicable with some modification to the case that the aforementioned ideas are utilized to improve the naive solution of the maximum likelihood.

6. Properties of Simulated Co-evolution, Networks and Cascades

In this section, we perform an empirical investigation of the properties of the networks and information cascades generated by our model. In particular, we show that our model can generate co-evolutionary retweet and link dynamics and a wide spectrum of static and temporal network patterns and information cascades.

6.1 Simulation Settings

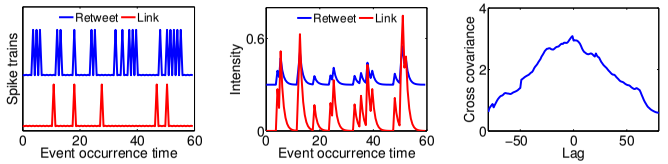
Throughout this section, if not said otherwise, we simulate the evolution of a 8,000-node network as well as the propagation of information over the network by sampling from our model using Algorithm 1. We set the exogenous intensities of the link and diffusion events to $\mu_u = \mu = 4 \times 10^{-6}$ and $\eta_u = \eta = 1.5$ respectively, and the triggering kernel parameter to $\omega_1 = \omega_2 = 1$. The parameter μ determines the independent growth of the network—roughly speaking, the expected number of links each user establishes spontaneously before time T is μT. Whenever we investigate a static property, we choose the same sparsity level of 0.001. The specific range of values we utilized in the current and the next section are such that the process remains stationary and doesn't blow up (Hawkes, 1971). Furthermore, the range of parameters are chosen such that all the results could be reported in a unified scale.

6.2 Retweet and Link Coevolution

Figures 8(a,b) visualize the retweet and link events, aggregated across different sources, and the corresponding intensities for one node and one realization, picked at random. Here, it is already apparent that retweets and link creations are clustered in time and often follow each other. Further, Figure 8(c) shows the cross-covariance of the retweet and link creation intensity, computed across multiple realizations, for the same node, *i.e.*, if $f(t)$ and $g(t)$ are two intensities, the cross-covariance is a function $h(\tau) = \int f(t+\tau)g(t)\,dt$. It can be seen that the cross-covariance has its peak around 0, *i.e.*, retweets and link creations are highly correlated and co-evolve over time. For ease of exposition, we illustrated co-evolution using one node, however, we found consistent results across nodes.

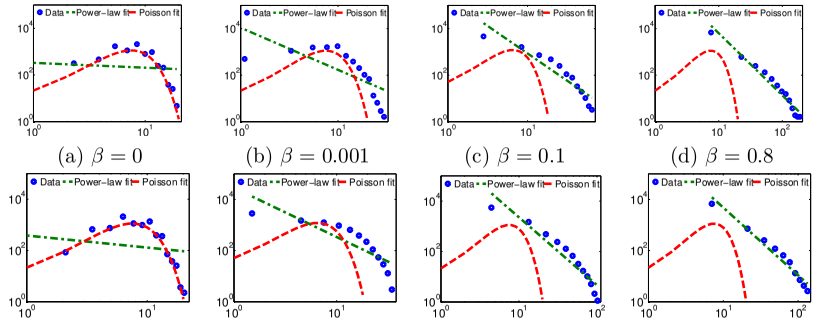
6.3 Degree Distribution

Empirical studies have shown that the degree distribution of online social networks and microblogging sites follow a power law (Chakrabarti et al., 2004; Kwak et al., 2010), and argued that it is a consequence of the rich get richer phenomena. The degree distribution of a network is a power law if the expected number of nodes m_d with degree d is given by $m_d \propto d^{-\gamma}$, where $\gamma > 0$. Intuitively, the higher the values of the parameters α and β, the closer the resulting degree distribution follows a power-law. This is because the network



(a) (b) (c)

Figure 8: Coevolutionary dynamics for synthetic data; (a) Spike trains of link and retweet events; (b) Link and retweet intensities with an exponentially decaying kernel; (c) Cross covariance of link and retweet intensities: The peak around 0 shows the coupling of retweet and link events.



(a) $\beta = 0$ (b) $\beta = 0.001$ (c) $\beta = 0.1$ (d) $\beta = 0.8$

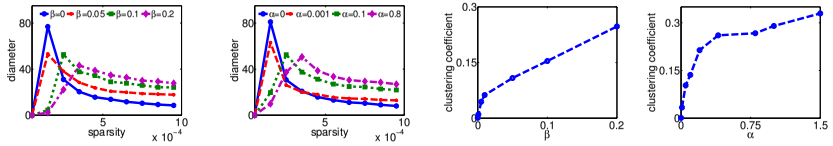
(a) $\alpha = 0$ (b) $\alpha = 0.05$ (c) $\alpha = 0.1$ (d) $\alpha = 0.2$

Figure 9: Degree distributions when network sparsity level reaches 0.001 for different β (α) values and fixed $\alpha = 0.1$ ($\beta = 0.1$) are shown in top (bottom) row. By varying the value of β (α) the degree distributions spans from random to scale-free networks illustrating the flexibility of the framework to model real-world networks.

grows more locally. Interestingly, the lower their values, the closer the distribution to an Erdos-Renyi random graph (Erdos and Rényi, 1960), because, the edges are added almost uniformly and independently without influence from the local structure. Figure 9 confirms this intuition by showing the degree distribution for different values of β and α.

6.4 Small (shrinking) Diameter

There is empirical evidence that the diameter of online social networks and microblogging sites exhibit relatively small diameter and shrinks (or flattens) as the network grows (Backstrom et al., 2012; Chakrabarti et al., 2004; Leskovec et al., 2005). Figures 10(a-b) show the diameter on the largest connected component (LCC) against the sparsity of the network



(a) Diameter, $\alpha = 0.1$ (b) Diameter, $\beta = 0.1$ (c) CC, $\alpha = 0.1$ (d) CC, $\beta = 0.1$

Figure 10: Diameter and clustering coefficient for network sparsity 0.001; (a,b) The diameter against sparsity over time for fixed $\alpha = 0.1$, and for fixed $\beta = 0.1$ respectively; (c,d) The clustering coefficient (CC) against β and α, respectively.

over time for different values of α and β. Although at the beginning, there is a short increase in the diameter due to the merge of small connected components, the diameter decreases as the network evolves. Moreover, larger values of α or β lead to higher levels of local growth in the network and, as a consequence, slower shrinkage. Here, nodes *arrive* to the network when they follow (or are followed by) a node in the largest connected component.

6.5 Clustering Coefficient

Triadic closure (Granovetter, 1973; Leskovec et al., 2008; Romero and Kleinberg, 2010) has been often presented as a plausible link creation mechanism. However, different social networks and microblogging sites present different levels of triadic closure (Ugander et al., 2013). Importantly, our method is able to generate networks with different levels of triadic closure, as shown by Figure 10(c-d), where we plot the clustering coefficient (Watts and Strogatz, 1998), which is proportional to the frequency of triadic closure, for different values of α and β.

6.6 Cascade Patterns

Our model can produce the most commonly occurring cascades structures as well as heavy-tailed cascade size and depth distributions, as observed in historical Twitter data (Goel et al., 2012). Figure 11 summarizes the results, which provide empirical evidence that the higher the α (β) value, the shallower and wider the cascades.

7. Experiments on Model Estimation and Prediction on Synthetic Data

In this section, we first show that our model estimation method can accurately recover the true model parameters from historical link and diffusion events data and then demonstrate that our model can accurately predict the network evolution and information diffusion over time, significantly outperforming two state of the art methods (Weng et al., 2013; Antoniades and Dovrolis, 2015; Myers and Leskovec, 2014) at predicting new links, and a baseline Hawkes process that does not consider network evolution at predicting new events.

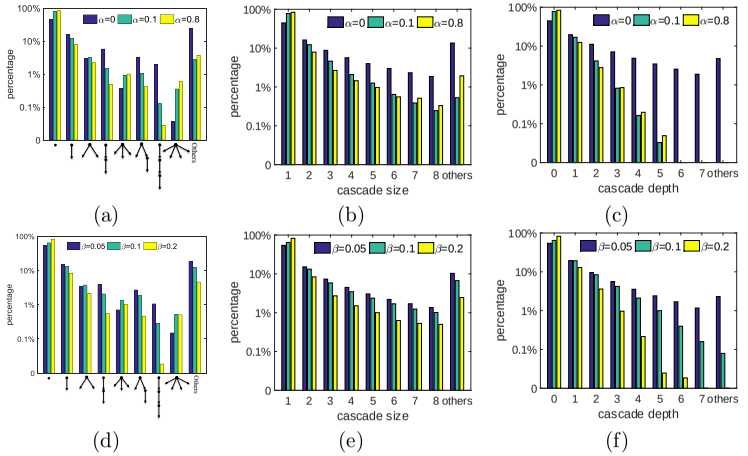


Figure 11: Distribution of cascade structure, size and depth for different values of α (β) and fixed $\beta = 0.2$ ($\alpha = 0.8$) on the top (bottom) row. The commom motifs of propagation are shown in (a) and (d).

7.1 Experimental Setup

Throughout this section, we experiment with our model considering m=400 nodes. We set the model parameters for each node in the network by drawing samples from $\mu \sim U(0, 0.0004)$, $\alpha \sim U(0, 0.1)$, $\eta \sim U(0, 1.5)$ and $\beta \sim U(0, 0.1)$. We then sample up to 60,000 link and information diffusion events from our model using Algorithm 1 and average over 8 different simulation runs.

7.2 Model Estimation

We evaluate the accuracy of our model estimation procedure via two measures: (i) the relative mean absolute error (*i.e.*, $\mathbb{E}[|x - \hat{x}|/x]$, MAE) between the estimated parameters (x) and the true parameters ($\hat{x}$), (ii) the Kendall's rank correlation coefficient between each estimated parameter and its true value, and (iii) test log-likelihood. Tables 1 and 2, and Figure 12 show the rank correlation, relative error, and the log likelihood on the test data, respectively. They show that as we feed more events into the estimation procedure, the estimation becomes more accurate.

7.3 Link Prediction

We use our model to predict the identity of the source for each test link event, given the historical events before the time of the prediction, and compare its performance with two

Table 1: Parameter Estimation Results (Rank Correlation)

# Events	α	β	η	μ
5×10^4	0.66 ± 0.05	0.82 ± 0.05	0.81 ± 0.05	0.72 ± 0.03
4×10^4	0.58 ± 0.03	0.79 ± 0.04	0.72 ± 0.05	0.65 ± 0.03
3×10^4	0.49 ± 0.05	0.76 ± 0.04	0.66 ± 0.04	0.58 ± 0.05
2×10^4	0.38 ± 0.05	0.65 ± 0.05	0.54 ± 0.05	0.47 ± 0.05
1×10^4	0.24 ± 0.03	0.43 ± 0.05	0.29 ± 0.04	0.26 ± 0.04

Table 2: Parameter Estimation Results (Relative Error)

# Events	α	β	η	μ
5×10^4	0.25 ± 0.05	0.1 ± 0.04	0.16 ± 0.05	0.18 ± 0.05
4×10^4	0.31 ± 0.05	0.17 ± 0.05	0.24 ± 0.03	0.28 ± 0.06
3×10^4	0.47 ± 0.04	0.32 ± 0.04	0.35 ± 0.04	0.39 ± 0.05
2×10^4	0.65 ± 0.05	0.49 ± 0.04	0.55 ± 0.04	0.57 ± 0.05
1×10^4	0.86 ± 0.05	0.71 ± 0.04	0.75 ± 0.06	0.79 ± 0.05

state of the art methods, which we denote as TRF (Antoniades and Dovrolis, 2015) and WENG (Weng et al., 2013).

Central to TRF model is the notion of TRF event: When a follower $L \in \mathcal{F}(R)$ receives a retweet of S through R, L can decide to follow S directly. They call this sequence a *Tweet-Retweet-Follow* (TRF) event, and refer to its three main actors as Speaker S, Repeater R, and Listener L. In addition to characterizing and identifying such events, TRF measures the probability of creating a link from a source at a given time by simply computing the proportion of new links created from the source over all total created links up to the given time. More specifically, it estimates the probability of new link as follows. Consider a tweet of Speaker S at time t_s . Suppose that this tweet is not retweeted by any of the followers of S in the period $[t_s, t_s + \Delta]$. Let $\Phi(S, t_s)$ be the set of followers of followers of S that are not directly following S at t_s. The exogenous probability of of following S is:

$$P_{EXO} = \frac{|L : L \in \Phi(S, t_s), L \in \mathcal{F}(S, t_s + \Delta|}{|\Phi(S, t_s)|}. \tag{35}$$

Similarly, consider a tweet of Speaker S at time t_s that has been retweeted by a follower of S, referred to as Repeater R, at time $t_r > t_s$. Let $\Phi_R(S, t_r)$ be the subset of $\Phi(S, t_r)$ that includes only followers of R. The endogenous probability of following S is:

$$P_{END} = \frac{|L : L \in \Phi_R(S, t_r), L \in \mathcal{F}(S, t_r + \Delta|}{|\Phi_R(S, t_r)|}. \tag{36}$$

And the probability of following S is simply $P_{EXO} + P_{END}$.

Weng et al. (2013) analyzed the complete graph and activity of Yahoo! Meme, to identify the effect of information diffusion on the evolution of the underlying network. WENG considers several link creation strategies and makes a prediction by combining these strategies. More specifically, they consider five link creation mechanisms and their combinations: Random: follow a randomly selected user who is not yet followed; Triadic closure: follow a randomly selected triadic node; Grandparent: follow a randomly selected grandparent;

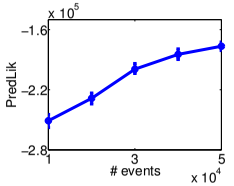
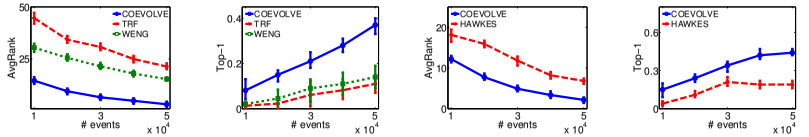


Figure 12: Performance of model estimation for the synthetic network reported in Log-likelihood on unseen test data.



(a) Links: AR (b) Links: Top-1 (c) Activity: AR Activity: Top-1

Figure 13: Prediction performance of future links and activities for a 400-node synthetic network on the test unseen data. The accuracy is reported by means of average rank (AR) and success probability that the true events rank among the top-1 events (Top-1).

Origin: follow a randomly selected origin; Traffic shortcut: follow a randomly selected grandparent or origin. Then, they parameterize the the probability of using a strategy and learned the it using maximum likelihood. Because of the contribution of multiple terms in the log-likelihood expression (due to the mixed effects of strategies) they numerically explore the values of parameters in the unit square to maximize the log-likelihood.

Here, we evaluate the performance by computing the probability of all potential links using our model, TRF and WENG. In our model, the intensity gives us a measure of how likely a link will be created at a specific point of time. The higher the intensity, the more likely the connection. Therefore, we can sort the future events based on the likelihood of happening. Then, we compute (i) the average rank of all true (test) events in our sorted list (AvgRank) and, (ii) the success probability that the true (test) events rank among the top-1 potential events at each test time (Top-1). Figure 13 summarizes the results, where we trained our model with an increasing number of events. Our model outperforms both TRF and WENG for a significant margin.

7.4 Activity Prediction

We use our model to predict the identity of the node that generates each test diffusion event, given the historical events before the time of the prediction, and compare its performance with a baseline consisting of a multi-dimensional Hawkes process without network evolution

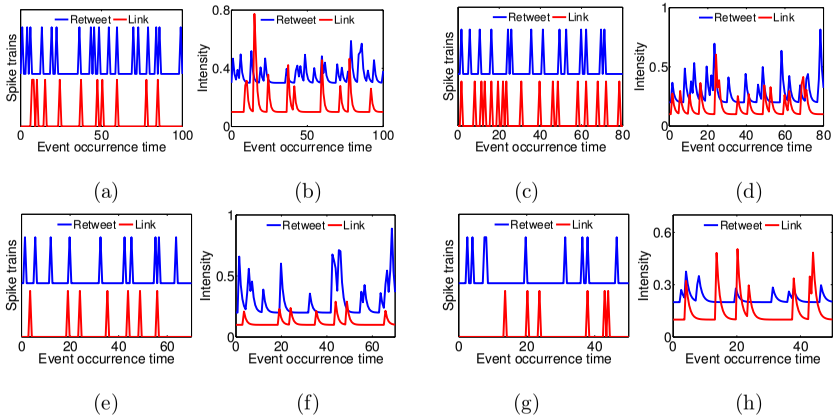


Figure 14: Link and retweet behavior of 4 typical users in the real-world data set. Panels (a,c,e,g) show the spike trains of link and retweet events and Panels (b,d,f,h) show the estimated link and retweet intensities with exponential decaying kernel.

part. For the Hawkes baseline, we take a snapshot of the network right before the prediction time, and use all historical retweeting events to fit the model and finding $(a_{uv})_{u,v\in[m]}$ and $(\eta_u)_{u\in[m]}$ similar to previous works (Farajtabar et al., 2014). Here, each user u performs activity with intensity

$$\gamma_u(t) = \eta_u + \sum_{v\in\mathcal{F}_u(t)} a_{uv}\ \kappa_{\omega_1}(t) \star dN_v(t). \tag{37}$$

Now, we have two Hawkes process based models, Coevolve, which considers information propagation coevolved with the link creation process, and the baseline Hawkes model which only contains the information propagation part to model activities. Similar to the link prediction we used the intensity to measure how likely is an event. We can sort future events according to their probability. Then, we evaluate the performance via the same two measures as in the link prediction task and summarize the results in Figure 13 against an increasing number of training events. Please refer to the previous subsection to learn about the measures used here. The results show that, by modeling the network evolution, our model performs significantly better than the baseline.

8. Experiments on Coevolution and Prediction on Real Data

In this section, we validate our model using a large Twitter data set containing nearly 550,000 tweet, retweet and link events from more than 280,000 users (Antoniades and Dovrolis, 2015). We will show that our model can capture the co-evolutionary dynamics

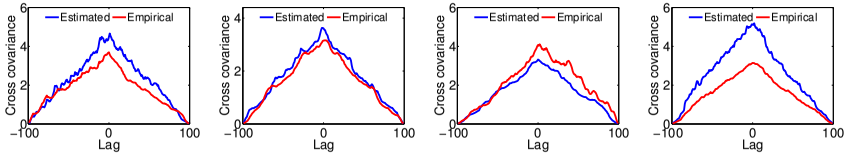


Figure 15: Empirical and simulated cross covariance of link and retweet intensities for four typical users.

and, by doing so, it predicts retweet and link creation events more accurately than several alternatives.

8.1 Data Set Description & Experimental Setup

We use a data set that contains both link events as well as tweets/retweets from millions of Twitter users (Antoniades and Dovrolis, 2015). In particular, the data set contains data from three sets of users in 20 days; nearly 8 million tweet, retweet, and link events by more than 6.5 million users. The first set of users (8,779 users) are source nodes s, for whom all their tweet times were collected. The second set of users (77,200 users) are the followers of the first set of users, for whom all their retweet times (and source identities) were collected. The third set of users (6,546,650 users) are the users that start following at least one user in the first set during the recording period, for whom all the link times were collected.

In our experiments, we focus on all events (and users) during a 10-day period (Sep 21 2012 - 30 Sep 2012) and used the information before Sep 21 to construct the initial social network (original links between users). We model the co-evolution in the second 10-day period using our framework. More specifically, in the coevolution modeling, we have 5,567 users in the first layer who post 221,201 tweets. In the second layer 101,465 retweets are generated by the whole 77,200 users in that interval. And in the third layer we have 198,518 users who create 219,134 links to 1978 users (out of 5567) in the first layer.

We split events into a training set (covering 85% of the retweet and link events) and a test set (covering the remaining 15%) according to time, *i.e.*, all events in the training set occur earlier than those in the test set. We then use our model estimation procedure to fit the parameters from an increasing proportion of events from the training data.

8.2 Retweet and Link Coevolution

Figures 14 visualizes the retweet and link events, aggregated across different targets, and the corresponding intensities given by our trained model for four source nodes, picked at random. Here, it is already apparent that retweets (of his posts) and link creations (to him) are clustered in time and often follow each other, and our fitted model intensities successfully track such behavior. Further, Figure 15 compares the cross-covariance between the empirical retweet and link creation intensities and between the retweet and link creation intensities given by our trained model, computed across multiple realizations, for the same nodes. For all nodes, the similarity between both cross-covariances is striking and both has

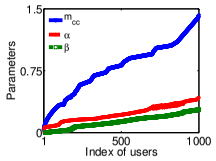


Figure 16: Empirical cross covariance and learned model parameters for 1,000 users, picked at random. The x-axis indexes the users while their α, β and the empirical cross covariance are shown in different colors on the y-axis.

their peak around 0, *i.e.*, retweets and link creations are highly correlated and co-evolve over time. For ease of exposition, as in Section 6, we illustrated co-evolution using four nodes, however, we found consistent results across nodes.

To further verify that our model can capture the coevolution, we compute the average value of the empirical cross covariance function, denoted by m_{cc}, per user. Intuitively, one could expect that our model estimation method should assign higher α and/or β values to users with high m_{cc}. Figure 16 confirms this intuition on 1,000 users, picked at random. Whenever a user has high α and/or β value, she exhibits a high cross covariance between her created links and retweets.

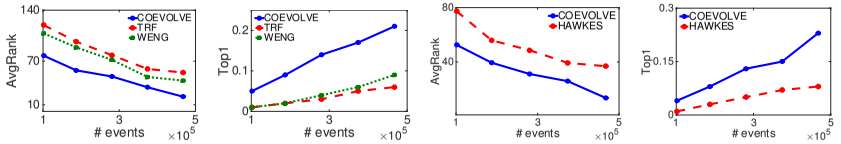
8.3 Link Prediction

We use our model to predict the identity of the source for each test link event, given the historical (link and retweet) events before the time of the prediction, and compare its performance with the same two state of the art methods as in the synthetic experiments, TRF (Antoniades and Dovrolis, 2015) and WENG (Weng et al., 2013).

We evaluate the performance by computing the probability of all potential links using different methods, and then compute (i) the average rank of all true (test) events (AvgRank) and, (ii) the success probability (SP) that the true (test) events rank among the top-1 potential events at each test time (Top-1). We summarize the results in Figure 17(a-b), where we consider an increasing number of training retweet/tweet events. Our model outperforms TRF and WENG consistently. For example, for $8 \cdot 10^4$ training events, our model achieves a SP 2.5x times larger than TRF and WENG.

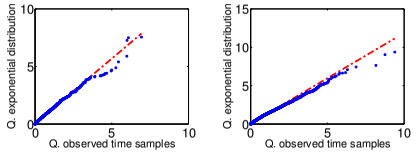
8.4 Activity Prediction

We use our model to predict the identity of the node that generates each test diffusion event, given the historical events before the time of the prediction, and compare its performance with a baseline consisting of a Hawkes process without network evolution. For the Hawkes baseline, we take a snapshot of the network right before the prediction time, and use all historical retweeting events to fit the model. Here, we evaluate the performance the via the same two measures as in the link prediction task and summarize the results in Figure 17(c-d) against an increasing number of training events. The results show that, by modeling the co-evolutionary dynamics, our model performs significantly better than the baseline.



(a) Links: AR (b) Links: Top-1 (c) Activity: AR Activity: Top-1

Figure 17: Prediction performance in the Twitter data set by means of average rank (AR) and success probability that the true (test) events rank among the top-1 events (Top-1).



(a) Link process (b) Retweet process

Figure 18: Quantile plots of the intensity integrals from the real link and retweet event time. The alignment of blue dots (empirical) and red ones (theoretical) shows the suitability of our model in representing observed traces of activities in Twitter.

8.5 Model Checking

Given all the subsequent event times generated using a Hawkes process, *i.e.*, t_i and t_{i+1}, according to the time changing theorem (Daley and Vere-Jones, 2007), the intensity integrals $\int_{t_i}^{t_{i+1}} \lambda(t)\, dt$ should conform to the unit-rate exponential distribution. Figure 18 presents the quantiles of the intensity integrals computed using intensities with the parameters estimated from the real Twitter data against the quantiles of the unit-rate exponential distribution. It clearly shows that the points approximately lie on the same line, giving empirical evidence that a Hawkes process is the right model to capture the real dynamics.

9. Related Work

In this section, we survey related works in modeling temporal networks followed by a subsection on co-evolution dynamics. Next, we review the literature on information diffusion models. Finally, we conclude this section by works that are closely related and are developed for almost the same goal.

Temporal Networks. Much effort has been devoted to modeling the evolution of social networks (Phan and Airoldi, 2015; Doreian and Stokman, 2013; Wang et al., 2011). Of the proposed methods in characterizing link creation, triadic closure (Granovetter, 1973) is a simple but powerful principle to model the evolution based on shared friends. Modeling timing and rich features of social interactions has been attracting increasing interest

in the social network modeling community (Goldenberg et al., 2010). However, most of these models use timing information as discrete indices. The dynamics of the resulting time-discretized model can be quite sensitive to the chosen discretization time steps; Too coarse a discretization will miss important dynamic features of the process, and too fine a discretization will increase the computational and inference costs of the algorithms. In contrast, the events we try to model tend to be asynchronous with a number of different time scales. Recently, Snijders and Luchini (2006) used a Cox-intensity Poisson model with exponential random graphs to model friendship dynamics. Brandes et al. (2009) extended this model to the temporal sequence of interactions that take place in the social network, but with insufficient model flexibility, and limited scalability. However, these works largely fail to model the interdependency between events generated by different users, which is one of the focuses of our proposed framework. Most of this line of work is summarized in a recent survey (Holme, 2015), with a short section devoted to point process based approaches.

Co-evolution Dynamics. In machine learning and several other communities, both the dynamics on the network and the dynamics of the network have been extensively studied, and combining the two is a natural next step. For example, Bhattacharya et al. (2015) claimed that content generation in social networks is influenced not just by their personal features like age and gender, but also by their social network structure. Furthermore, research has been done to address the co-evolution problems, for example, in the complex network literature, under the name of *adaptive system* (Gross and Blasius, 2008). The main premise is that the evolution of the topology depends on the dynamics of the nodes in the network, and a feedback loop can be created between the two, which allows dynamical exchange of information. In a different context, epidemiologists have found that nodes may rewire their links to try to avoid contact with the infected ones (Gross et al., 2006). Co-evolutionary models have been also developed for collective opinion formation, investigating whether the coevolutionary dynamics will eventually lead to consensus or fragmentation of the population (Zschaler et al., 2012). However, this line of research tends to be less data-driven.Moreover, although the general nonlinear dynamic-system based methods usually address co-evolutionary phenomena that are macroscopic in nature, they lack the inference power of statistical generative models which are more adapted to teasing out microscopic details from the data. Finally, we would also like to mention a different line of research exemplified by the actor-oriented models developed by Snijders (2014), where a continuous-time Markov chain on the space of directed networks is specified by local node-centric probabilistic link change rules, and MCMC and method of moments are used for parameter estimation. Hawkes processes we used are generally non-Markovian and making use of event history far into the past.

Information Diffusion. The presence of timing information in event data and the ability to model such information bring up the interesting question of how to use the learned model for time-sensitive inference or decision making. Furthermore, the development of online social networks has attracted a lot of empirical studies of the online influence patterns of online communities (Guo et al., 2015), micro blogs (Bakshy et al., 2011) and so on. However, these works usually consider only relatively simple models for the influence, which may not be very predictive. For more mathematically oriented works, based on information cascades (a special case of asynchronous event data) from social networks, discrete-time

diffusion models have been fitted to the cascades (Goyal et al., 2010a) and used for decision making, such as identifying influencer (Agarwal et al., 2008), maximizing information spread (Rodriguez and Schölkopf, 2012), and marketing planing (Richardson and Domingos, 2002; Bhagat et al., 2012). Several recent experimental comparisons on both synthetic and real world data showed that continuous-time models yield significant improvement in settings such as recovering hidden diffusion network topologies from cascade data (Du et al., 2012; Gomez-Rodriguez et al., 2011; Yang and Zha, 2013; Zarezade et al., 2015), predicting the timings of future events (Rodriguez et al., 2013), finding source of information cascades (Farajtabar et al., 2015a). Besides this, Point process modeling of activity in network is becoming increasingly popular (Parikh et al., 2012; Hall and Willett, 2016; Farajtabar et al., 2017). These time-sensitive modeling and decision making problems can usually be framed into optimization problems and are usually difficult to solve. This brings up interesting optimization problems, such as efficient submodular function optimization with provable guarantees (Goyal et al., 2010b; Kempe et al., 2003), sampling methods (Lian et al., 2014; Gunawardana et al., 2011) for inference and prediction, and convex framework proposed in (Farajtabar et al., 2014; Karimi et al., 2016) to make decisions to shape the activity to a variety of objectives and intervene in networks (Farajtabar et al., 2017). Furthermore, the high dimensional nature of modern event data makes the evaluation of objective function of the optimization problem even more expensive. Therefore, more accurate modeling and sophisticated algorithm needed to be designed to tackle the challenges posed by modern event data applications.

The work most closely related to ours is the empirical study of information diffusion and network evolution (Gross and Blasius, 2008; Singer et al., 2012; Weng et al., 2013; Antoniades and Dovrolis, 2015; Myers and Leskovec, 2014). Among them, (Weng et al., 2013) was the first to show experimental evidence that information diffusion influences network evolution in microblogging sites both at system-wide and individual levels. In particular, they studied *Yahoo! Meme*, a social micro-blogging site similar to Twitter, which was active between 2009 and 2012, and showed that the likelihood that a user u starts following a user s increases with the number of messages from s seen by u. Antoniades and Dovrolis (2015) investigated the temporal and statistical characteristics of retweet-driven connections within the Twitter network and then identified the number of retweets as a key factor to infer such connections, in agreement with (Weng et al., 2013). Besides link prediction, they argue about which potential network motifs TRF events can lead to in the network structure. Myers and Leskovec (2014) showed that the Twitter network can be characterized by steady rates of change, interrupted by sudden bursts of new connections, triggered by retweet cascades. They also developed a method to predict which retweets are more likely to trigger these bursts. Finally, Tran et al. (2015) utilized multivariate Hawkes process to establish a connection between temporal properties of activities and the structure of the network. In contrast to our work they studied the static properties, *e.g.*, community structure and inferred the latent clusters using the observed activities.

However, there are fundamental differences between the above-mentioned studies and our work. First, they only characterize the effect that information diffusion has on the network dynamics, but not the bidirectional influence. In contrast, our probabilistic generative model takes into account the bidirectional influence between information diffusion and network dynamics. Second, previous studies are mostly empirical and only make binary

predictions on link creation events. For example, the work of (Weng et al., 2013; Antoniades and Dovrolis, 2015) predict whether a new link will be created based on the number of retweets; and, (Myers and Leskovec, 2014) predict whether a burst of new links will occur based on the number of retweets and users' similarity. However, our model is able to learn parameters from real world data, and predict the precise timing of both diffusion and new link events.

10. Conclusion and Future Works

In this work, we proposed a joint continuous-time model of information diffusion and network evolution, which can capture the coevolutionary dynamics, can mimic the most common static and temporal network patterns observed in real-world networks and information diffusion data, and can predict the network evolution and information diffusion more accurately than previous state-of-the-arts. Using point processes to model intertwined events in information and social networks opens up many interesting venues for future. Our current model is just a show-case of a rich set of possibilities offered by a point process framework, which have been rarely explored before in large scale social network modeling. There are quite a few directions that remain as future work and are very interesting to explore. For example:

- A large and diverse range of point processes can also be used instead in the framework and augment the current model without changing the efficiency of simulation and the convexity of parameter estimation.
- We can incorporate features from previous state of the diffusion or network structure. For example, one can model information overload by adding a nonlinear transfer function on top of the diffusion intensity, or model peer pressure by adding a nonlinear transfer function depending on the number of neighbors.
- There are situations that the processes are naturally evolve in different time scales. For example, link dynamics is meaningful in the scale of days, however, the resolution in which information propagation occurs is usually in hours or even minutes. Developing an efficient mechanism to account for heterogeneity in time resolution would improve the model's ability to predict.
- We may augment the framework to allow time-varying parameters. The simulation would not be affected and the estimation of time-varying interaction can still be carried out via a convex optimization problem (Zhou et al., 2013b).
- Alternatively, one can use different triggering kernels for the Hawkes processes and learn them to capture finer details of temporal dynamics.

Acknowledgments

The authors would like to thank Demetris Antoniades and Constantine Dovrolis for providing them with the data set. The research was supported in part by NSF/NIH BIGDATA 1R01GM108341, ONR N00014-15-1-2340, NSF IIS-1218749, NSF CAREER IIS-1350983.

Appendix A. Notations

Table 3: Table of Notations of the COEVOLVE framework

Notation	Description
m	number of nodes in the network
$e^r = (u, s, t)$	retweet event: node u retweets at time t a tweet by source node s
$e^l = (u, s, t)$	link event: node u create at time t a link to source node s
$\mathcal{H}^r_{us}(t)$	history of retweet events by node u up to time t due to source s
$\mathcal{H}^r(t)$	history of all retweet events up to time t
$\mathcal{H}^l(t)$	history of all link events up to time t
$N_{us}(t)$	count process of the number of retweets of u due to source s up to time t
$\boldsymbol{N}(t)$	count process matrix of all pairs $\boldsymbol{N}(t) := (\ N_{us}(t)\ )_{u,s\in[m]}$
$d\boldsymbol{N}(t)$	differential of retweet process matrix $d\boldsymbol{N}(t) := (\ dN_{us}(t)\ )_{u,s\in[m]}$
$A_{us}(t)$	survival process indicating whether u is directly following s at time t
$\boldsymbol{A}(t)$	survival process matrix of all paris $\boldsymbol{A}(t) := (\ A_{us}(t)\ )_{u,s\in[m]}$
$d\boldsymbol{A}(t)$	differential of link creation process matrix $d\boldsymbol{A}(t) := (\ dA_{us}(t)\ )_{u,s\in[m]}$
$\gamma^*_{us}(t)$	conditional intensity function of user u retweet an event due to source s
$\boldsymbol{\Gamma}^*(t)$	matrix of information diffusion intensity functions $\Gamma(t) = (\ \gamma^*_{us}(t)\ )_{u,s\in[m]}$
$\mathbb{I}[\cdot]$	indicator function
$\mathcal{F}_u(t)$	set of followers of u at time t
η_u	intensity of original tweets by a user u on his own initiative
β_s	amount of increase in retweeting intensity when a tweet by source s hits
$\kappa_\omega(t)$	decaying kernel with bandwidth ω; $\kappa_\omega(t) = e^{-\omega t}$
$\lambda^*_{us}(t)$	conditional intensity function for user u follows source s at time t
$\boldsymbol{\Lambda}^*(t)$	matrix of link creation intensity functions $\boldsymbol{\Lambda}^*(t) = (\ \lambda^*_{us}(t)\ )_{u,s\in[m]}$
μ_u	intensity of node u deciding to follow a source at her own initiative
α_u	amount of increase in linking intensity when a tweet by the source hits
$\mathcal{E}$	collection of retweet events observed
$\mathcal{A}$	collection of link creation events
$\mathfrak{L}$	likelihood of observation
ν_{ij}	auxiliary variable for lower-bounding the log-likelihood

Appendix B. Basic Simulation Algorithm

In this section, we revisit Ogata's algorithm in more details. Consider a U-dimensional point process in which each dimension u is characterized by a conditional intensity function $\lambda_u^*(t)$.

Ogata's algorithm starts with summing the intensities, $\lambda_{sum}^*(\tau) = \sum_{u=1}^{U} \lambda_u^*(\tau)$. Then, assuming we have simulated up to time t, the next sample time, t', is the first event drawn from the non-homogenous Poisson process with intensity $\lambda_{sum}^*(\tau)$ which begins at time t. Here, the algorithm exploits that, given a fixed history, the Hawkes Process is a non-homogenous Poisson process, which runs until the next event happens. Then, the new event will result in an update of the intensities and a new non-homogenous Poisson process starts until the next event.

It can be shown that the waiting time of a non-homogeneous Poisson process is an exponentially distributed random variable with rate equal to integral of the intensity (Ross, 2011), *i.e.* $s \sim Exponential\left(\int_t^{t+s} \lambda_{sum}^*(\tau)\, d\tau\right)$. Thus, the next sample time can be computed as

$$t' = \underbrace{t}_{\text{current time}} + \underbrace{s.}_{\text{waiting time for the first event}} \tag{38}$$

Sampling from a non-homogenous Poisson process is not straight-forward, therefore, Ogata's algorithm uses rejection sampling with a homogenous Poisson process as the proposal distribution. More in detail, given $\hat{\lambda} = \max_{t \le \tau \le T} \lambda_{sum}^*(\tau)$, t' is the time of first event of homogenous Poisson Process with rate $\hat{\lambda}$. Then, we accept the sample time with probability $\lambda_{sum}^*(t')/\hat{\lambda}$. Finally, the dimension firing the event is determined by sampling proportionally to the contribution of the intensity of that user to the total intensity, *i.e.*, $\lambda_u^*(t')/\lambda_{sum}^*(t')$ for $1 \le u \le U$. This procedure is iterated until we reach the end of simulation time T. Algorithm 5 presents the complete procedure.

Ogata's algorithm would scale poorly with the dimension of the process, because, after each sample, we would need to re-evaluate the affected intensities and find the upper bound. As a consequence, a naive implementation to draw n samples require $O(Un^2)$ time complexity, where U is the number of dimensions. This is because for each sample we need to find the new summation of intensities, which involves $O(U)$ individual ones, each taking $O(n)$ time to accumulate over this history. In our social networks application, we have $m^2 - m$ point processes for link creation and m^2 ones for retweeting, *i.e.*, $U = O(m^2)$. Therefore, Ogata's algorithm takes $O(m^2 n^2)$ time complexity.

Algorithm 5 Ogata's Algorithm

Input: U dimensional Hawkes process $\{\lambda_u^*(t)\}_{u=1\ldots U}$, Due time: T
Output: Set of events: $\mathcal{H} = \{(t_1, u_1), \ldots, (t_n, u_n)\}$
$t \leftarrow 0$
$i \leftarrow 0$
while $t < T$ **do**
  $\lambda_{sum}^*(\tau) \leftarrow \sum_{u=1}^{U} \lambda_u^*(\tau)$ — Sampling next event time
  $\hat{\lambda} \leftarrow \max_{t \le \tau \le T} \lambda_{sum}^*(\tau)$ — Sampling next event time
  $s \sim Exponential(\hat{\lambda})$ — Sampling next event time
  $t' \leftarrow t + s$ — Sampling next event time
  if $t' \ge T$ **then** — Sampling next event time
    break — Sampling next event time
  end if — Sampling next event time
  $\bar{\lambda} \leftarrow \lambda_{sum}^*(t')$ — Rejection test
  $d \sim Uniform(0, 1)$ — Rejection test
  if $d \times \hat{\lambda} > \bar{\lambda}$ **then** — Rejection test
    $t \leftarrow t'$ — Rejection test
    Goto 6 — Rejection test
  end if — Rejection test
  $S \leftarrow 0$ — Attribution test
  $d \sim Uniform(0, 1)$ — Attribution test
  for $u \leftarrow 1$ to U **do** — Attribution test
    $S \leftarrow S + \lambda_u^*(t')$ — Attribution test
    if $S \ge d$ **then** — Attribution test
      $i \leftarrow i + 1$ — Attribution test
      $u_i \leftarrow u$ — Attribution test
      $t_i \leftarrow t'$ — Attribution test
      $t \leftarrow t'$ — Attribution test
      Goto 6 — Attribution test
    end if — Attribution test
  end for — Attribution test
  Given the new event just sampled update intensity functions $\lambda_u^*(\tau)$
end while

Appendix C. Proofs

Lemma 1 *Assume we have U independent non-homogeneous Poisson processes with intensity $\lambda_1^*(\tau), \ldots, \lambda_U^*(\tau)$. Take random variable τ_u equal to the time of process u's first event after time t. Define $\tau_{min} = \min_{1\le u\le U}\{\tau_u\}$ and $u_{min} = \operatorname{argmin}_{1\le u\le U}\{\tau_u\}$. Then,*

(a) τ_{min} is the first event after time t of the Poisson process with intensity $\lambda_{sum}^*(\tau)$. In other words, τ_{min} has the same distribution as the next event (t') in Ogata's algorithm.

(b) u_{min} follows the conditional distribution $\mathbb{P}(u_{min} = u|\tau_{min} = x) = \frac{\lambda_U^*(x)}{\lambda_{sum}^*(x)}$. I.e. the dimension firing the event comes from the same distribution as the one in Ogata's algorithm.

Proof (a) The waiting time of the first event of a dimension u is exponentially distributed[1] random variable (Ross, 2011); *i.e.*, $\tau_u - t \sim Exponential\left(\int_t^{t+\tau_u}\lambda_u^*(\tau)\,d\tau\right)$. We have:

$$\begin{aligned}\mathbb{P}(\tau_{min}\le x|x>t) &= 1-\mathbb{P}(\tau_{min}>x|x>t) = 1-\mathbb{P}(\min(\tau_1,\ldots,\tau_U)>x|x>t)\\ &= 1-\mathbb{P}(\tau_1>x,\ldots,\tau_U>x|x>t) = 1-\prod_{u=1}^{U}\mathbb{P}(\tau_u>x|x>t)\\ &= 1-\prod_{u=1}^{U}\exp\left(-\int_t^{t+x}\lambda_u^*(\tau)\,d\tau\right) = 1-\exp\left(-\int_t^{t+x}\lambda_{sum}^*(\tau)\,d\tau\right).\end{aligned} \tag{39}$$

Therefore, $\tau_{min}-t$ is exponentially distributed with parameter $\int_t^{\tau_{min}}\lambda_{sum}^*(\tau)\,d\tau$ which can be seen as the first event of a non-homogenous poisson process with intensity $\lambda_{sum}^*(\tau)$ after time t.

(b) To find the distribution of u_{min} we have

$$\begin{aligned}\mathbb{P}(u_{min}=u|\tau_{min}=x) &= \lambda_u^*(x)\exp\left(-\int_t^{t+x}\lambda_u^*(\tau)\,d\tau\right)\prod_{v\ne u}\exp\left(-\int_t^{t+x}\lambda_v^*(\tau)\,d\tau\right)\\ &= \lambda_u^*(x)\prod_{v}\exp\left(-\int_t^{t+x}\lambda_v^*(\tau)\,d\tau\right).\end{aligned} \tag{40}$$

After normalization we get $\mathbb{P}(u_{min}=u|\tau_{min}=x) = \frac{\lambda_U^*(x)}{\lambda_{sum}^*(x)}$. ∎

Theorem 2 *The sequence of samples from Ogata's algorithm and our proposed algorithm follow the same distribution.*

Proof Using the chain rule the probability of observing $\mathcal{H}_T = \{(t_1,u_1),\ldots,(t_n,u_n)\}$ is written as:

$$\mathbb{P}\{(t_1,u_1),\ldots,(t_n,u_n)\} = \prod_{i=1}^{n}\mathbb{P}\{(t_i,u_i)|(t_{i-1},u_{i-1}),\ldots,(t_1,u_1)\} = \prod_{i=1}^{n}\mathbb{P}\{(t_i,u_i)|\mathcal{H}_{t_i}\}.$$

By fixing the history up to some time, say t_i, all dimensions of multivariate Hawkes process become independent of each other (until next event happens). Therefore, the above lemma can be applied to show that the next sample time from Ogata's algorithm and the proposed one come from the same distribution, *i.e.*, for every i, $\mathbb{P}\{(t_i,u_i)|\mathcal{H}_{t_i}\}$ is the same for both algorithms. Thus, the multiplication of individual terms is also equal for both. This will prove the theorem. ∎

1. If random variable X is exponentially distributed with parameter r, then $f_X(x) = r\exp(-rx)$ is its probability distribution function and $F_X(x) = 1-\exp(-rx)$ is the cumulative distribution function.

Theorem 3 *The optimization problem defined by Equation (25) is jointly convex.*

Proof We expand the likelihood by replacing the intensity functions into Equation (24):

$$\begin{aligned}\mathfrak{L} = &\sum_{e_i^r\in\mathcal{E}}\log\left(\mathbb{I}[u_i=s_i]\,\eta_{u_i}+\mathbb{I}[u_i\ne s_i]\,\beta_{s_i}\sum\nolimits_{v\in\mathcal{F}_{u_i}(t_i)}\left(\kappa_{\omega_1}(t)\star(A_{u_iv}(t)\,dN_{vs_i}(t))\right)\Big|_{t=t_i}\right)\\ &-\sum_{u,s\in[m]}\mathbb{I}[u=s]\,\eta_u\int_0^T dt+\mathbb{I}[u\ne s]\,\beta_s\sum\nolimits_{v\in\mathcal{F}_u(t)}\int_0^T\kappa_{\omega_1}(t)\star(A_{uv}(t)\,dN_{vs}(t))\,dt\\ &+\sum_{e_i^l\in\mathcal{A}}\log\left(\mu_{u_i}+\alpha_{u_i}\sum_{v\in\mathcal{F}_{u_i}(t_i)}\left(\kappa_{\omega_2}(t)\star dN_{vs}(t)\right)\Big|_{t=t_i}\right)\\ &-\sum_{u,s\in[m]}\mu_u\int_0^T(1-A_{us}(t))\,dt+\alpha_u\int_0^T(1-A_{us}(t))\Big(\sum_{v\in\mathcal{F}_u(t)}\kappa_{\omega_2}(t)\star dN_{vs}(t)\Big)\,dt.\end{aligned} \tag{41}$$

If we stack all parameters in a vector $\boldsymbol{x} = (\{\mu_u\},\{\alpha_u\},\{\eta_u\},\{\beta_s\})$, one can easily notice that the log-likelihood $\mathfrak{L}$ can be written as $\sum_j\log(\boldsymbol{a}_j^\top\boldsymbol{x}) - \sum_k\boldsymbol{b}_k^\top\boldsymbol{x}$, which is clearly a concave function with respect to $\boldsymbol{x}$ (Boyd and Vandenberghe, 2004), and thus $-\mathfrak{L}$ is convex. Moreover, the constraints are linear inequalities and thus the domain is a convex set. This completes the proof for convexity of the optimization problem. ∎

Lemma 4 *The log-likelihood in Equation (24) is lower-bounded as follows:*

$$\begin{aligned}\mathfrak{L}\ge\mathfrak{L}' = &\sum_{e_i^r\in\mathcal{E}}\mathbb{I}[u_i=s_i]\,\log(\eta_{u_i})+\sum_{e_i^r\in\mathcal{E}}\mathbb{I}[u_i\ne s_i]\,\log(\beta_{s_i})\\ &+\sum_{e_i^r\in\mathcal{E}}\mathbb{I}[u_i\ne s_i]\,\log\Big(\sum\nolimits_{v\in\mathcal{F}_{u_i}(t_i)}\left(\kappa_{\omega_1}(t)\star(A_{u_iv}(t)\,dN_{vs_i}(t))\right)\Big|_{t=t_i}\Big)\\ &-\sum_{u,s\in[m]}\eta_uT+\beta_s\sum\nolimits_{v\in\mathcal{F}_u(t)}\int_0^T\kappa_{\omega_1}(t)\star(A_{uv}(t)\,dN_{vs}(t))\,dt\\ &+\sum_{e_i^l\in\mathcal{A}}\nu_{i1}\log(\mu_{u_i})+\nu_{i2}\log(\alpha_{u_i})+\nu_{i2}\log\Big(\sum_{v\in\mathcal{F}_{u_i}(t_i)}\left(\kappa_{\omega_2}(t)\star dN_{vs}(t)\right)\Big|_{t=t_i}\Big)\\ &-\sum_{e_i^l\in\mathcal{A}}\nu_{i1}\log(\nu_{i1})+\nu_{i2}\log(\nu_{i2})\\ &-\sum_{u,s\in[m]}\mu_u\int_0^T(1-A_{us}(t))\,dt+\alpha_u\int_0^T(1-A_{us}(t))(\kappa_{\omega_2}(t)\star dN_{us}(t))\,dt.\end{aligned} \tag{42}$$

Proof We can lower-bound the logarithm in the log-likelihood (Equation (41)) using Jensen's inequality as follows:

$$\begin{aligned}
&\log\left(\mu_{u_i} + \alpha_{u_i} \sum_{v\in\mathcal{F}_{u_i}(t_i)} \big(\kappa_{\omega_2}(t) \star dN_{vs}(t)\big)\big|_{t=t_i}\right) \\
&= \log\left(\nu_{i1}\frac{\mu_{u_i}}{\nu_{i1}} + \nu_{i2}\frac{\alpha_{u_i}}{\nu_{i2}} \sum_{v\in\mathcal{F}_{u_i}(t_i)} \big(\kappa_{\omega_2}(t) \star dN_{vs}(t)\big)\big|_{t=t_i}\right) \\
&\geq \nu_{i1}\log\left(\frac{\mu_{u_i}}{\nu_{i1}}\right) + \nu_{i2}\log\left(\frac{\alpha_{u_i}}{\nu_{i2}} \sum_{v\in\mathcal{F}_{u_i}(t_i)} \big(\kappa_{\omega_2}(t) \star dN_{vs}(t)\big)\big|_{t=t_i}\right) \\
&\geq \nu_{i1}\log(\mu_{u_i}) + \nu_{i2}\log(\alpha_{u_i}) + \nu_{i2}\log\left(\sum_{v\in\mathcal{F}_{u_i}(t_i)} \big(\kappa_{\omega_2}(t) \star dN_{vs}(t)\big)\big|_{t=t_i}\right) \\
&\quad - \nu_{i1}\log(\nu_{i1}) - \nu_{i2}\log(\nu_{i2}).
\end{aligned} \tag{43}$$

■

Replacing the logarithm with the lower-bound gets the result.

Appendix D. More Properties of the Coevolution Model

In this section we complement the properties of simulated coevolved networks and cascades by visualizing the outcomes, *i.e.*, the networks and cascades.

D.1 Network Visualization

Figure 19 visualizes several snapshots of the largest connected component (LCC) of two 300-node networks for two particular realizations of our model, under two different values of β. In both cases, we used $\mu = 2 \times 10^{-4}$, $\alpha = 1$, and $\eta = 1.5$. The top two rows correspond to $\beta = 0$ and represent one end of the spectrum, *i.e.*, Erdos-Renyi random network. Here, the network evolves uniformly. The bottom two rows correspond to $\beta = 0.8$ and represent the other end, *i.e.*, scale-free networks. Here, the network evolves locally, and clusters emerge naturally as a consequence of the local growth. They are depicted using a combination of forced directed and Fruchterman Reingold layout with Gephi[2]. Moreover, the figure also shows the retweet events (from others as source) for two nodes, A and B, on the bottom row. These two nodes arrive almost at the same time and establish links to two other nodes. However, node A's followees are more central, therefore, A is being exposed to more retweets. Thus, node A performs more retweets than B does. It again shows how information diffusion is affected by network structure. Overall, this figure clearly illustrates that by careful choice of parameters we can generate networks with a very different structure.

2. http://gephi.github.io/

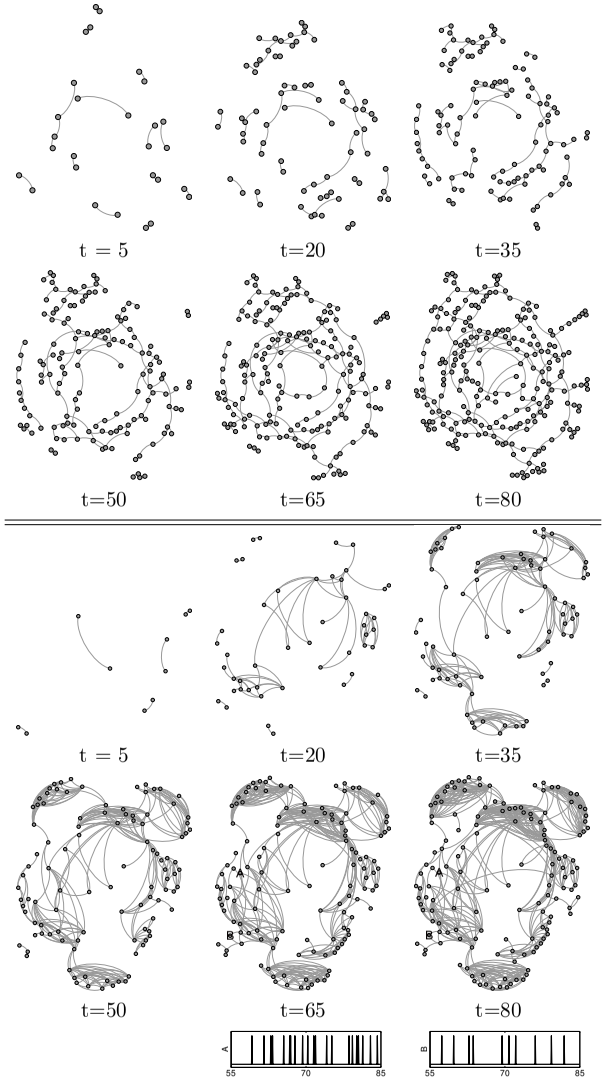


Figure 19: Evolution of two networks: one with $\beta = 0$ (1st and 2nd rows) and another one with $\beta = 0.8$ (3rd and 4th rows), and spike trains of nodes A and B (5th row).

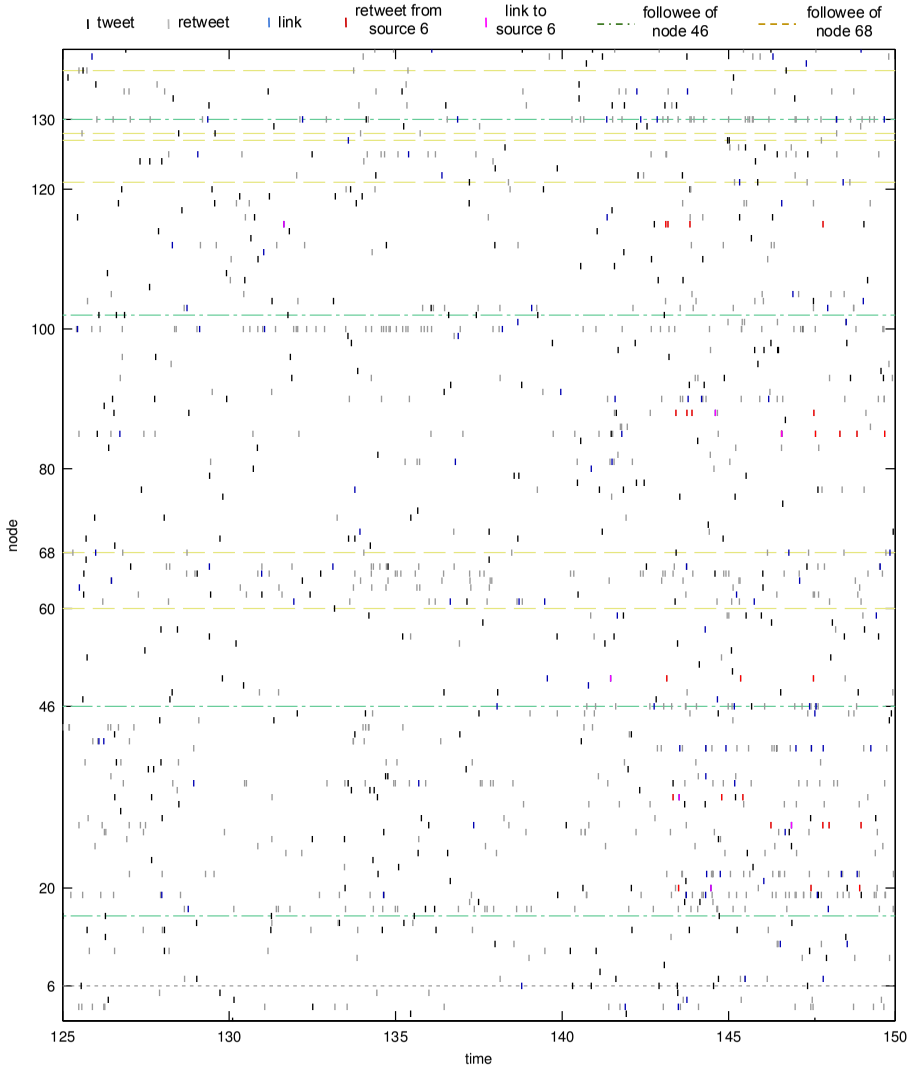


Figure 20: Coevolutionary dynamics of events for the network shown in Figure 21. Explanation is provided in section D.2.

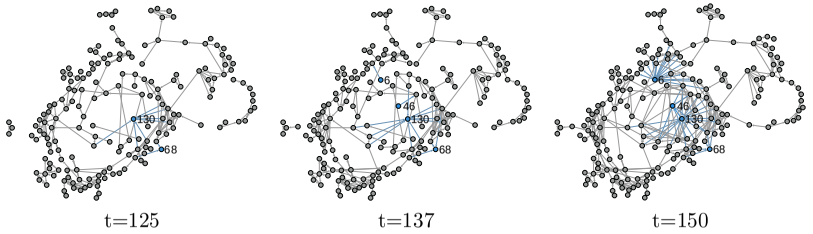


Figure 21: Network structure in which events from Figure 20 take place shown at different time points with the important nodes and links higlighted.

D.2 Cascade Visualization

Figure 20 illustrates the spike trains (tweet, retweet, and link events) for the first 140 nodes of a network simulated with a similar set of parameters as above and Figure 21 shows three snapshots of the network at different times. First, consider node 6 in the network. After she joins the network, a few nodes begin to follow him. Then, when she starts to tweet, her tweets are retweeted many times by others (red spikes) in the figure and these retweets subsequently boost the number of nodes that link to her (Magenta spikes). This clearly illustrates the scenario in which information diffusion triggers changes on the network structure. Second, consider nodes 46 and 68 and compare their associated events over time. After some time, node 46 becomes much more active than node 68. To understand why, note that soon after time 137, node 46 followed node 130, which is a very central node (*i.e.* following a lot of people), while node 68 did not. This clearly illustrates the scenario in which network evolution triggers changes on the dynamics of information diffusion.

Appendix E. Extensions

The basic model presented in Section 3 is just a show-case of the potential of point processes in modeling networks and processes over them. In this section, we extend our model in a variety of ways. More specifically, we explain how the model can be augmented to support link removal, node birth and death, and connection specific parameters. We did not perform experiments with these extensions because our real-world data set does not contain information regarding to link removal and node birth and death. Curating a comprehensive data set that can be used in modeling all these aspects of networks is left as interesting future work.

E.1 Link Deletion

We can generalize our model to support link deletion by introducing an intensity matrix $\boldsymbol{\Xi}^*(t) = (\boldsymbol{\xi}^*_{us}(t))_{u,s\in[m]}$ and model each individual intensity as a survival process. Assume $\boldsymbol{A}^+(t)$ is the previously defined counting matrix $\boldsymbol{A}(t)$, which indicates the existence of an edge at time t. Then, we introduce a new counting matrix $\boldsymbol{A}^-(t) = (A^-_{us}(t))_{u,s\in[m]}$, which

indicates the lack of an edge at time t, and we define it via its intensity function as

$$\mathbb{E}[d\boldsymbol{A}^{-}(t) \,|\, \mathcal{H}^{r}(t) \cup \mathcal{H}^{l}(t)] = \boldsymbol{\Xi}^{*}(t)\, dt, \tag{44}$$

Then, we define the intensity as

$$\boldsymbol{\xi}^{*}_{us}(t) = A^{+}_{us}(t)(\zeta_u + \nu_s \sum_{v \in \mathcal{F}u} \kappa_{\omega_3}(t) \star dA^{-}_{vs}(t)), \tag{45}$$

where the term $A^{+}_{us}(t)$ guarantees that the link has positive intensity to be removed only if it already exists, just like the term $1 - A_{us}(t)$ in Equation (21), the parameter ζ_u is the base rate of link deletion and $\nu_s \sum_{v \in \mathcal{F}u} \kappa_{\omega_3}(t) \star dA^{-}_{vs}(t)$ is the increased link deletion intensity due to increased number of followees of u who decided to unfollow s. This is an excitation term due to deleted links to source s; given s is unfollowed by some followees of u, then u may find s not a good source of information too.

Given a pair of nodes (u, s), the process starts with $A^{+}_{us}(t) = 0$. Whenever a link is created this process ends and a removal process $A^{-}_{us}(t)$ starts. Similarly, when the removal process fires, the connection is removed and a new link creation process is instantiated. These two processes interleave until the end.

E.2 Node Birth and Death

We can augment our model to consider the number of nodes $m(t)$ to change over time:

$$m(t) = m_b(t) - m_d(t) \tag{46}$$

where $m_b(t)$ and $m_d(t)$ are counting processes modeling the numbers of nodes that join and left the network till time t, respectively. The way we construct $m_b(t)$ and $m_d(t)$ guarantees that $m(t)$ is always non-negative.

The birth process, $m_b(t)$, is characterized by a conditional intensity function $\phi^{*}(t)$:

$$\mathbb{E}[dm_b(t) \,|\, \mathcal{H}^{r}(t) \cup \mathcal{H}^{l}(t)] = \phi^{*}(t)\, dt, \tag{47}$$

where

$$\phi^{*}(t) = \epsilon + \theta \sum_{u,s \in [m(t)]} \kappa_{\omega_4}(t) \star dN_{us}(t). \tag{48}$$

Here, ϵ is the constant rate of arrival and $\theta \sum_{u,s \in [m(t)]} \kappa_{\omega_4}(t) \star dN_{us}(t)$ is the increased rate of node arrival due to the increased activity of nodes. Intuitively, the higher the overall activity in the existing network, the larger the number of new users.

The construction of the death process, $m_d(t)$, is more involved. Every time a new user joins the network, we start a survival process that controls whether she leaves the network. Thus, we can stack all these survival processes in a vector, $\boldsymbol{l}(t) = (l_u(t))_{u \in [m]}$, characterized by a multidimensional conditional intensity function $\boldsymbol{\sigma}^{*}(t) = (\sigma_u(t))_{u \in [m_b(t)]}$:

$$\mathbb{E}[d\boldsymbol{l}(t) | \mathcal{H}^{r}(t) \cup \mathcal{H}^{l}(t)] = \boldsymbol{\sigma}^{*}(t)\, dt. \tag{49}$$

Intuitively, we expect the nodes with lower activity to be more likely to leave the network and thus its conditional intensity function to adopt the following form:

$$\sigma^{*}_{u}(t) = (1 - l_u(t)) \left[\sum_{j=1}^{J} \pi_j g_j(t) + \left(h(t) - \sum_{s \in [m(t)]} \kappa_{\omega_5}(t) \star dN_{us}(t) \right)_{+} \right], \tag{50}$$

where the term $(1 - l_u(t))$ ensures that a node is deleted only once, $\sum_{j=1}^{J} \pi_j g_j(t)$ is the history-independent typical rate of death, shared across nodes, which we represent by a grid of known temporal kernels, $\{g_j(t)\}$ with unknown coefficients, $\{\pi_j\}$, and the second term is capturing the effect of activity on the probability of leaving the network. More specifically, if a node is not active, we assume its intensity is upper bounded by $h(t)$ and the most active she becomes, the lower its probability of leaving the network and the larger the term $\sum_{s \in [m(t)]} \kappa_{\omega_5}(t) \star dN_{us}(t)$. The hinge function $(\cdot)_{+}$ guarantees the intensity is always positive.

Then, given the individual death processes the total death process is

$$m_d(t) = \sum_{u=1}^{m_b(t)} l_u(t), \tag{51}$$

which completes the modeling of the time-varying number of nodes.

E.3 Incorporating Features

One can simply enrich the model by taking into account the longitudinal or static information of the networked data, *e.g.*, by conditioning the intensity on additional external features, such as node attributes or edge types (Li and Zha, 2014; Tran et al., 2015). Let us assume each user u comes with a K-dimensional feature vector $\boldsymbol{x}_u$ including properties such as her age, job, location, number of followers, number of tweets, and more generally structural features (Fadaee et al., 2015). A good feature set can represent the local and global structure of the networks. For example, when the feature set includes number of followers or location information, the link creation process can be reduced to Barabasi-Albert model (Barabási and Albert, 1999), Watts-Strogatz model(Watts and Strogatz, 1998), or their many variations depending on the coefficients.

Interestingly, we can augment the information diffusion intensity as follows. We introduce a K-dimensional link intensity parameter $\boldsymbol{\eta}_u$ in which each dimension reflects the contribution of the corresponding element in the feature vector to the intensity and replace the baseline rate η_u by $\boldsymbol{\eta}_u^{\top} \boldsymbol{x}_u$. Similarly, we introduce a K-dimensional vector $\boldsymbol{\beta}_s$ where each dimension has a corresponding element in the feature vector $\boldsymbol{x}_s$ and substitute β_s by $\boldsymbol{\beta}_s \boldsymbol{x}_s$. Therefore, one can rewrite the original information diffusion intensity given by Equation (19) as:

$$\gamma^{*}_{us}(t) = \mathbb{I}[u = s]\, \boldsymbol{\eta}_u^{\top} \boldsymbol{x}_u + \mathbb{I}[u \neq s]\, \boldsymbol{\beta}_s^{\top} \boldsymbol{x}_s \sum\nolimits_{v \in \mathcal{F}_u(t)} \kappa_{\omega_1}(t) \star (A_{uv}(t)\; dN_{vs}(t))\,. \tag{52}$$

Similarly, we can parameterize the coefficients of the link creation intensity by a K-dimensional vector and write the counter-part of Equation (20) incorporating features of the node for computing the intensity:

$$\lambda^{*}_{us}(t) = (1 - A_{us}(t))(\boldsymbol{\mu}_u^{\top} \boldsymbol{x}_u + \boldsymbol{\alpha}_u^{\top} \boldsymbol{x}_u \sum_{v \in \mathcal{F}_u(t)} \kappa_{\omega_2}(t) \star dN_{vs}(t)). \tag{53}$$

Through this extension we can easily incorporate content quality of events as well; Tweets containing interesting ideas, hot news, or specific language will be retweeted more. Furthermore, these high quality tweets can increase the likelihood of new connections. All these contextual features can be easily incorporated.

Another important feature that can be used is the time. The framework can handle situations that the base rates are time dependent, *i.e.*, $\eta_u(t)$ or $\mu_u(t)$. Please refer to (Tabibian et al., 2017; Du et al., 2015) for good instances of time dependent base rates. For example, where there are seasonal increase or decrease in the follow or retweet processes, we can write the base rate as $\mu_u(t) = \sum_{k=1}^{K} r_k h_k(t)$ where $h_k(t)$'s are sinusoidal kernels with different width and r_k's are the coefficients to be learned.

Surprisingly enough, all the results for convexity for parameter learning, and efficient simulation techniques are still valid for this case too. As far as the features contribute to the intensity linearly, the log-likelihood is concave and we can simulate the model as efficiently as the original model (Li and Zha, 2014; Tran et al., 2015).

E.4 Connection Specific Parameters

Up to this point, the parameters of the link creation and removal, node birth and death and the information diffusion intensities depend on one end point of the interactions. For example β_s and η_u in the information diffusion intensity given by Equation (19) only depend on the source and the actor, respectively. However, proceeding with this example, parameters can be made connection specific, *i.e.*, Equation (19) can be restated as

$$\gamma^*_{us}(t) = \mathbb{I}[u = s]\, \eta_{us} + \mathbb{I}[u \neq s]\, \beta_{us} \sum\nolimits_{v \in \mathcal{F}_u(t)} \kappa_{\omega_1}(t) \star (A_{uv}(t)\ dN_{vs}(t))\,, \tag{54}$$

where η_{us} is the base intensity of u retweeting a tweet originated by s and β_{us} is the coefficient of excitement of u to retweet s when one of her followees retweets something from s.

Given enough computational resources and large amounts of historical data, one can take into account more complex scenarios and larger and more flexible models. For example, the middle user, say v, who is along the path of diffusion and forwards the tweet originated from s to u can also be taking into consideration, *i.e.*, defining β_{svu} as the amount of increase in intensity of user u retweeting from s when user v has just retweeted a post from s. All desirable properties of simulation algorithm and parameter estimation method still hold.

E.5 Source-oblivious Information Propagation

Our framework, Coevolve, relies on the assumption that the identity of the original source is preserved through the information propagation process. However, some platforms may only keep the last hop in the propagation. Furthermore, this assumption may be violated when users copy the content of the posts rather than reshare or retweet it. The current framework, can be easily extended to handle such cases. Basically, the source-oblivious problem can be posed as a missing (or noisy) data problem which amounts to model and infer the propagation process via incomplete observation and has been studied recently (Zipkin et al., 2016; Du et al., 2015; Stomakhin et al., 2011). Furthermore, the unobserved identity of the source can be modeled as a latent variable and inferred simultaneously with the Hawkes diffusion network (Yang and Zha, 2013).

References

Odd Aalen, Ornulf Borgan, and Hakon Gjessing. *Survival and event history analysis: a process point of view*. Springer, 2008.

Nitin Agarwal, Huan Liu, Lei Tang, and Philip S Yu. Identifying the influential bloggers in a community. In *International Conference on Web Search and Data Mining (WSDM)*, 2008.

Demetris Antoniades and Constantine Dovrolis. Co-evolutionary dynamics in social networks: A case study of twitter. *Computational Social Networks*, 2(1):14, 2015.

Lars Backstrom, Paolo Boldi, Marco Rosa, Johan Ugander, and Sebastiano Vigna. Four degrees of separation. In *Proceedings of the 4th Annual ACM Web Science Conference*, pages 33–42, 2012.

Eytan Bakshy, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. Everyone's an influencer: Quantifying influence on twitter. In *International Conference on Web Search and Data Mining (WSDM)*, 2011.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

Smriti Bhagat, Amit Goyal, and Laks VS Lakshmanan. Maximizing product adoption in social networks. In *International Conference on Web Search and Data Mining (WSDM)*, 2012.

Prasanta Bhattacharya, Tuan Q Phan, and Edoardo M Airoldi. Analyzing the co-evolution of network structure and content generation in online social networks. *ECIS 2015 Completed Research Papers*, page 18, 2015.

Charles Blundell, Jeff Beck, and Katherine A Heller. Modelling reciprocating relationships with hawkes processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, England, 2004.

Ulrik Brandes, Jürgen Lerner, and Tom AB Snijders. Networks evolving step by step: Statistical analysis of dyadic event data. In *IEEE International Conference on Advances in Social Network Analysis and Mining (ASONAM)*, 2009.

Deepayan Chakrabarti, Yiping Zhan, and Christos Faloutsos. R-mat: A recursive model for graph mining. *Computer Science Department*, page 541, 2004.

Justin Cheng, Lada Adamic, P Alex Dow, Jon Michael Kleinberg, and Jure Leskovec. Can cascades be predicted? In *International Conference on World Wide Web (WWW)*, 2014.

Daryl J Daley and David Vere-Jones. *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media, 2007.

Patrick Doreian and Frans Stokman. *Evolution of social networks*. Routledge, 2013.

Nan Du, Le Song, Ming Yuan, and Alex J Smola. Learning networks of heterogeneous influence. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

Nan Du, Le Song, Manuel Gomez-Rodriguez, and Hongyuan Zha. Scalable influence estimation in continuous-time diffusion networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

Nan Du, Mehrdad Farajtabar, Amr Ahmed, Alexander J Smola, and Le Song. Dirichlet-hawkes processes with applications to clustering continuous-time document streams. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2015.

Paul Erdos and A Rényi. On the evolution of random graphs. *Publications of tke Matkemafical Insfifufe of the Hungarian Academy of Sciences*, 5:17–61, 1960.

Saber Shokat Fadaee, Mehrdad Farajtabar, Ravi Sundaram, Javed A Aslam, and Nikos Passas. On the network you keep: analyzing persons of interest using cliqster. *Social Network Analysis and Mining*, 5(1):1–14, 2015.

Mehrdad Farajtabar, Nan Du, Manuel Gomez-Rodriguez, Isabel Valera, Hongyuan Zha, and Le Song. Shaping social activity by incentivizing users. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

Mehrdad Farajtabar, Manuel Gomez-Rodriguez, Nan Du, Mohammad Zamani, Hongyuan Zha, and Le Song. Back to the past: Source identification in diffusion networks from partially observed cascades. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015a.

Mehrdad Farajtabar, Yichen Wang, Manuel Gomez-Rodriguez, Suang Li, Hongyuan Zha, and Le Song. Coevolve: A joint point process model for information diffusion and network co-evolution. In *Advances in Neural Information Processing Systems (NIPS)*, 2015b.

Mehrdad Farajtabar, Xiaojing Ye, Sahar Harati, Le Song, and Hongyuan Zha. Multistage campaigning in social networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

Mehrdad Farajtabar, Jiachen Yang, Xiaojing Ye, Huan Xu, Rakshit Trivedi, Elias Khalil, Shuang Li, Le Song, and Hongyuan Zha. Fake news mitigation via point process based intervention. *arXiv preprint arXiv:1703.07823*, 2017.

Sharad Goel, Duncan J Watts, and Daniel G Goldstein. The structure of online diffusion networks. In *Proceedings of the 13th ACM Conference on Electronic Commerce (EC)*, 2012.

Anna Goldenberg, Alice X Zheng, Stephen E Fienberg, and Edoardo M Airoldi. A survey of statistical network models. *Foundations and Trends® in Machine Learning*, 2(2): 129–233, 2010.

Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2010.

Manuel Gomez-Rodriguez, David Balduzzi, and Bernhard Schölkopf. Uncovering the temporal dynamics of diffusion networks. In *International Conference on Machine Learning (ICML)*, 2011.

Amit Goyal, Francesco Bonchi, and Laks VS Lakshmanan. Learning influence probabilities in social networks. In *International Conference on Web Search and Data Mining (WSDM)*, 2010a.

Amit Goyal, Francesco Bonchi, Laks VS Lakshmanan, and Suresh Venkatasubramanian. Approximation analysis of influence spread in social networks. *arXiv preprint arXiv:1008.2005*, 2010b.

Mark Granovetter. The strength of weak ties. *American Journal of Sociology*, pages 1360–1380, 1973.

Thilo Gross and Bernd Blasius. Adaptive coevolutionary networks: a review. *Journal of The Royal Society Interface*, 5(20):259–271, 2008.

Thilo Gross, Carlos J Dommar DLima, and Bernd Blasius. Epidemic dynamics on an adaptive network. *Physical Review Letters*, 96(20):208701, 2006.

Asela Gunawardana, Christopher Meek, and Puyang Xu. A model for temporal dependencies in event streams. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

Fangjian Guo, Charles Blundell, Hanna Wallach, and Katherine Heller. The bayesian echo chamber: modeling social influence via linguistic accommodation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

Eric C Hall and Rebecca M Willett. Tracking dynamic point processes on networks. *IEEE Transactions on Information Theory*, 62(7):4327–4346, 2016.

Alan G Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

Petter Holme. Modern temporal network theory: A colloquium. *The European Physical Journal B*, 88(9):1–30, 2015.

Seyed Abbas Hosseini, Keivan Alizadeh, Ali Khodadadi, Ali Arabzadeh, Mehrdad Farajtabar, Hongyuan Zha, and Hamid R Rabiee. Recurrent poisson factorization for temporal recommendation. *arXiv preprint arXiv:1703.01442*, 2017.

David Hunter, Padhraic Smyth, Duy Q Vu, and Arthur U Asuncion. Dynamic egocentric models for citation networks. In *International Conference on Machine Learning (ICML)*, 2011.

David R Hunter and Kenneth Lange. A tutorial on mm algorithms. *The American Statistician*, 58(1):30–37, 2004.

Tomoharu Iwata, Amar Shah, and Zoubin Ghahramani. Discovering latent influence in online social activities via shared cascade poisson processes. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2013.

Mohammad Reza Karimi, Erfan Tavakoli, Mehrdad Farajtabar, Le Song, and Manuel Gomez-Rodriguez. Smart broadcasting: Do you want to be seen? In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2016.

David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2003.

Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is Twitter, a social network or a news media? In *International Conference on World Wide Web (WWW)*, 2010.

Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2005.

Jure Leskovec, Lars Backstrom, Ravi Kumar, and Andrew Tomkins. Microscopic evolution of social networks. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2008.

Jure Leskovec, Deepayan Chakrabarti, Jon Kleinberg, Christos Faloutsos, and Zoubin Ghahramani. Kronecker graphs: An approach to modeling networks. *Journal of Machine Learning Research*, 11(Feb):985–1042, 2010.

Liangda Li and Hongyuan Zha. Learning parametric models for social infectivity in multi-dimensional hawkes processes. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

Shuang Li, Yao Xie, Mehrdad Farajtabar, Apurv Verma, and Le Song. Detecting weak changes in dynamic events over networks. *IEEE Transactions on Signal and Information Processing over Networks*, 2017.

Wenzhao Lian, Vinayak Rao, Brian Eriksson, and Lawrence Carin. Modeling correlated arrival events with latent semi-markov processes. In *International Conference on Machine Learning (ICML)*, 2014.

Scott W Linderman and Ryan P Adams. Discovering latent network structure in point process data. In *International Conference on Machine Learning (ICML)*, 2014.

Seth A Myers and Jure Leskovec. The bursty dynamics of the twitter information network. In *International Conference on World Wide Web (WWW)*, 2014.

Yosihiko Ogata. On lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, 27(1):23–31, 1981.

Ankur P Parikh, Asela Gunawardana, and Christopher Meek. Conjoint modeling of temporal dependencies in event streams. In *UAI Bayesian Modelling Applications Workshop*. Citeseer, 2012.

Tuan Q Phan and Edoardo M Airoldi. A natural experiment of social network formation and dynamics. *Proceedings of the National Academy of Sciences*, 112(21):6595–6600, 2015.

Matthew Richardson and Pedro Domingos. Mining knowledge-sharing sites for viral marketing. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2002.

Manuel Gomez Rodriguez and Bernhard Schölkopf. Influence maximization in continuous time diffusion networks. In *International Conference on Machine Learning (ICML)*, 2012.

Manuel Gomez Rodriguez, Jure Leskovec, and Bernhard Schölkopf. Modeling information propagation with survival theory. In *International Conference on Machine Learning (ICML)*, 2013.

Daniel Mauricio Romero and Jon Kleinberg. The directed closure process in hybrid social-information networks, with an analysis of link formation on twitter. In *International Conference on Web and Social Media (ICWSM)*, 2010.

Sheldon M. Ross. *Introduction to Probability Models, Tenth Edition*. Academic Press, Inc., 2011. ISBN 978-0-12-375686-2.

Philipp Singer, Claudia Wagner, and Markus Strohmaier. Factors influencing the co-evolution of social and content networks in online social media. In *Modeling and Mining Ubiquitous Social Media*, pages 40–59. Springer, 2012.

Tom AB Snijders. Siena: Statistical modeling of longitudinal network data. In *Encyclopedia of Social Network Analysis and Mining*, pages 1718–1725. Springer, 2014.

Tom AB Snijders and SR Luchini. Statistical methods for network dynamics. In *Proceedings of the XLIII Scientific Meeting, Italian Statistical Society*. CLEUP, 2006.

Alexey Stomakhin, Martin B Short, and Andrea L Bertozzi. Reconstruction of missing data in social networks based on temporal patterns of interactions. *Inverse Problems*, 27(11): 115013, 2011.

Behzad Tabibian, Isabel Valera, Mehrdad Farajtabar, Le Song, Bernhard Schölkopf, and Manuel Gomez-Rodriguez. Distilling information reliability and source trustworthiness from digital traces. In *International Conference on World Wide Web (WWW)*, 2017.

Long Tran, Mehrdad Farajtabar, Le Song, and Hongyuan Zha. Netcodec: Community detection from individual activities. In *SIAM International Conference on Data Mining (SDM)*, 2015.

Johan Ugander, Lars Backstrom, and Jon Kleinberg. Subgraph frequencies: Mapping the empirical and extremal geography of large graph collections. In *International conference on World Wide Web (WWW)*, 2013.

Isabel Valera and Manuel Gomez-Rodriguez. Modeling adoption of competing products and conventions in social media. In *IEEE International Conference on Data Mining (ICDM)*, 2015.

Duy Q Vu, David Hunter, Padhraic Smyth, and Arthur U Asuncion. Continuous-time regression models for longitudinal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

Eric Wang, Jorge Silva, Rebecca Willett, and Lawrence Carin. Time-evolving modeling of social networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011.

Duncan J Watts and Steven H Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, June 1998.

Lilian Weng, Jacob Ratkiewicz, Nicola Perra, Bruno Gonçalves, Carlos Castillo, Francesco Bonchi, Rossano Schifanella, Filippo Menczer, and Alessandro Flammini. The role of information diffusion in the evolution of social networks. In *International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2013.

Shuai Xiao, Junchi Yan, Mehrdad Farajtabar, Le Song, Xiaoang Yang, and Hongyuan Zha. Joint modeling of event sequence and time series with attentional twin recurrent neural networks. *arXiv preprint arXiv:1703.08524*, 2017.

Hongteng Xu, Mehrdad Farajtabar, and Hongyuan Zha. Learning granger causality for hawkes processes. In *International Conference on Machine Learning (ICML)*, 2016.

Shuang-Hong Yang and Hongyuan Zha. Mixture of mutually exciting processes for viral diffusion. In *International Conference on Machine Learning (ICML)*, 2013.

Ali Zarezade, Ali Khodadadi, Mehrdad Farajtabar, Hamid R Rabiee, and Hongyuan Zha. Correlated cascades: Compete or cooperate. *arXiv preprint arXiv:1510.00936*, 2015.

Ke Zhou, Hongyuan Zha, and Le Song. Learning social infectivity in sparse low-rank networks using multi-dimensional hawkes processes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2013a.

Ke Zhou, Hongyuan Zha, and Le Song. Learning triggering kernels for multi-dimensional hawkes processes. In *International Conference on Machine Learning (ICML)*, 2013b.

Joseph R Zipkin, Frederic P Schoenberg, Kathryn Coronges, and Andrea L Bertozzi. Point-process models of social network interactions: parameter estimation and missing data recovery. *European Journal of Applied Mathematics*, 27(03):502–529, 2016.

Gerd Zschaler, Gesa A Böhme, Michael Seißinger, Cristián Huepe, and Thilo Gross. Early fragmentation in the adaptive voter model on directed networks. *Physical Review E*, 85 (4):046107, 2012.

Journal of Machine Learning Research 18 (2017) 1-60 Submitted 4/16; Revised 10/16; Published 4/17

Learning Local Dependence In Ordered Data

Guo Yu GY63@CORNELL.EDU
Department of Statistical Science
Cornell University, 1173 Comstock Hall
Ithaca, NY 14853, USA

Jacob Bien JBIEN@CORNELL.EDU
Department of Biological Statistics and Computational Biology and Department of Statistical Science
Cornell University, 1178 Comstock Hall
Ithaca, NY 14853, USA

Editor: Saharon Rosset

Abstract

In many applications, data come with a natural ordering. This ordering can often induce local dependence among nearby variables. However, in complex data, the width of this dependence may vary, making simple assumptions such as a constant neighborhood size unrealistic. We propose a framework for learning this local dependence based on estimating the inverse of the Cholesky factor of the covariance matrix. Penalized maximum likelihood estimation of this matrix yields a simple regression interpretation for local dependence in which variables are predicted by their neighbors. Our proposed method involves solving a convex, penalized Gaussian likelihood problem with a hierarchical group lasso penalty. The problem decomposes into independent subproblems which can be solved efficiently in parallel using first-order methods. Our method yields a sparse, symmetric, positive definite estimator of the precision matrix, encoding a Gaussian graphical model. We derive theoretical results not found in existing methods attaining this structure. In particular, our conditions for signed support recovery and estimation consistency rates in multiple norms are as mild as those in a regression problem. Empirical results show our method performing favorably compared to existing methods. We apply our method to genomic data to flexibly model linkage disequilibrium. Our method is also applied to improve the performance of discriminant analysis in sound recording classification.

Keywords: Local dependence, Gaussian graphical models, precision matrices, Cholesky factor, hierarchical group lasso

1. Introduction

Estimating large inverse covariance matrices is a fundamental problem in modern multivariate statistics. Consider a random vector $X = (X_1, \ldots, X_p)^T \in \mathbb{R}^p$ with mean zero and covariance matrix $E(XX^T) = \Sigma$. Unlike the covariance matrix, which captures marginal correlations among variables in X, the inverse covariance matrix $\Omega = \Sigma^{-1}$ (also known as the precision matrix) characterizes conditional correlations and, under a Gaussian model, $\Omega_{jk} = 0$ implies that X_j and X_k are conditionally independent given all other variables. When p is large, it is common to regularize the precision matrix estimator by making it sparse (see, e.g., Pourahmadi, 2013). This paper focuses on the special context in which variables have a natural ordering, such as when data are collected over time or along a genome. In such a context, it is often reasonable to assume that random variables that are far away in the ordering are less dependent than those that are close together. For example, it is known that genetic mutations that occur close together on a chromosome are more likely to be coinherited than mutations that are located far apart. We propose a method for estimating the precision matrix based on this assumption while also allowing each random variable to have its own notion of closeness.

In general settings where variables do not necessarily have a known ordering, two main types of convex methods with strong theoretical results have been developed for introducing sparsity in Ω. The first approach, known as the *graphical lasso* (Yuan and Lin, 2007; Banerjee et al., 2008; Friedman et al., 2008; Rothman et al., 2008), performs penalized maximum likelihood, solving $\min_{\Omega \succ 0, \Omega = \Omega^T} \mathcal{L}(\Omega) + \lambda P(\Omega)$, where $\mathcal{L}(\Omega) = -\log\det\Omega + n^{-1}\sum_{i=1}^n x_i^T \Omega x_i$ is, up to constants, the negative log-likelihood of a sample of n independent Gaussian random vectors and $P(\Omega)$ is the (vector) ℓ_1-norm of Ω. Zhang and Zou (2014) introduce a new convex loss function called the *D-trace loss* and propose a positive definite precision matrix estimator by minimizing an ℓ_1-penalized version of this loss. The second approach is through penalized pseudo-likelihood, the most well-known of which is called *neighborhood selection* (Meinshausen and Bühlmann, 2006). Estimators in this category are usually solved by a column-by-column approach and thus are more amenable to theoretical analysis (Yuan, 2010; Cai et al., 2011; Liu and Luo, 2012; Liu and Wang, 2012; Sun and Zhang, 2013; Khare et al., 2014). However they are not guaranteed to be positive definite and do not exploit the symmetry of Ω. Peng et al. (2009) propose a partial correlation matrix estimator that develops a symmetric version of neighborhood selection; however, positive definiteness is still not guaranteed.

In the context of variables with a natural ordering, by contrast, almost no work uses convex optimization to flexibly estimate Ω while exploiting the ordering structure. Sparsity is usually induced via the Cholesky decomposition of Σ, which leads to a natural interpretation of sparsity. Consider the Cholesky decomposition $\Sigma = QQ^T$, which implies $\Omega = L^T L$ for $L = Q^{-1}$ for lower triangular matrices Q and L with positive diagonals. The assumption that $X \sim N(0, \Sigma)$ is then equivalent to a set of linear models in terms of rows of L, i.e., $L_{11}X_1 = \varepsilon_1$ and

$$L_{rr}X_r = -\sum_{k=1}^{r-1} L_{rk}X_k + \varepsilon_r \quad r = 2, \ldots, p, \tag{1}$$

where $\varepsilon \sim N(0, I_p)$. Thus, $L_{rk} = 0$ (for $k < r$) can be interpreted as meaning that in predicting X_r from the previous random variables, one does not need to know X_k. This observation has motivated previous work, including Pourahmadi (1999); Wu and Pourahmadi (2003); Huang et al. (2006); Shojaie and Michailidis (2010); Khare et al. (2016). While these methods assume sparsity in L, they do not require local dependence because each variable is allowed to be dependent on predecessors that are distant from it (compare the upper left to the upper right panel of Figure 10).

©2017 Guo Yu and Jacob Bien.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-198.html.

The assumption of "local dependence" can be expressed as saying that each variable X_r can be best explained by exactly its K_r closest predecessors:

$$L_{rr}X_r = -\sum_{k=r-K_r}^{r-1} L_{rk}X_k + \varepsilon_r, \quad \text{for} \quad L_{rk} \neq 0, \quad r - K_r \leq k \leq r-1, \quad r = 2, \ldots, p. \quad (2)$$

Note that this does not describe all patterns of a variable depending on its nearby variables. For example, X_r can be dependent on X_{r-2} but not on X_{r-1}. In this case, the dependence is still local, but would not be captured by (2). We focus on the restricted class (2) since it greatly simplifies the interpretation of the learned dependence structure by capturing the extent of this dependence in a single number K_r, the neighborhood size.

Another desirable property of model (2) is that it admits a simple connection between the sparsity pattern of L and the sparsity pattern of the precision matrix Ω in the Gaussian graphical model. In particular, straightforward algebra shows that for $j < k$,

$$L_{kj} = \cdots = L_{pj} = 0 \implies \Omega_{jk} = 0. \quad (3)$$

Statistically, this says that if none of the variables $X_k, \ldots, X_p$ depends on X_j in the sense of (1), then X_j and X_k are conditionally independent given all other variables.

Bickel and Levina (2008) study theoretical properties in the case that all bandwidths, K_r, are equal, in which case model (2) is a K_r-ordered antedependence model (Zimmerman and Nunez-Anton, 2009). A banded estimate of L then induces a banded estimate of Ω. The *nested lasso* approach of Levina et al. (2008) provides for "adaptive banding", allowing K_r to vary with r (which corresponds to variable-order antedependence models in Zimmerman and Nunez-Anton, 2009); however, the nested lasso is non-convex, meaning that the proposed algorithm does not necessarily minimize the stated objective and theoretical properties of this estimator have not been established.

In this paper, we propose a penalized likelihood approach that provides the flexibility of the nested lasso but is formulated as a convex optimization problem, which allows us to prove strong theoretical properties and to provide an efficient, scalable algorithm for computing the estimator. The theoretical development of our method allows us to make clear comparisons with known results for the graphical lasso (Rothman et al., 2008; Ravikumar et al., 2011) in the non-ordered case. Both methods are convex penalized likelihood approaches, so this comparison highlights the similarities and differences in the ordered and non-ordered problems.

There are two key choices we make that lead to a convex formulation. First, we express the optimization problem in terms of the Cholesky factor L. The nested lasso and other methods (starting with Pourahmadi 1999) use the modified Cholesky decomposition, $\Omega = T^T D^{-1} T$, where T is a lower-triangular matrix with ones on its diagonal and D is a diagonal matrix with positive entries. While $\mathcal{L}(\Omega)$ is convex in Ω, the negative log-likelihood $\mathcal{L}(T^T D^{-1} T)$ is not jointly convex in T and D. By contrast,

$$\mathcal{L}\left(L^T L\right) = -\log\det\left(L^T L\right) + \frac{1}{n}\sum_{i=1}^{n} x_i^T L^T L x_i = -2\sum_{r=1}^{p}\log L_{rr} + \frac{1}{n}\sum_{i=1}^{n}\|Lx_i\|_2^2 \quad (4)$$

is convex in L. This parametrization is considered in Aragam and Zhou (2015), Khare et al. (2014), and Khare et al. (2016). Maximum likelihood estimation of L preserves the regression interpretation by noting that

$$\mathcal{L}\left(L^T L\right) = -2\sum_{r=1}^{p}\log L_{rr} + \frac{1}{n}\sum_{r=1}^{p}\sum_{i=1}^{n} L_{rr}^2\left(x_{ir} + \sum_{k=1}^{r-1} L_{rk}x_{ik}/L_{rr}\right)^2.$$

This connection has motivated previous work with the modified Cholesky decomposition, in which $T_{rk} = -L_{rk}/L_{rr}$ are the coefficients of a linear model in which X_r is regressed on its predecessors, and $D_{rr} = L_{rr}^{-2}$ corresponds to the error variance. The second key choice is our use of a hierarchical group lasso in place of the nested lasso's nonconvex penalty.

We introduce here some notation used throughout the paper. For two sequences of constants $a(n)$ and $b(n)$, the notation $a(n) = o\left(b(n)\right)$ means that for every $\varepsilon > 0$, there exists a constant $N > 0$ such that $|a(n)/b(n)| \leq \varepsilon$ for all $n \geq N$. And the notation $a(n) = \mathcal{O}\left(b(n)\right)$ means that there exists a constant $N > 0$ and a constant $M > 0$ such that $|a(n)/b(n)| \leq M$ for all $n \geq N$. For a sequence of random variables $A(n)$, the notation $A(n) = \mathcal{O}_P\left(b(n)\right)$ means that for every $\varepsilon > 0$, there exists a constant $M > 0$ such that $\mathrm{P}\left(|A(n)/b(n)| > M\right) \leq \varepsilon$ for all n.

For a vector $v = (v_1, \ldots, v_p) \in \mathbb{R}^p$, we define $\|v\|_1 = \sum_{j=1}^p |v_j|$, $\|v\|_2 = (\sum_{j=1}^p v_j^2)^{1/2}$ and $\|v\|_\infty = \max_j |v_j|$. For a matrix $M \in \mathbb{R}^{n\times p}$, we define the element-wise norms by two vertical bars. Specifically, $\|M\|_\infty = \max_{jk}|M_{jk}|$ and Frobenius norm $\|M\|_F = (\sum_{j,k} M_{jk}^2)^{1/2}$. For $q \geq 1$, we define the matrix-induced (operator) q-norm by three vertical bars: $|||M|||_q = \max_{\|v\|_q=1}\|Mv\|_q$. Important special cases include $|||M|||_2$, also known as the spectral norm, which is the largest singular value of M, as well as $|||M|||_1 = \max_k \sum_{j=1}^p |M_{jk}|$ and $|||M|||_\infty = \max_j \sum_{k=1}^p |M_{jk}|$. Note that $|||M|||_1 = |||M|||_\infty$ when M is symmetric.

Given a p-vector v, a $p \times p$ matrix M, and an index set T, let $v_T = (v_i)_{i\in T}$ be the $|T|$-subvector and M_T the $p \times |T|$ submatrix with columns selected from T. Given a second index set T', let $M_{TT'}$ be the $|T| \times |T'|$ submatrix with rows and columns of M indexed by T and T', respectively. Specifically, we use $L_{r\cdot}$ to denote the r-th row of L.

2. Estimator

For a given tuning parameter $\lambda \geq 0$, we define our estimator $\hat{L}$ to be a minimizer of the following penalized negative Gaussian log-likelihood

$$\hat{L} \in \underset{\substack{L:L_{rr}>0 \\ L_{rk}=0 \text{ for } r<k}}{\arg\min} \left\{-2\sum_{r=1}^{p}\log L_{rr} + \frac{1}{n}\sum_{i=1}^{n}\|Lx_i\|_2^2 + \lambda\sum_{r=2}^{p} P_r\left(L_{r\cdot}\right)\right\}. \quad (5)$$

The penalty P_r, which is applied to the r-th row, is defined by

$$P_r\left(L_{r\cdot}\right) = \sum_{\ell=1}^{r-1}\left\|W^{(\ell)} * L_{g_{r,\ell}}\right\|_2 = \sum_{\ell=1}^{r-1}\left(\sum_{m=1}^{\ell} w_{\ell m}^2 L_{rm}^2\right)^{1/2}, \quad (6)$$

where $W^{(\ell)} = (w_{\ell 1}, \ldots, w_{\ell\ell}) \in \mathbb{R}^\ell$ is a vector of weights, $*$ denotes element-wise multiplication, and $L_{g_{r,\ell}}$ denotes the vector of elements of L from the group $g_{r,\ell}$, which corresponds

to the first ℓ elements in the r-th row (for $1 \leq \ell \leq r-1$):

$$g_{r,\ell} = \left\{(r, \ell') : \ell' \leq \ell\right\}.$$

Since $g_{r,1} \subset g_{r,2} \subset \cdots \subset g_{r,r-1}$, each row r of L is penalized with a sum of $r-1$ nested, weighted ℓ_2-norm penalties. This is a hierarchical group lasso penalty (Yuan and Lin, 2007; Zhao et al., 2009; Jenatton et al., 2011; Yan and Bien, 2015) with group structure conveyed in Figure 1.

With $w_{\ell m} > 0$, this nested structure always puts more penalty on those elements that are further away from the diagonal. Since the group lasso has the effect of setting to zero a subset of groups, it is apparent that this choice of groups ensures that whenever the elements in $g_{r,\ell}$ are set to zero, elements in $g_{r,\ell'}$ are also set to zero for all $\ell' \leq \ell$. In other words, for each row of $\hat{L}$, the non-zeros are those elements within some (row-specific) distance of the diagonal. This is in contrast to the ℓ_1-penalty as used in Khare et al. (2016), which produces sparsity patterns with no particular structure (compare the top-left and top-right panels of Figure 10).

The choice of weights, $w_{\ell m}$, affects both the empirical and theoretical performance of the estimator. We focus primarily on a quadratically decaying set of weights,

$$w_{\ell m} = \frac{1}{(\ell - m + 1)^2}, \tag{7}$$

but also consider the unweighted case (in which $w_{\ell m} = 1$). The decay counteracts the fact that the elements of L appear in differing numbers of groups (for example L_{r1} appears in $r-1$ groups whereas $L_{r,r-1}$ appears in just one group). In a related problem, Bien et al. (2016) choose weights that decay more slowly with $\ell - m$ than (7). Our choice makes the enforcement of hierarchy weaker so that our penalty behaves more closely to the lasso penalty (Tibshirani, 1996). The choice of weight sequence in (7) is more amenable to theoretical analysis; however, in practice the unweighted case is more efficiently implemented and works well empirically.

Problem (5) is convex in L. While $-\log\det(\cdot)$ is strictly convex, $-\sum_r \log(L_{rr})$ is not strictly convex in L. Thus, the arg min in (5) may not be unique. In Section 4, we provide sufficient conditions to ensure uniqueness with high probability.

In Appendix A, we show that (5) decouples into p independent subproblems, each of which estimates one row of L. More specifically, let $\mathbf{X} \in \mathbb{R}^{n \times p}$ be a sample matrix with independent rows $x_i \sim N(0, \Sigma)$, $\hat{L}_{11} = n^{1/2}(\mathbf{X}_1^T \mathbf{X}_1)^{-1/2}$ and for $r = 2, \ldots, p$,

$$\hat{L}_{r,1:r} = \underset{\beta \in \mathbb{R}^r; \beta_r > 0}{\arg\min} \left\{ -2\log\beta_r + \frac{1}{n}\left\|\mathbf{X}_{1:r}\beta\right\|_2^2 + \lambda \sum_{\ell=1}^{r-1} \left(\sum_{m=1}^{\ell} w_{\ell m}^2 \beta_m^2 \right)^{1/2} \right\}. \tag{8}$$

This observation means that the computation can be easily parallelized, which potentially can achieve a linear speed up with the number of CPU cores. Theoretically, to analyze the properties of $\hat{L}$ it is easier to start by studying an estimator of each row, i.e., a solution to (8). We will see in Section 4 that problem (8) has connections to a penalized regression problem, meaning that both the assumptions and results we can derive are better than if we were working with a penalty based on Ω.

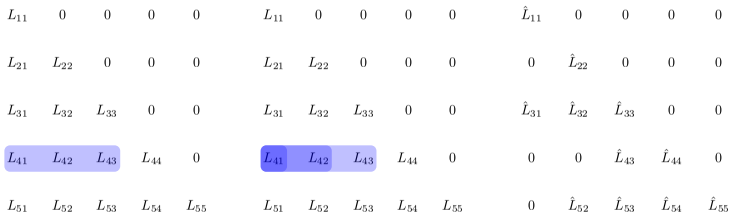


Figure 1: There are $\binom{p}{2}$ groups used in the penalty, with each row r having $r-1$ nested groups $g_{r,1} \subset g_{r,2} \subset \cdots \subset g_{r,r-1}$. Left: the group $g_{4,3}$. Middle: the nested group structure $g_{4,1} \subset g_{4,2} \subset g_{4,3}$. Right: A possible sparsity pattern in $\hat{L}$, where elements in $g_{2,1}, g_{4,2}$ (and thus $g_{4,1}$) and $g_{5,1}$ are set to zero.

In light of the regression interpretation of (1), $\hat{L}$ provides an interpretable notion of local dependence; however, we can of course also use our estimate of L to estimate Ω: $\hat{\Omega} = \hat{L}^T\hat{L}$. By construction, this estimator is both symmetric and positive definite. Unlike a lasso penalty, which would induce unstructured sparsity in the estimate of L and thus would not be guaranteed to produce a sparse estimate of Ω, the adaptively banded structure in our estimator of L can yield a generally banded $\hat{\Omega}$ with sparsity pattern determined by (3) (See the top-left and bottom-left panels in Figure 10 for an example).

3. Computation

As observed above, we can compute $\hat{L}$ by solving (in parallel across r) problem (8). Consider an alternating direction method of multipliers (ADMM) approach that solves the equivalent problem

$$\min_{\beta, \gamma \in \mathbb{R}^r; \beta_r > 0} \left\{ -2\log\beta_r + \frac{1}{n}\left\|\mathbf{X}_{1:r}\beta\right\|_2^2 + \lambda \sum_{\ell=1}^{r-1} \left(\sum_{m=1}^{\ell} w_{\ell m}^2 \gamma_m^2 \right)^{1/2} \quad \text{s.t.} \quad \beta = \gamma \right\}.$$

Algorithm 1 presents the ADMM algorithm, which repeatedly minimizes this problem's augmented Lagrangian over β, then over γ, and then updates the dual variable $u \in \mathbb{R}^r$. The main computational effort in the algorithm is in solving (9) and (10). Note that (9) has a smooth objective function. Straightforward calculus gives the closed-form solution (see Appendix B for detailed derivation),

$$\begin{aligned}
\beta_r^{(t+1)} &= \frac{-B - \sqrt{B^2 - 8A}}{2A} > 0 \\
\beta_{-r}^{(t+1)} &= -\left(2S_{-r,-r}^{(r)} + \rho I\right)^{-1} \left(2S_{-r,r}^{(r)}\beta_r^{(t+1)} + u_{-r}^{(t)} - \rho\gamma_{-r}^{(t)}\right),
\end{aligned}$$

Algorithm 1 ADMM algorithm to solve (8)

Require: $\beta^{(0)}$, $\gamma^{(0)}$, $u^{(0)}$, $\rho > 0$, $t = 1$.

repeat

$$\beta^{(t)} \leftarrow \underset{\beta \in \mathbb{R}^r : \beta_r > 0}{\arg\min} \left\{ -2 \log \beta_r + \frac{1}{n} \left\| \mathbf{X}_{1:r} \beta \right\|_2^2 + \left(\beta - \gamma^{(t-1)} \right)^T u^{(t-1)} + \frac{\rho}{2} \left\| \beta - \gamma^{(t-1)} \right\|_2^2 \right\} \tag{9}$$

$$\gamma^{(t)} \leftarrow \underset{\gamma \in \mathbb{R}^r}{\arg\min} \left\{ \frac{\rho}{2} \left\| \gamma - \beta^{(t)} - \rho^{-1} u^{(t-1)} \right\|_2^2 + \lambda \sum_{\ell=1}^{r-1} \left(\sum_{m=1}^{\ell} w_{\ell m}^2 \gamma_m^2 \right)^{1/2} \right\} \tag{10}$$

$u^{(t)} \leftarrow u^{(t-1)} + \rho \left(\beta^{(t)} - \gamma^{(t)} \right)$

$t \leftarrow t + 1$

until convergence

return $\gamma^{(t)}$

where

$$\begin{aligned} S^{(r)} &= \frac{1}{n} \mathbf{X}_{1:r}^T \mathbf{X}_{1:r} \\ A &= 4 S_{r,-r}^{(r)} \left(2 S_{-r,-r}^{(r)} + \rho I \right)^{-1} S_{-r,r}^{(r)} - 2 S_{r,r}^{(r)} - \rho < 0 \\ B &= 2 S_{r,-r}^{(r)} \left(2 S_{-r,-r}^{(r)} + \rho I \right)^{-1} \left(u_{-r}^{(t)} - \rho \gamma_{-r}^{(t)} \right) - u_r^{(t)} + \rho \gamma_r^{(t)}. \end{aligned}$$

The closed-form update above involves matrix inversion. With $\rho > 0$, the matrix $2S_{-r,-r}^{(r)} + \rho I$ is invertible even when $r > n$. Since determining a good choice for the ADMM parameter ρ is in general difficult, we adapt the dynamic ρ updating scheme described in Section 3.4.1 of Boyd et al. (2011).

Solving (10) requires evaluating the proximal operator of the hierarchical group lasso with general weights. We adopt the strategy developed in Bien et al. (2016) (based on a result of Jenatton et al. 2011), which solves the dual problem of (10) by performing Newton's method on at most $r - 1$ univariate functions. The detailed implementation is given in Algorithm 3 in Appendix C. Each application of Newton's method corresponds to performing an elliptical projection, which is a step of blockwise coordinate ascent on the dual of (10) (see Appendix D for details). Finally we observe in Algorithm 2 that for the unweighted case ($w_{\ell m} = 1$), solving (10) is remarkably efficient.

The `R` package `varband` provides `C++` implementations of Algorithms 1 and 2.

4. Statistical Properties

In this section we study the statistical properties of our estimator. In what follows, we consider a lower triangular matrix L having row-specific bandwidths, K_r. The first $J_r = r - 1 - K_r$ elements of row r are zero, and the band of non-zero off-diagonals (of size K_r) is denoted $\mathcal{I}_r = \{J_r + 1, \ldots, r-1\}$. We also denote $\mathcal{I}_r^c = \{1, 2, \ldots, r\} \setminus \mathcal{I}_r$. See Figure 2 for a graphical example of $K_5, J_5, \mathcal{I}_4$, and $\mathcal{I}_4^c$.

Algorithm 2 Algorithm for solving (10) for unweighted estimator

Require: $\beta^{(t)}, u^{(t-1)} \in \mathbb{R}^r$, $\lambda, \rho > 0$.

Initialize $\gamma^{(t)} = \beta^{(t)} + u^{(t-1)}/\rho$ and $\tau = \lambda/\rho$

for $\ell = 1, \ldots, r-1$ **do**

$$\left(\gamma^{(t)} \right)_{1:\ell} \leftarrow \left(1 - \frac{\tau}{\left\| \left(\gamma^{(t)} \right)_{1:\ell} \right\|_2} \right)_+ \left(\gamma^{(t)} \right)_{1:\ell}$$

return $\gamma^{(t)}$.

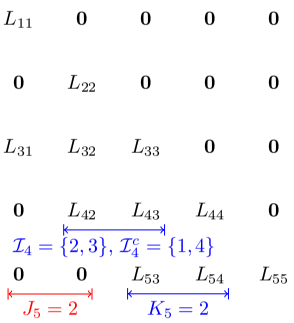


Figure 2: Schematic showing $J_r, K_r, \mathcal{I}_r$, and $\mathcal{I}_r^c$.

Our theoretical analysis is built on the following assumptions:

A1 *Gaussian assumption*: The sample matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ has n independent rows with each row x_i drawn from $N(\mathbf{0}, \Sigma)$.

A2 *Sparsity assumption*: The true Cholesky factor $L \in \mathbb{R}^{p \times p}$ is the lower triangular matrix with positive diagonal elements such that the precision matrix $\Omega = \Sigma^{-1} = L^T L$. The matrix L has row-specific bandwidths K_r such that $L_{rj} = 0$ for $0 < j < r - K_r$.

A3 *Irrepresentable condition*: There exists some $\alpha \in (0, 1]$ such that

$$\max_{2 \le r \le p} \max_{\ell \in \mathcal{I}_r^c} \left\| \Sigma_{\ell \mathcal{I}_r} \left(\Sigma_{\mathcal{I}_r \mathcal{I}_r} \right)^{-1} \right\|_1 \le \frac{6}{\pi^2} (1 - \alpha)$$

A4 *Bounded singular values*: There exists a constant κ such that

$$0 < \kappa^{-1} \le \sigma_{\min}(L) \le \sigma_{\max}(L) \le \kappa$$

When $\max_r K_r < n$, the Gaussianity assumption **A1** implies that $\mathbf{X}_{\mathcal{I}_r}$ has full column rank for all r with probability one. Our analysis applies to the general high-dimensional scaling scheme where $K_r = K_r(n)$ and $p = p(n)$ can grow with n.

For $r = 2, \ldots, p$ and $\ell \in \mathcal{I}_r^c = \{1, \ldots, J_r, r\}$, let

$$\theta_r^{(\ell)} := \mathrm{Var}\left(X_\ell | X_{\mathcal{I}_r}\right) \quad \text{and} \quad \theta_r := \max_{\ell \in \mathcal{I}_r^c} \theta_r^{(\ell)}.$$

By Assumption **A1**, $\theta_r^{(\ell)} = \Sigma_{\ell\ell} - \Sigma_{\ell\mathcal{I}_r} \left(\Sigma_{\mathcal{I}_r\mathcal{I}_r}\right)^{-1} \Sigma_{\mathcal{I}_r\ell}$ represents the noise variance when regressing X_ℓ on $X_{\mathcal{I}_r}$, i.e., for $\ell = 1, \ldots, J_r, r$,

$$X_\ell = \Sigma_{\ell\mathcal{I}_r} (\Sigma_{\mathcal{I}_r\mathcal{I}_r})^{-1} X_{\mathcal{I}_r}^T + E_\ell \qquad \text{with} \qquad E_\ell \sim N\left(0, \theta_r^{(\ell)}\right). \tag{11}$$

In words, $\theta_r^{(\ell)}$ measures the degree to which X_ℓ cannot be explained by the variables in the support and θ_r is the maximum such value over all ℓ outside of the support $\mathcal{I}_r$ in the r-th row. Intuitively, the difficulty of the estimation problem increases with θ_r. Note that for $r = 1, \ldots, p$, (1) implies $\theta_r^{(r)} = 1/L_{rr}^2$.

Assumption **A3** (along with the $\beta_{\min}$ condition) is essentially a necessary and sufficient condition for support recovery of lasso-type methods (see, e.g., Zhao and Yu, 2006; Meinshausen and Bühlmann, 2006; Wainwright, 2009; Van de Geer and Bühlmann, 2009; Ravikumar et al., 2011). The constant $\alpha \in (0, 1]$ is usually referred to as the irrepresentable (incoherence) constant (Wainwright, 2009). Intuitively, the irrepresentable condition requires low correlations between signal and noise predictors, and thus a value of α that is close to 1 implies that recovering the support is easier to achieve. The constant $6\pi^{-2}$ is determined by the choice of weight (7) and can be eliminated by absorbing its reciprocal into the definition of the weights $w_{\ell m}$. Doing so, one finds that our irrepresentable condition is essentially the same as the one found in the regression setting (Wainwright, 2009) despite the fact that our goal is estimating a precision matrix.

Assumption **A4** is a bounded singular value condition. Recalling that $\Omega = L^T L$,

$$0 < \kappa^{-2} \leq \sigma_{\min}(\Sigma) \leq \sigma_{\max}(\Sigma) \leq \kappa^2, \tag{12}$$

which is equivalent to the commonly used bounded eigenvalue condition in other literatures.

4.1 Row-Specific Results

We start by analyzing support recovery properties of our estimator for each row, i.e., the solution to the subproblem (8). For $r > n$, the Hessian of the negative log-likelihood is not positive definite, meaning that the objective function may not be strictly convex in β and the solution not necessarily unique. Intuitively, if the tuning parameter λ is large, the resulting row estimate $\hat{L}_{r\cdot}$ is sparse and thus includes most variation in a small subset of the r variables. More specifically, for large λ, $\hat{\mathcal{I}}_r \subseteq \mathcal{I}_r$ and thus by Assumption **A1**, $\mathbf{X}_{\hat{\mathcal{I}}_r}$ has full rank, which implies that $\hat{L}_{r\cdot}$ is unique. The series of technical lemmas in Appendix E precisely characterizes the solution.

The first part of the theorem below shows that with an appropriately chosen tuning parameter λ the solution to (8) is sparse enough to be unique and that we will not overestimate the true bandwidth. Knowing that the support of the unique row estimator $\hat{L}_{r\cdot}$ is

contained in the true support reduces the dimension of the parameter space, and thus leads to a reasonable error bound. Of course, if our goal were simply to establish the uniqueness of $\hat{L}_{r\cdot}$ and that $\hat{K}_r \leq K_r$, we could trivially take $\lambda = \infty$ (resulting in $\hat{K}_r = 0$). The latter part of the theorem thus goes on to provide a choice of λ that is sufficiently small to guarantee that $\hat{K}_r = K_r$ (and, furthermore, that the signs of all non-zeros are correctly recovered).

Theorem 1 *Consider the family of tuning parameters*

$$\lambda = \frac{8}{\alpha} \sqrt{\frac{\theta_r \log r}{n}} \tag{13}$$

and weights given by (7)*. Under Assumptions **A1**–**A4**, if the tuple (n, J_r, K_r) satisfies*

$$n > \alpha^{-2} \left(3\pi^2 K_r + 8\right) \theta_r \kappa^2 \log J_r, \tag{14}$$

then with probability greater than $1 - c_1 \exp\{-c_2 \min(K_r, \log J_r)\} - 7\exp(-c_3 n)$ for some constants c_1, c_2, c_3 independent of n and J_r, the following properties hold:

1. *The row problem* (8) *has a unique solution $\hat{L}_{r\cdot}$ and $\hat{K}_r \leq K_r$.*
2. *The estimate $\hat{L}_{r\cdot}$ satisfies the element-wise ℓ_∞ bound,*

$$\left\| \hat{L}_{r\cdot} - L_{r\cdot} \right\|_\infty \leq \lambda \left(4 \left\| \left(\Sigma_{\mathcal{I}_r\mathcal{I}_r}\right)^{-1} \right\|_\infty + 5\kappa^2 \right). \tag{15}$$

3. *If in addition,*

$$\min_{j \geq J_r + 1} |L_{rj}| > \lambda \left(4 \left\| \left(\Sigma_{\mathcal{I}_r\mathcal{I}_r}\right)^{-1} \right\|_\infty + 5\kappa^2 \right), \tag{16}$$

then exact signed support recovery holds: For all $j \leq r$, $\mathrm{sign}(\hat{L}_{rj}) = \mathrm{sign}(L_{rj})$.

Proof See Appendix F. ■

In the classical setting where the ambient dimension r is fixed and the sample size n is allowed to go to infinity, $\lambda \to 0$ and the above scaling requirement is satisfied. By (15) the row estimator $\hat{L}_{r\cdot}$ is consistent as is the classical maximum likelihood estimator. Moreover, it recovers the true support since (16) holds automatically. In high-dimensional scaling, however, both n and r are allowed to change, and we are interested in the case where r can grow much faster than n. Theorem 1 shows that, if $\left\| \left(\Sigma_{\mathcal{I}_r\mathcal{I}_r}\right)^{-1} \right\|_\infty = \mathcal{O}(1)$ and if n can grow as fast as $K_r \log J_r$, then the row estimator $\hat{L}_{r\cdot}$ still recovers the exact support of $L_{r\cdot}$ when the signal is at least $\mathcal{O}(\sqrt{\frac{\log r}{n}})$ in size, and the estimation error $\max_j |\hat{L}_{rj} - L_{rj}|$ is $\mathcal{O}(\sqrt{\frac{\log r}{n}})$. Intuitively, for the row estimator to detect the true support, we require that the true signal be sufficiently large. The condition (16) imposes limitations on how fast the signal is allowed to decay, which is the analogue to the commonly known "$\beta_{\min}$ condition" that is assumed for establishing support recovery of the lasso.

Remark 2 *Both the choice of tuning parameter* (13) *and the error bound* (15) *depend on the true covariance matrix via* θ_r*. This quantity can be bounded by* κ^2 *as in* (12) *using the fact that* $(\Sigma_{\mathcal{I}_r\mathcal{I}_r})^{-1}$ *is positive definite:*

$$\theta_r = \max_{\ell \in \mathcal{I}_r^c} \theta_r^{(\ell)} = \max_{\ell \in \mathcal{I}_r^c} \left\{ \Sigma_{\ell\ell} - \Sigma_{\ell\mathcal{I}_r} \left(\Sigma_{\mathcal{I}_r\mathcal{I}_r}\right)^{-1} \Sigma_{\mathcal{I}_r\ell} \right\} \leq \max_{\ell \in \mathcal{I}_r^c} \Sigma_{\ell\ell} \leq \kappa^2.$$

The proof of Theorem 1 shows that the results in this theorem still hold true if we replace θ_r *by* κ^2*. This observation leads to the fact that we can select a tuning parameter having the properties of the theorem that does not depend on the unknown sparsity level* K_r*. Therefore, our estimator is adaptive to the underlying unknown bandwidths.*

4.1.1 Connections to the regression setting

In (1) we showed that estimation of the r-th row of L can be interpreted as a regression of X_r on its predecessors. It is thus very interesting to compare Theorem 1 to the standard high-dimensional regression results. Consider the following linear model of a vector $\mathbf{y} \in \mathbb{R}^n$ of the form

$$\mathbf{y} = \mathbf{Z}\eta + \omega \qquad \omega \sim N(\mathbf{0}, \sigma^2 I_n) \tag{17}$$

where $\eta \in \mathbb{R}^p$ is the unknown but fixed parameter to estimate, $\mathbf{Z} \in \mathbb{R}^{n\times p}$ is the design matrix with each row an observation of p predictors, σ^2 is the variance of the zero-mean additive noise ω. A standard approach in the high-dimensional setting where $p \gg n$ is the lasso (Tibshirani, 1996), which solves the convex optimization problem,

$$\min_{\eta\in\mathbb{R}^p} \frac{1}{2n} \|\mathbf{y} - \mathbf{Z}\eta\|_2^2 + \lambda \|\eta\|_1 , \tag{18}$$

where $\lambda > 0$ is a regularization parameter. In the setting where η is assumed to be sparse, the lasso solution is known to be able to successfully recover the signed support of the true η with high probability when λ is of the scale $\sigma\sqrt{\frac{\log p}{n}}$ and certain technical conditions are satisfied (Wainwright, 2009).

Despite the added complications of working with the log term in the objective of (8), Theorem 1 gives a clear indication that, in terms of difficulty of support recovery, the row estimate problem (8) is essentially the same as a lasso problem with random design, i.e., with each row $z_i \sim N(\mathbf{0}, \Sigma)$ (Theorem 3, Wainwright, 2009). Indeed, a comparison shows that the two irrepresentable conditions are equivalent. Moreover, θ_r plays the same role as Wainwright (2009)'s $\max_i \left(\Sigma_{S^cS^c} - \Sigma_{S^cS}\left(\Sigma_{SS}\right)^{-1}\Sigma_{SS^c}\right)_{ii}$, a threshold constant of the conditional covariance, where S is the support of the true η.

Städler et al. (2010) introduce an alternative approach to the lasso, in the context of penalized mixture regression models, that solves the optimization problem,

$$(\hat{\phi}, \hat{\rho}) = \arg\min_{\phi,\rho} \left\{ -2\log\rho + \frac{1}{n}\|\rho\mathbf{y} + \mathbf{Z}\phi\|_2^2 + \lambda\|\phi\|_1 \right\}, \tag{19}$$

where $\hat{\sigma} = \hat{\rho}^{-1}$ and $\hat{\eta} = -\hat{\phi}/\hat{\rho}$. Note that (19) basically coincides with (8) except for the penalty.

In Städler et al. (2010), the authors study the asymptotic and non-asymptotic properties of the ℓ_1-penalized estimator for the general mixture regression models where the loss functions are non-convex. The theoretical properties of (19) are studied in Sun and Zhang (2010), which partly motivates the scaled lasso (Sun and Zhang, 2012).

The theoretical work of Sun and Zhang (2010) differs from ours both in that they study the ℓ_1 penalty (instead of the hierarchical group lasso) and in their assumptions. The nature of our problem requires the sample matrix to be random (as in **A1**), while Sun and Zhang (2010) considers the fixed design setting, which does not apply in our context. Moreover, they provide prediction consistency and a deviation bound of the regression parameters estimation in ℓ_1 norm. We give exact signed support recovery results for the regression parameters as well as estimation deviation bounds in various norm criteria. Also, they take an asymptotic point of view while we give finite sample results.

4.2 Matrix Bandwidth Recovery Result

With the properties of the row estimators in place, we are ready to state results about estimation of the matrix L. The following theorem gives an analogue to Theorem 1 in the matrix setting. Under similar conditions, with one particular choice of tuning parameter, the estimator recovers the true bandwidth for all rows adaptively with high probability.

Theorem 3 *Let* $\theta = \max_r \theta_r$ *and* $K = \max_r K_r$*, and take*

$$\lambda = \frac{8}{\alpha}\sqrt{\frac{2\theta\log p}{n}} \tag{20}$$

and weights given by (7)*. Under Assumptions* ***A1****–****A4****, if* (n, p, K) *satisfies*

$$n > \alpha^{-2}\theta\kappa^2\left(12\pi^2 K + 32\right)\log p, \tag{21}$$

then with probability greater than $1 - cp^{-1}$ *for some constant* c *independent of* n *and* p*, the following properties hold:*

1. *The estimator* $\hat{L}$ *is unique, and it is at least as sparse as* L*, i.e.,* $\hat{K}_r \leq K_r$ *for all* r*.*
2. *The estimator* $\hat{L}$ *satisfies the element-wise* ℓ_∞ *bound,*

$$\left\|\hat{L} - L\right\|_\infty \leq \lambda\left(4\max_r \left\|\left(\Sigma_{\mathcal{I}_r\mathcal{I}_r}\right)^{-1}\right\|_\infty + 5\kappa^2\right). \tag{22}$$

3. *If in addition,*

$$\min_r \min_{j\geq J_r+1} |L_{rj}| > \lambda\left(4\max_r \left\|\left(\Sigma_{\mathcal{I}_r\mathcal{I}_r}\right)^{-1}\right\|_\infty + 5\kappa^2\right), \tag{23}$$

then exact signed support recovery holds: $\operatorname{sign}(\hat{L}_{rj}) = \operatorname{sign}(L_{rj})$ *for all* r *and* j*.*

Proof See Appendix G. ∎

As discussed in Remark 2, we can replace θ with its upper bound κ^2, and the results remain true. This theorem shows that one can properly estimate the sparsity pattern across all rows

exactly using only one tuning parameter chosen without any prior knowledge of the true bandwidths. In Section 4.1.1, we noted that the conditions required for support recovery and the element-wise ℓ_∞ error bound for estimating a row of L is similar to those of the lasso in the regression setting. A union bound argument allows us to translate this into exact bandwidth recovery in the matrix setting and to derive a reasonable convergence rate under conditions as mild as that of a lasso problem with random design. This technique is similar in spirit to neighborhood selection (Meinshausen and Bühlmann, 2006), though our approach is likelihood-based.

Comparing (21) to (14), we see that the sample size requirement for recovering L is determined by the least sparse row. While intuitively one would expect the matrix problem to be harder than any single row problem, we see that in fact the two problems are basically of the same difficulty (up to a multiplicative constant).

In the setting where variables exhibit a natural ordering, Shojaie and Michailidis (2010) proposed a penalized likelihood framework like ours to estimate the structure of directed acyclic graphs (DAGs). Their method focuses on variables which are standardized to have unit variance. In this special case, penalized likelihood does not involve the log-determinant term and under similar assumptions to ours, they proved support recovery consistency. However, they use lasso and adaptive lasso (Zou, 2006) penalties, which do not have the built-in notion of local dependence. Since these ℓ_1-type penalties do not induce structured sparsity in the Cholesky factor, the resulting precision matrix estimate is not necessarily sparse. By contrast, our method does not assume unit variances and learns an adaptively banded structure for $\hat{L}$ that leads to a sparse $\hat{\Omega}$ (thereby encoding conditional dependencies).

To study the difference between the ordered and non-ordered problems, we compare our method with Ravikumar et al. (2011), who studied the graphical lasso estimator in a general setting where variables are not necessarily ordered. Let $\mathcal{S}$ index the edges of the graph specified by the sparsity pattern of $\Omega = \Sigma^{-1}$. The sparsity recovery result and convergence rate are established under an irrepresentable condition imposed on $\Gamma = \Sigma \otimes \Sigma \in \mathbb{R}^{p^2 \times p^2}$:

$$\max_{e \in \mathcal{S}^c} \left\| \Gamma_{e\mathcal{S}} \left(\Gamma_{\mathcal{S}\mathcal{S}} \right)^{-1} \right\|_1 \leq (1 - \alpha) \tag{24}$$

for some $\alpha \in (0, 1]$. Our Assumption **A3** is on each variable through the entries of the true covariance Σ while (24) imposes such a condition on the edge variables $Y_{(j,k)} = X_j X_k - \mathrm{E}(X_j X_k)$, resulting in a vector ℓ_1-norm restriction on a much larger matrix Γ, which can be more restrictive for large p. More specifically, condition (24) arises in Ravikumar et al. (2011) to tackle the analysis of the $\log\det\Omega$ term in the graphical lasso problem. By contrast, in our setting the parameterization in terms of L means that the $\log\det$ term is simply a sum of log terms on diagonal elements and is thus easier to deal with, leading to the milder irrepresentable assumption. Another difference is that they require the sample size $n > c\kappa_\Gamma^2 d^2 \log p$ for some constant c. The quantity d measures the maximum number of non-zero elements in each row of the true Σ, which in our case is $2K+1$, and $\kappa_\Gamma = \left\| \left(\Gamma_{\mathcal{S}\mathcal{S}} \right)^{-1} \right\|_\infty$ can be much larger than κ^2. Thus, comparing to (21), one finds that their sample size requirement is much more restrictive. A similar comparison could also be made with the lasso penalized D-trace estimator (Zhang and Zou, 2014), whose irrepresentable condition involves $\Gamma = (\Sigma \otimes I + I \otimes \Sigma)/2 \in \mathbb{R}^{p^2 \times p^2}$. Of course, the results in both Ravikumar et al. (2011) and Zhang and Zou (2014) apply to estimators invariant to permutation of variables; additionally, the random vector only needs to satisfy an exponential-type tail condition.

4.3 Precision Matrix Estimation Consistency

Although our primary target of interest is L, the parameterization $\Omega = L^T L$ makes it natural for us to try to connect our results of estimating L with the vast literature in directly estimating Ω, which is the standard estimation target when the known ordering is not available. In this section, we consider the estimation consistency of Ω using the results we obtained for L. The following theorem gives results of how well $\hat{\Omega} = \hat{L}^T \hat{L}$ performs in estimating the true precision matrix $\Omega = L^T L$ in terms of various matrix norm criteria.

Theorem 4 *Let $\theta = \max_r \theta_r$, $K = \max_r K_r$ and $s = \sum_r K_r$ denote the total number of non-zero off-diagonal elements in L. Define $\zeta_\Sigma = \frac{8\sqrt{2\theta}}{\alpha}\left(4 \max_r \left\| \left(\Sigma_{\mathcal{I}_r \mathcal{I}_r} \right)^{-1} \right\|_\infty + 5\kappa^2\right)$. Under the assumptions in Theorem 3, the following deviation bounds hold with probability greater than $1 - cp^{-1}$ for some constant c independent of n and p:*

$$\begin{aligned}
\left\| \hat{\Omega} - \Omega \right\|_\infty &\leq 2\zeta_\Sigma |||L|||_\infty \sqrt{\frac{\log p}{n}} + \zeta_\Sigma^2 (K+1) \frac{\log p}{n}, \\
\left\| \left\| \hat{\Omega} - \Omega \right\| \right\|_\infty &\leq 2\zeta_\Sigma |||L|||_\infty (K+1) \sqrt{\frac{\log p}{n}} + \zeta_\Sigma^2 (K+1)^2 \frac{\log p}{n}, \\
\left\| \left\| \hat{\Omega} - \Omega \right\| \right\|_2 &\leq 2\zeta_\Sigma |||L|||_\infty (K+1) \sqrt{\frac{\log p}{n}} + \zeta_\Sigma^2 (K+1)^2 \frac{\log p}{n}, \\
\left\| \hat{\Omega} - \Omega \right\|_F &\leq 2\kappa\zeta_\Sigma \sqrt{\frac{(s+p)\log p}{n}} + \zeta_\Sigma^2 (K+1) \sqrt{s+p} \frac{\log p}{n}.
\end{aligned}$$

When the quantities ζ_Γ, $|||L|||_\infty$, and κ are treated as constants, these bounds can be summarized more succinctly as follows:

Proof See Appendix H. ■

Corollary 5 *Using the notation and conditions in Theorem 4, if ζ_Γ, $|||L|||_\infty$, and κ remain constant, then the scaling $(K+1)^2 \log p = o(n)$ is sufficient to guarantee the following estimation error bounds:*

$$\begin{aligned}
\left\| \hat{\Omega} - \Omega \right\|_\infty &= \mathcal{O}_P\left(\sqrt{\frac{\log p}{n}} \right), \\
\left\| \left\| \hat{\Omega} - \Omega \right\| \right\|_\infty &= \mathcal{O}_P\left((K+1) \sqrt{\frac{\log p}{n}} \right), \\
\left\| \left\| \hat{\Omega} - \Omega \right\| \right\|_2 &= \mathcal{O}_P\left((K+1) \sqrt{\frac{\log p}{n}} \right), \\
\left\| \hat{\Omega} - \Omega \right\|_F &= \mathcal{O}_P\left(\sqrt{\frac{(s+p)\log p}{n}} \right).
\end{aligned}$$

The conditions for these deviation bounds to hold are those required for support recovery as in Theorem 3. In many cases where estimation consistency is more of interest than support recovery, we can still deliver the desired error rate in Frobenius norm, matching the rate derived in Rothman et al. (2008). In particular, we can drop the strong irrepresentable assumption (**A3**) and weaken the Gaussian assumption (**A1**) to the following marginal sub-Gaussian assumption:

A5 *Marginal sub-Gaussian assumption*: The sample matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$ has n independent rows with each row drawn from the distribution of a zero-mean random vector $X = (X_1, \cdots, X_p)^T$ with covariance Σ and sub-Gaussian marginals, i.e.,

$$\mathrm{E}\exp\left(tX_j/\sqrt{\Sigma_{jj}}\right) \leq \exp\left(Ct^2\right)$$

for all $j = 1, \ldots, p$, $t \geq 0$ and for some constant $C > 0$ that does not depend on j.

Theorem 6 *Under Assumption **A2**, **A4** and **A5**, with tuning parameter λ of scale $\sqrt{\frac{\log p}{n}}$ and weights as in (7), the scaling $(s+p)\log p = o(n)$ is sufficient for the following estimation error bounds in Frobenius norm to hold:*

$$\left\|\hat{L} - L\right\|_F = \mathcal{O}_P\left(\sqrt{\frac{(s+p)\log p}{n}}\right),$$
$$\left\|\hat{\Omega} - \Omega\right\|_F = \mathcal{O}_P\left(\sqrt{\frac{(s+p)\log p}{n}}\right).$$

Proof See Appendix I. ■

The rates in Corollary 5 (and Theorem 6) essentially match the rates obtained in methods that directly estimate Ω (e.g., the graphical lasso estimator, studied in Rothman et al. 2008, Ravikumar et al. 2011, and the column-by-column methods as in Cai et al. 2011, Liu and Wang 2012, and Sun and Zhang 2013). However, the exact comparison in rates with these methods is not straightforward. First, the targets of interest are different. In the setting where the variables have a known ordering, we are more interested in the structural information among variables that is expressed in L, and thus accurate estimation of L is more important. When such ordering is not available as considered in Rothman et al. (2008); Cai et al. (2011); Liu and Wang (2012) and so on, however, the conditional dependence structure encoded by the sparsity pattern in Ω is more of interest, and the accuracy of directly estimating Ω is the focus. Moreover, deviation bounds of different methods are built upon assumptions that treat different quantities as constants. Quantities that are assumed to remain constant in the analysis of one method might actually be allowed to scale with ambient dimension in a nontrivial manner in another method, which makes direct rate comparison among different methods complicated and less illuminating.

Our analysis can be extended to the unweighted version of our estimator, i.e., with weight $w_{\ell m} = 1$, but under more restrictive conditions and with slower rates of convergence. Specifically, Assumption **A3** becomes $\max_{\ell\in\mathcal{I}_r^c}\left\|\Sigma_{\ell\mathcal{I}_r}\left(\Sigma_{\mathcal{I}_r\mathcal{I}_r}\right)^{-1}\right\|_1 \leq (1-\alpha)/K_r$ for each

$r = 2, \ldots, p$. With the same tuning parameter choice (13) and (20), the terms of K_r and K in sample size requirements (14) and (21) are replaced with K_r^2 and K^2, respectively. The estimation error bounds in all norms are multiplied by an extra factor of K. All of the above indicates that in highly sparse situations (in which K is very small), the unweighted estimator has very similar theoretical performance to the weighted estimator.

5. Simulation Study

In this section we study the empirical performance of our estimators (both with weights as in (7) and with no weights, i.e., $w_{\ell m} = 1$) on simulated data. For comparison, we include two other sparse precision matrix estimators designed for the ordered-variable case:

- **Non-Adaptive Banding** (Bickel and Levina, 2008): This method estimates L as a lower-triangular matrix with a fixed bandwidth K applying across all rows. The regularization parameter used in this method is the fixed bandwidth K.
- **Nested Lasso** (Levina et al., 2008): This method yields an adaptive banded structure by solving a set of penalized least-squares problems (both the loss function and the nested-lasso penalty are non-convex). The regularization parameter controls the amount of penalty and thus the sparsity level of the resulting estimate.

All simulations are run at a sample size of $n = 100$, where each sample is drawn independently from the p-dimensional normal distribution $N(\mathbf{0}, (L^TL)^{-1})$. We compare the performance of our estimators with the methods above both in terms of support recovery (in Section 5.1) and in terms of how well $\hat{L}$ estimates L (in Section 5.2). For support recovery, we consider $p = 200$ and for estimation accuracy, we consider $p = 50, 100, 200$, which corresponds to settings where $p < n$, $p = n$, and $p > n$, respectively.

We simulate under the following models for L. We adapt the parameterization $L = D^{-1}T$ as in Khare et al. (2016), where D is a diagonal matrix with diagonal elements drawn randomly from a uniform distribution on the interval $[2, 5]$, and T is a lower-triangular matrix with ones on its diagonal and off-diagonal elements defined as follows:

- **Model 1**: Model 1 is at one extreme of bandedness of the Cholesky factor L, in which we take the lower triangular matrix $L \in \mathbb{R}^{p\times p}$ to have a strictly banded structure, with each row having the same bandwidth $K_r = K = 1$ for all r. Specifically, we take $T_{r,r} = 1$, $T_{r,r-1} = 0.8$ and $T_{r,j} = 0$ for $j < r - 1$.
- **Model 2**: Model 2 is at the other extreme, in which we allow K_r to vary with r. We take T to be a block diagonal matrix with 5 blocks, each of size $p/5$. Within each block, with probability 0.5 each row r is assigned with a non-zero bandwidth that is randomly drawn from a uniform distribution on $\{1, \ldots, r-1\}$ (for $r > 1$). Each non-zero element in T is then drawn independently from a uniform distribution on the interval $[0.1, 0.4]$, and is assigned with a positive/negative sign with probability 0.5.
- **Model 3**: Model 3 is a denser and thus more challenging version of Model 2, with T a block diagonal matrix with only 2 blocks. Each of the blocks is of size $p/2$ but is otherwise generated as in Model 2.

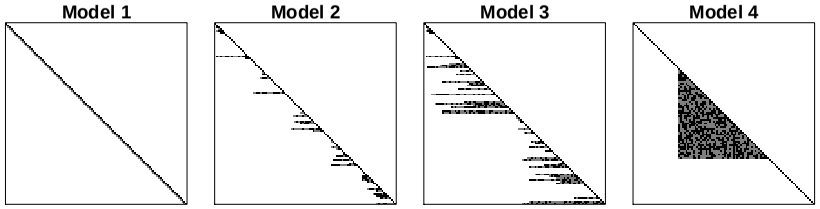


Figure 3: Schematic of four simulation scenarios with $p = 100$: (from left to right) Model 1 is strictly banded, Model 2 has small variable bandwidth, Model 3 has large variable bandwidth, and Model 4 is block-diagonal. Black, gray, and white stand for positive, negative, and zero entries, respectively. The proportion of elements that are non-zero is 4%, 6%, 15%, and 26%, respectively.

- **Model 4**: Model 4 is a dense block diagonal model. The matrix T has a completely dense lower-triangular block from the $p/4$-th row to the $3p/4$-th row and is zero everywhere else. Within this block, all off-diagonal elements are drawn uniformly from $[0.1, 0.2]$, and positive/negative signs are then assigned with probability 0.5.

Model 1 is a stationary autoregressive model of order 1. By the regression interpretation (1), for each r, it can be verified that the autoregressive polynomial of the r-th row of Models 2, 3, and 4 has all roots outside the unit circle, which characterizes stationary autoregressive models of orders equal to the corresponding row-wise bandwidths. See Figure 3 for examples of the four sparsity patterns for $p = 100$. The non-adaptive banding method should benefit from Model 1 while the nested lasso and our estimators are expected to perform better in the other three models where each row has its own bandwidth.

For all four models and every value of p considered, we verified that Assumptions **A3** and **A4** hold and then simulated $n = 100$ observations according to each of the four models based on Assumption **A1**.

5.1 Support Recovery

We first study how well the different estimators identify zeros in the four models above. We generate $n = 100$ random samples from each model with $p = 200$. The tuning parameter $\lambda \geq 0$ in (5) measures the amount of regularization and determines the sparsity level of the estimator. We use 100 tuning parameter values for each estimator and repeat the simulation 10 times.

Figure 4 shows the sensitivity (fraction of true non-zeros that are correctly recovered) and specificity (fraction of true zeros that are correctly set to zero) of each method parameterized by its tuning parameter (in the case of non-adaptive banding, the parameter is the bandwidth itself, ranging from 0 to $p - 1$). Each set of 10 curves of the same color corresponds to the results of one estimator, and each curve within the set corresponds to

the result of one draw from 10 simulations. Curves closer to the upper-right corner indicate better classification performance (the $x + y = 1$ line corresponds to random guessing).

The sparsity level of the non-adaptive banding estimator depends only on the pre-specified bandwidth (which is the method's tuning parameter) and not on the data itself. Consequently, the sensitivity-specificity curves for the non-adaptive banding do not vary across replications when simulating from a particular underlying model. The sparsity levels of the nested lasso and our methods, by contrast, hinge on the data, thus giving a different curve for each replication.

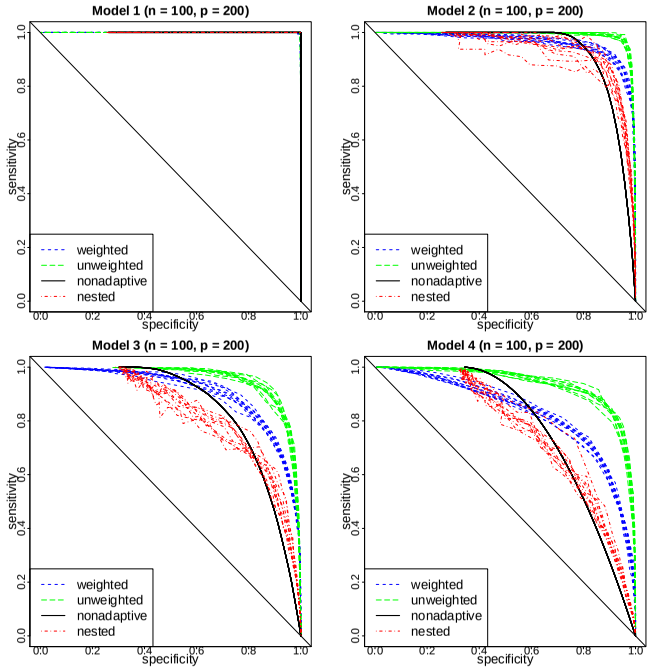


Figure 4: ROC curves showing support recovery when the true L (top-left) is strictly banded, (top-right) has small variable bandwidth, (bottom-left) has large variable bandwidth, and (bottom-right) is block-diagonal, over 10 replications.

In practice, we find that our methods and the nested lasso sometimes produce entries with very small, but non-zero, absolute values. To study support recovery, we set all estimates whose absolute values are below 10^{-10} to zero, both in our estimators and the nested lasso.

In Model 1, we observe that all methods considered attain perfect classification accuracy for some value of their tuning parameter. While the non-adaptive approach is guaranteed to do so in this scenario, it is reassuring to see that the more flexible methods can still perfectly recover this sparsity pattern.

In Model 2, we observe that our two methods outperform the nested lasso, which itself, as expected, outperforms the non-adaptive banding method. As the model becomes more challenging (from Model 2 to Model 4), the performances of all four methods start deteriorating. Interestingly, the nested lasso no longer retains its advantage over non-adaptive banding in Models 3 and 4, while the performance advantage of our methods become even more substantial.

The fact that the unweighted version of our method outperforms the weighted version stems from the fact that all models are comparatively sparse for $p = 200$, and so the heavier penalty on each row delivered by the unweighted approach recovers the support more easily than the weighted version.

5.2 Estimation Accuracy

We proceed by comparing the estimators in terms of how far $\hat{L}$ is from L. To this end, we generate $n = 100$ random samples from the four models with $p = 50$, $p = 100$, and $p = 200$. Each method is computed with its tuning parameter selected to maximize the Gaussian likelihood on the validation data in a 5-fold cross-validation. For comparison, we report the estimation accuracy of each estimate in terms of the scaled Frobenius norm $\frac{1}{p}\left\|\hat{L} - L\right\|_F^2$, the matrix infinity norm $\left\|\left\|\hat{L} - L\right\|\right\|_\infty$, the spectral norm $\left\|\left\|\hat{L} - L\right\|\right\|_2$, and the (scaled) Kullback-Leibler loss $\frac{1}{p}\left[\mathrm{tr}(\Omega^{-1}\hat{\Omega}) - \log\det(\Omega^{-1}\hat{\Omega}) - p\right]$ (Levina et al., 2008).

The simulation is repeated 50 times, and the results are summarized in Figure 5 through Figure 8. Each figure corresponds to a model, and consists of a 4-by-3 panel layout. Each row corresponds to an error measure, and each column corresponds to a value of p.

As expected, the non-adaptive banding estimator does better than the other estimators in Model 1. In Models 2, 3, and 4, where bandwidths vary with row, our estimators and the nested lasso outperform non-adaptive banding.

A similar pattern is observed as in support recovery. As the model becomes more complex and p gets larger, the performance of the nested lasso degrades and gradually becomes worse than non-adaptive banding. By contrast, as the estimation problem becomes more difficult, the advantage in performance of our methods becomes more obvious.

We again observe that the unweighted estimator performs better than the weighted one. As shown in Section 4, the overall performance of our method hinges on the underlying model complexity (measured in terms of $\max_r K_r$) as well as the relative size of n and p. When n is relatively small, usually a more constrained method (like the unweighted estimator) is preferred over a more flexible method (like the weighted estimator). So in our simulation setting, it is reasonable to observe that the unweighted method works better. Note that as the underlying L becomes denser (from Model 1 to Model 4), the performance difference between the weighted and the unweighted estimator diminishes. This corroborates our discussion in the end of Section 4 that the performance of the unweighted estimator becomes worse when the underlying model is dense.

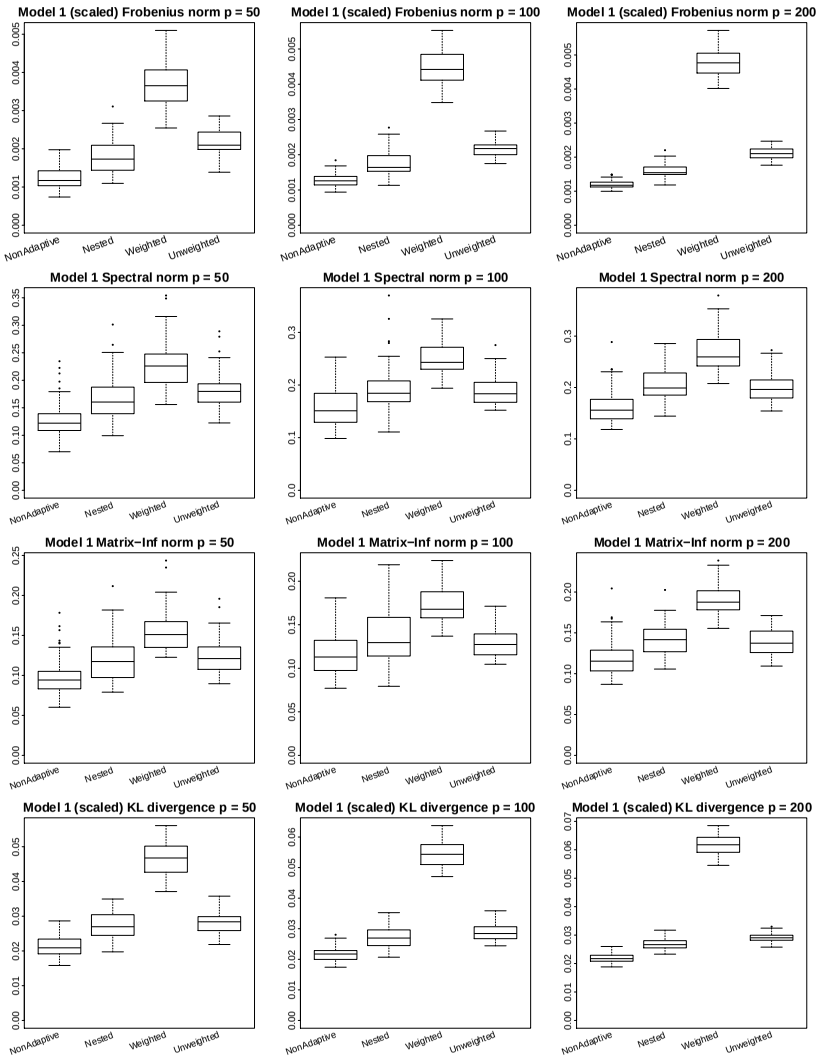


Figure 5: Estimation accuracy when data are generated from Model 1, which is strictly banded.

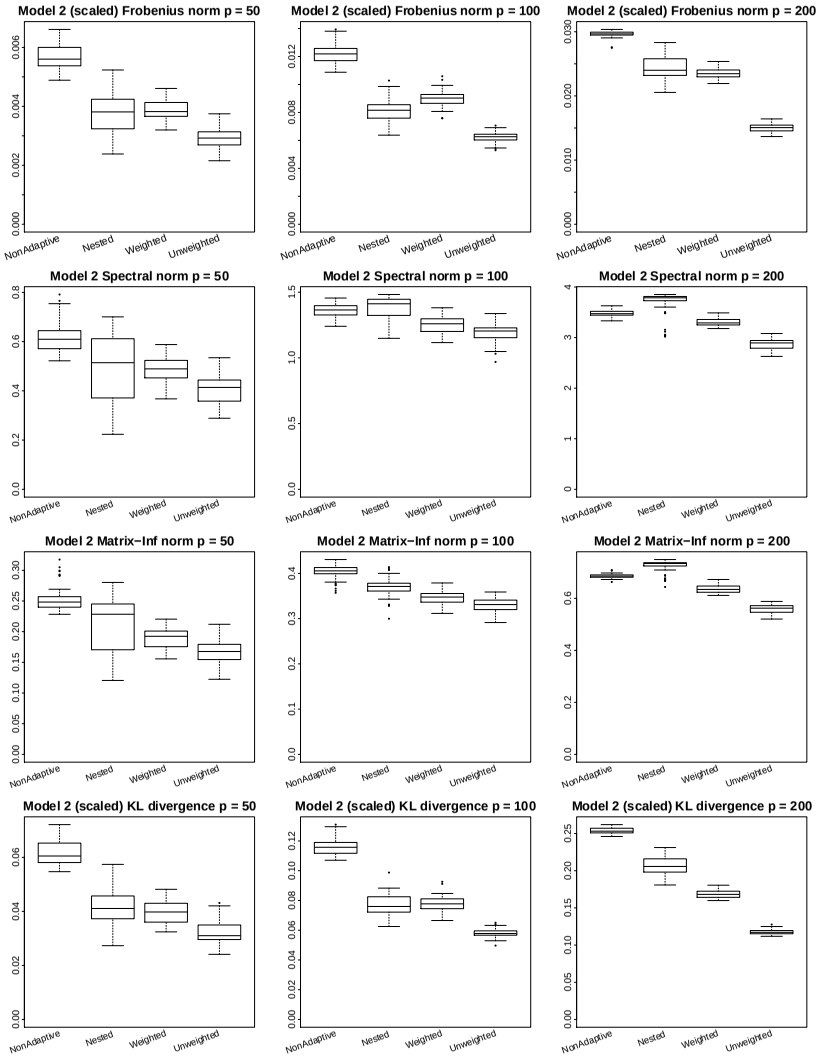


Figure 6: Estimation accuracy when data are generated from Model 2, which has small variable bandwidth.

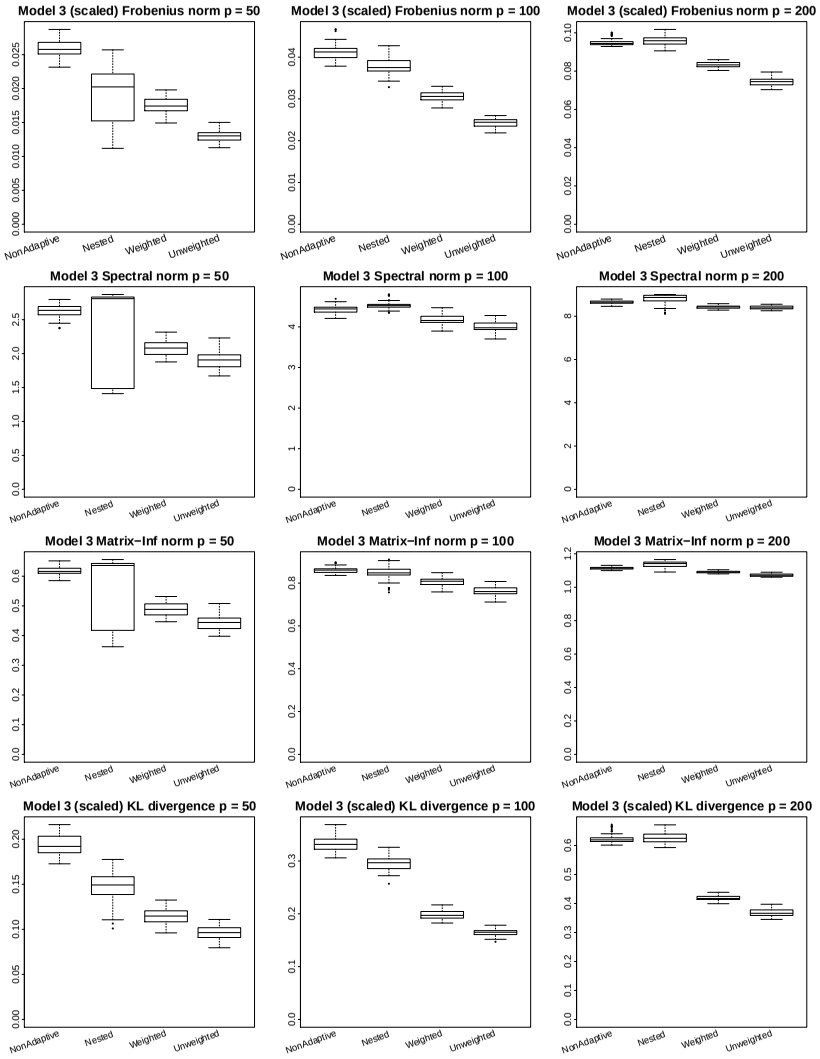


Figure 7: Estimation accuracy when data are generated from Model 3, which has large variable bandwidth.

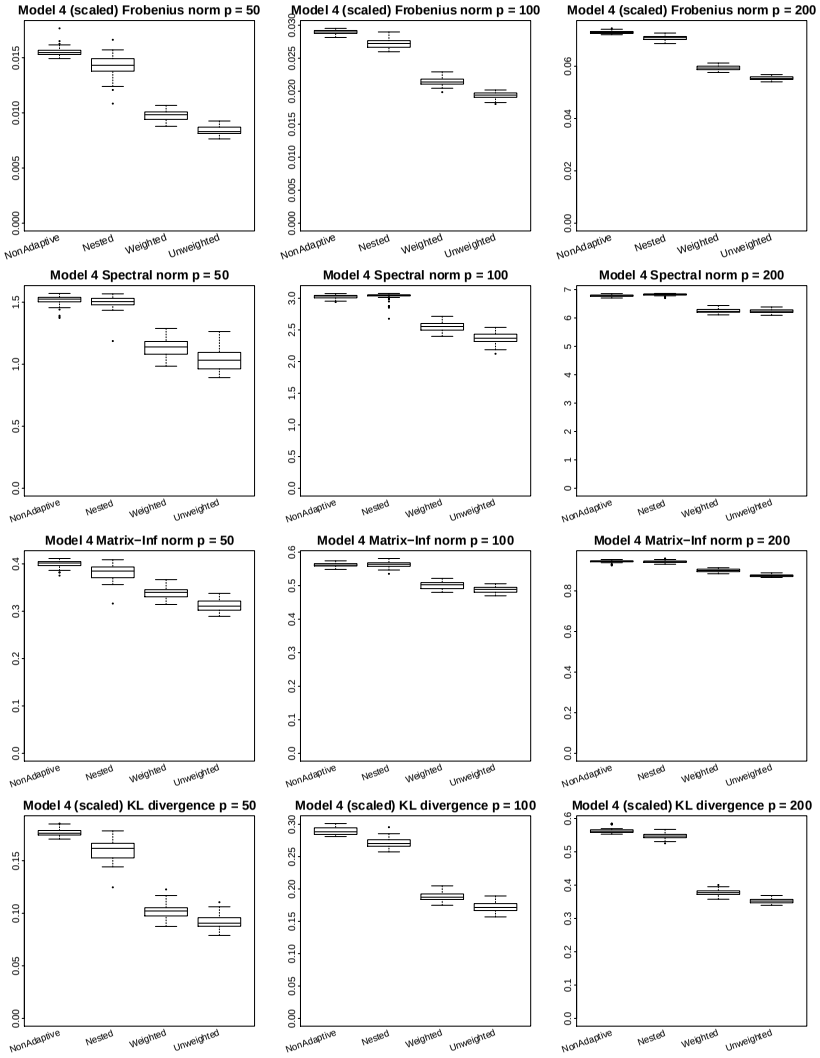


Figure 8: Estimation accuracy when data are generated from Model 4, which is block-diagonal.

6. Applications to Data Examples

In this section, we illustrate the practical merits of our proposed method by applying it to two data examples. We start with an application to genomic data where our method can help model the local correlations along the genome. In Section 6.2 we compare our method with other estimators within the context of a sound recording classification problem.

6.1 An Application to Genomic Data

We consider an application of our estimator to modeling correlation along the genome. Genetic mutations that occur close together on a chromosome are more likely to be co-inherited than mutations that are located far apart (or on separate chromosomes). This leads to local correlations between genetic variants in a population. Biologists refer to this local dependence as *linkage disequilibrium* (LD). The width of this dependence is known to vary along the genome due to the variable locations of recombination hotspots, which suggests that adaptively banded estimators may be quite suitable in these contexts.

We study HapMap phase 3 data from the International HapMap project (Consortium et al., 2010). The data consist of $n = 167$ humans from the YRI (Yoruba in Ibadan, Nigeria) population, and we focus on $p = 201$ consecutive tag SNPs on chromosome 22 (after filtering out infrequent sites with minor allele frequency $\leq 10\%$).

While tag SNP data, which take discrete values $\{0, 1, 2\}$, are non-Gaussian, we argue that our estimator is still sensible to use in this case. First, the parameterization $\Omega = L^T L$ does not depend on the Gaussian assumption. Moreover the estimator corresponds to minimizing a penalized Bregman divergence of the log-determinant function (Ravikumar et al., 2011). Furthermore, the least-squares term in (5) can be interpreted as minimizing the prediction error in the linear models (1) while the log terms act as log-barrier functions to impose positive diagonal entries (which ensures that the resulting $\hat{L}$ is a valid Cholesky factor).

To gauge the performance of our estimator on modeling LD, we randomly split the 167 samples into training and testing sets of sizes 84 and 83, respectively. Along a path of tuning parameters with decreasing values, estimators $\hat{L}$ are computed on the training data. To evaluate $\hat{L}$ on a vector $\tilde{x}$ from the test data set, we can compute the error in predicting $\hat{L}_{rr}\tilde{x}_r$ using $-\sum_{k=1}^{r-1}\hat{L}_{r,k}\tilde{x}_k$ via (1) for each r, giving the error

$$\mathrm{err}(\tilde{x}) = \frac{1}{p-1}\sum_{r=2}^{p}\left(\hat{L}_{rr}\tilde{x}_r + \sum_{k=1}^{r-1}\hat{L}_{rj}\tilde{x}_k\right)^2. \tag{25}$$

This quantity (with mean and the standard deviation over test samples) is reported in Figure 9 for our estimator under the two weighting schemes. Recall that the quadratically decaying weights (7) act essentially like the ℓ_1 penalty. For numerical comparison, we also include the result of the estimator with ℓ_1 penalty, which is the *CSCS (Convex Sparse Cholesky Selection)* method proposed in Khare et al. (2016). For both the non-adaptive banding and the nested lasso methods, we found that their implementations fail to work due to the collinearity of the columns of $\mathbf{X}$.

Figure 9 shows that our estimators are effective in improving modeling performance over a diagonal estimator (attained when λ is sufficiently large) and strongly outperform

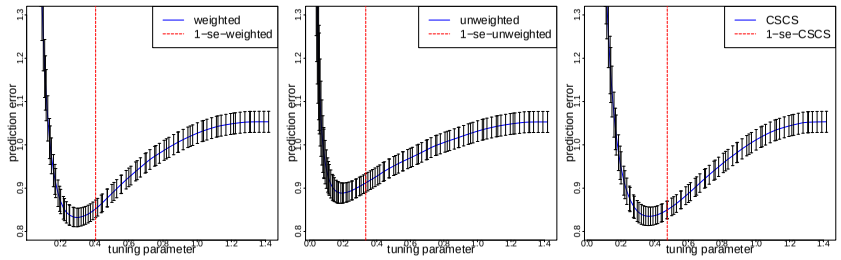


Figure 9: Prediction error (computed on an independent test set) of the weighted (left), unweighted (middle), and CSCS (right) estimators.

the plain MLE (as evidenced by the sharp increase in prediction error as $\lambda \to 0$). As expected, the weighted estimator performs very similarly to the CSCS estimator, which uses the ℓ_1 penalty. Both of these perform better than the unweighted one. However, the sparsity pattern obtained by the two penalties are different (as shown in Figure 10).

In Figure 10 we show the recovered signed support of the weighted, unweighted, and CSCS estimators and their corresponding precision matrices. Black, gray, and white stand for positive, negative, and zero entries, respectively. Tuning parameters are chosen using the one-standard-error rule (see, e.g., Hastie et al., 2009). The r-th row of the estimated matrix $\hat{L}$ reveals the number of neighboring SNPs necessary for reliably predicting the state of the r-th SNP. Interestingly, we see some evidence of small block-like structures in $\hat{L}$, consistent with the hotspot model of recombination as previously described. This regression-based perspective to modeling LD may be a useful complement to the more standard approach, which focuses on raw marginal correlations. Finally, the sparsity recovered by the CSCS estimator, which uses the ℓ_1 penalty, is less easily interpretable, since some entries far from the diagonal are non-zero, losing the notion of 'local'.

6.2 An Application to Phoneme Classification

In this section, we develop an application of our method to a classification problem described in Hastie et al. (2009). The data contain $n = 1717$ continuous speech recordings, which are categorized into two vowel sounds: 'aa' ($n_1 = 695$) and 'ao' ($n_2 = 1022$). Each observation (x_i, y_i) has a predictor $x_i \in \mathbb{R}^p$ representing the (log) intensity of the sound across $p = 256$ frequencies and a class label $y_i \in \{1, -1\}$. It may be reasonable to apply our method in this problem since the features are frequencies, which come with a natural ordering

In linear discriminant analysis (LDA), one models the features as multivariate Gaussian conditional on the class: $x_i|y_i = k \sim N_p(\mu^{(k)}, \Sigma)$ for $k \in \{1, -1\}$; in quadratic discriminant analysis (QDA), one allows each class to have its own covariance matrix: $x_i|y_i = k \sim N_p(\mu^{(k)}, \Sigma^{(k)})$. The LDA/QDA classification rules assign an observation $x \in \mathbb{R}^p$ to class k that maximizes $\hat{P}(y = k|x) \propto \hat{P}(x|y = k)\hat{P}(y = k)$, where the estimated probability

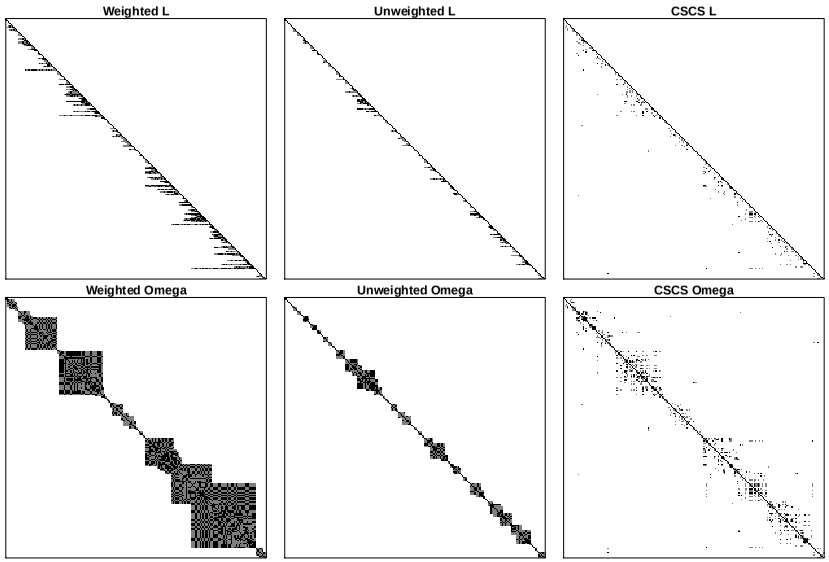


Figure 10: Estimates of linkage disequilibrium with tuning parameters selected by the one-standard-error rule and their corresponding precision matrix estimates.

	Unweighted	Weighted	Nested Lasso	Non-adaptive	CSCS
LDA	0.271	0.246	0.250	0.268	0.245
QDA	0.232	0.256	0.221	0.246	0.267

Table 1: Average test data classification error rate of discriminant analysis of phoneme data

$\hat{P}(x|y = k)$ is calculated using maximum likelihood estimates $\hat{\mu}^{(k)}$, $\hat{\Sigma}$, and $\hat{\Sigma}^{(k)}$. More precisely, in the ordered case, the resulting class k maximizes the LDA/QDA scores:

$$\begin{aligned}\delta^{(k)}_{\text{LDA}}(x) &= x^T\hat{\Omega}\hat{\mu}^{(k)} - \frac{1}{2}(\hat{\mu}^{(k)})^T\hat{\Omega}\hat{\mu}^{(k)} + \log\hat{\pi}^{(k)} \\ &= (\hat{L}x)^T\hat{L}\hat{\mu}^{(k)} - \frac{1}{2}\left\|\hat{L}\hat{\mu}^{(k)}\right\|_2^2 + \log\hat{\pi}^{(k)} \qquad (26)\\ \delta^{(k)}_{\text{QDA}}(x) &= x^T\hat{\Omega}^{(k)}\hat{\mu}^{(k)} - \frac{1}{2}(\hat{\mu}^{(k)})^T\hat{\Omega}^{(k)}\hat{\mu}^{(k)} + \log\hat{\pi}^{(k)} \\ &= (\hat{L}^{(k)}x)^T\hat{L}^{(k)}\hat{\mu}^{(k)} - \frac{1}{2}\left\|\hat{L}^{(k)}\hat{\mu}^{(k)}\right\|_2^2 + \log\hat{\pi}^{(k)}. \qquad (27)\end{aligned}$$

Note that it is the precision matrix, not the covariance matrix, that is used in the above scores. In the setting where $p > n$, the MLE of Ω or $\Omega^{(k)}$ does not exist. A regularized estimate of precision matrix that exploits the natural ordering information can be helpful in this setting.

To demonstrate the use of our estimator in the high-dimensional setting, we randomly split the data into two parts, with 10% of the data assigned to the training set and the remaining 90% of the data assigned to the test set. On the training set, we use 5-fold cross-validation to select the tuning parameter minimizing misclassification error on the validation data. The estimates $\hat{L}$ and $\hat{L}^{(k)}$ are then plugged into (26) and (27) along with $\hat{\mu}^{(k)} = \sum_{i\in\text{class }k} x_i/n^{(k)}$ and $\hat{\pi}^{(k)} = n^{(k)}/n_{\text{train}}$ to calculate the misclassification error in the test set. For comparison, we also include non-adaptive banding, the nested lasso, and CSCS. We compute the classification error (summarized in Table 1), averaged over 10 random train-test splits.

We first observe that, in general, the adaptive methods perform better than the non-adaptive one (which assumes a fixed bandwidth). It is again found that the performance of the weighted estimator is very similar to the one using ℓ_1 penalty (i.e., the CSCS method). And our results are comparable to the nested lasso both in LDA and QDA. Interestingly, we find that the weighted estimator does better in LDA while the unweighted estimator performs better in QDA. The reason, we suspect, is that QDA requires the estimation of more parameters than LDA and therefore favors more constrained methods like the unweighted estimator, which more strongly discourages non-zeros from being far from the diagonal than the weighted one.

An `R` (R Core Team, 2016) package, named `varband`, is available on `CRAN`, implementing our estimator. The estimation is very fast with core functions coded in `C++`, allowing us to solve large-scale problems in substantially less time than is possible with the `R`-based implementation of the nested lasso.

7. Conclusion

We have presented a new flexible method for learning local dependence in the setting where the elements of a random vector have a known ordering. The model amounts to sparse estimation of the inverse of the Cholesky factor of the covariance matrix with variable bandwidth. Our method is based on a convex formulation that allows it to simultaneously yield a flexible adaptively-banded sparsity pattern, enjoy efficient computational algorithms, and be studied theoretically. To our knowledge, no previous method has all these properties. We show how the matrix estimation problem can be decomposed into independent row estimation problems, each of which can be solved via an ADMM algorithm having efficient updates. We prove that our method recovers the signed support of the true Cholesky factor and attains estimation consistency rates in several matrix norms under assumptions as mild as those in linear regression problems. Simulation studies show that our method compares favorably to two pre-existing estimators in the ordered setting, both in terms of support recovery and in terms of estimation accuracy. Through a genetic data example, we illustrate how our method may be applied to model the local dependence of genetic variations in genes along a chromosome. Finally, we illustrate that our method has favorable performance in a sound recording classification problem.

Acknowledgments

We thank Kshitij Khare for a useful discussion in which he pointed us to the parametrization in terms of L. We thank Adam Rothman for providing `R` code for the non-adaptive banding and the nested lasso methods and Amy Williams for useful discussions about linkage disequilibrium. We also thank three referees and an action editor for helpful comments on an earlier manuscript. This work was supported by NSF DMS-1405746.

Appendix A. Decoupling Property

Let $S = \frac{1}{n}\mathbf{X}^T\mathbf{X} \in \mathbb{R}^{p\times p}$ be the sample covariance matrix. Then the estimator (5) is the solution to the following minimization problem:

$$\min_{\substack{L:L_{rr}>0\\ L_{rk}=0 \text{ for } r<k}} \left\{ -2\sum_{r=1}^{p}\log L_{rr} + \operatorname{tr}(SL^TL) + \lambda\sum_{r=2}^{p}\sum_{\ell=1}^{r-1}\sqrt{\sum_{m=1}^{\ell} w_{\ell m}^2 L_{rm}^2} \right\}.$$

First note that under the lower-triangular constraint

$$\operatorname{tr}\left(SL^TL\right) = \frac{1}{n}\sum_{r=1}^{p}\operatorname{tr}\left(\mathbf{X}L_{r\cdot}^TL_{r\cdot}\mathbf{X}^T\right) = \frac{1}{n}\sum_{r=1}^{p}\left\|\mathbf{X}L_{r\cdot}^T\right\|_2^2 = \frac{1}{n}\sum_{r=1}^{p}\left\|\mathbf{X}_{1:r}L_{1:r,r}^T\right\|_2^2,$$

where $\mathbf{X}_{1:r}$ is a matrix of the first r columns of $\mathbf{X}$. Thus

$$\begin{aligned}
&-2\sum_{r=1}^{p}\log L_{rr} + \operatorname{tr}(SL^TL) + \lambda\sum_{r=2}^{p}\sum_{\ell=1}^{r-1}\sqrt{\sum_{m=1}^{\ell} w_{\ell m}^2 L_{rm}^2}\\
=&-2\log L_{11} + \frac{1}{n}\left\|\mathbf{X}_1L_{11}\right\|_2^2 + \sum_{r=2}^{p}\left(-2\log L_{rr} + \frac{1}{n}\left\|\mathbf{X}_{1:r}L_{1:r,r}^T\right\|_2^2 + \lambda\sum_{\ell=1}^{r-1}\sqrt{\sum_{m=1}^{\ell} w_{\ell m}^2 L_{rm}^2}\right).
\end{aligned}$$

Therefore the original problem can be decoupled into p separate problems. In particular, a solution $\hat{L}$ can be written in a row-wise form with

$$\hat{L}_{11} = \operatorname*{arg\,min}_{L_{11}>0}\left\{-2\log L_{11} + \frac{1}{n}\left\|\mathbf{X}_1L_{11}\right\|_2^2\right\} = \frac{1}{\sqrt{S_{11}}},$$

and for $r = 2,\ldots,p$,

$$\hat{L}_{1:r,r}^T = \operatorname*{arg\,min}_{\beta\in\mathbb{R}^r;\beta_r>0}\left\{-2\log\beta_r + \frac{1}{n}\left\|\mathbf{X}_{1:r}\beta\right\|_2^2 + \lambda\sum_{\ell=1}^{r-1}\sqrt{\sum_{m=1}^{\ell} w_{\ell m}^2\beta_m^2}\right\}.$$

Appendix B. A Closed-Form Solution to (9)

The objective function in (9) is a smooth function. Taking the derivative with respect to β and setting to zero gives the following system of equations:

$$-2\frac{1}{\beta_r}\mathbf{e}_r + \frac{2}{n}\mathbf{X}_{1:r}^T\mathbf{X}_{1:r}\beta + u^{(t-1)} + \rho\left(\beta - \gamma^{(t-1)}\right) = \mathbf{0}.$$

Letting $S^{(r)} = \frac{1}{n}\mathbf{X}_{1:r}^T\mathbf{X}_{1:r}$, then the equations above can be further decomposed into

$$\begin{aligned}
&-\frac{2}{\beta_r} + \left(2S_{rr}^{(r)} + \rho\right)\beta_r + 2S_{r,-r}^{(r)}\beta_{-r} + u_r^{(t-1)} - \rho\gamma_r^{(t-1)} = 0,\\
&\left(2S_{-r,-r}^{(r)} + \rho I\right)\beta_{-r} + 2S_{-r,r}^{(r)}\beta_r + u_{-r}^{(t-1)} - \rho\gamma_{-r}^{(t-1)} = \mathbf{0}.
\end{aligned}$$

Solving for β_{-r} in the second system of equations gives

$$\beta_{-r} = -\left(2S_{-r,-r}^{(r)} + \rho I\right)^{-1}\left(2S_{-r,r}^{(r)}\beta_r + u_{-r}^{(t-1)} - \rho\gamma_{-r}^{(t-1)}\right),$$

which is then plugged back in the first equation to give

$$2\frac{1}{\beta_r} + A\beta_r + B = 0,$$

where

$$\begin{aligned}
A &= 4S_{r,-r}^{(r)}\left(2S_{-r,-r}^{(r)} + \rho I\right)^{-1}S_{-r,r}^{(r)} - 2S_{r,r}^{(r)} - \rho,\\
B &= 2S_{r,-r}^{(r)}\left(2S_{-r,-r}^{(r)} + \rho I\right)^{-1}\left(u_{-r}^{(t-1)} - \rho\gamma_{-r}^{(t-1)}\right) - u_r^{(t-1)} + \rho\gamma_r^{(t-1)}.
\end{aligned}$$

Solving for β_r gives the closed-form update.

Appendix C. Dual Problem of (10)

Lemma 7 *A dual problem of* (10) *is*

$$\min_{a^{(\ell)}\in\mathbb{R}^r}\left\{\left\|y^{(t)} - \frac{\lambda}{\rho}\sum_{\ell=1}^{r-1}W^{(\ell)} * a^{(\ell)}\right\|_2^2 \quad \text{s.t.} \quad \left\|\left(a^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 \le 1, \quad \left(a^{(\ell)}\right)_{g_{r,\ell}^c} = 0\right\}, \tag{28}$$

where $y^{(t)} = \beta^{(t)} + \frac{1}{\rho}u^{(t-1)}$. *Also, given a solution* $\hat{a}^{(1)},\ldots,\hat{a}^{(r-1)}$, *the solution to* (10) *can be written as*

$$\gamma^{(t)} = y^{(t)} - \frac{\lambda}{\rho}\sum_{\ell=1}^{r-1}W^{(\ell)} * \hat{a}^{(\ell)}. \tag{29}$$

Proof Note that

$$\begin{aligned}
\sqrt{\sum_{m=1}^{\ell} w_{\ell m}^2\gamma_m^2} &= \left\|\left(W^{(\ell)} * \gamma\right)_{g_{r,\ell}}\right\|_2\\
&= \max\left\{\left\langle W^{(\ell)} * a^{(\ell)}, \gamma\right\rangle, \quad \text{s.t.} \quad \left\|\left(a^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 \le 1, \quad \left(a^{(\ell)}\right)_{g_{r,\ell}^c} = 0\right\}.
\end{aligned}$$

Thus, the minimization problem in (10) becomes

$$\begin{aligned}
&\min_{\gamma}\left\{\frac{1}{2}\left\|\gamma - y^{(t)}\right\|_2^2 + \frac{\lambda}{\rho}\sum_{\ell=1}^{r-1}\left\|\left(W^{(\ell)} * \gamma\right)_{g_{r,\ell}}\right\|_2\right\}\\
=&\min_{\gamma}\left\{\max_{a^{(\ell)}}\left\{\frac{1}{2}\left\|\gamma - y^{(t)}\right\|_2^2 + \frac{\lambda}{\rho}\sum_{\ell=1}^{r-1}\left\langle W^{(\ell)} * a^{(\ell)}, \gamma\right\rangle, \left\|\left(a^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 \le 1, \left(a^{(\ell)}\right)_{g_{r,\ell}^c} = 0\right\}\right\}\\
=&\max_{a^{(\ell)}}\left\{\min_{\gamma}\left\{\frac{1}{2}\left\|\gamma - y^{(t)}\right\|_2^2 + \frac{\lambda}{\rho}\sum_{\ell=1}^{r-1}\left\langle W^{(\ell)} * a^{(\ell)}, \gamma\right\rangle, \left\|\left(a^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 \le 1, \left(a^{(\ell)}\right)_{g_{r,\ell}^c} = 0\right\}\right\},
\end{aligned}$$

where $y^{(t)} = \beta^{(t)} + \frac{1}{\rho}u^{(t-1)}$. We solve the inner minimization problem by setting the derivative to zero,

$$\gamma - y^{(t)} + \frac{\lambda}{\rho}\sum_{\ell=1}^{r-1} W^{(\ell)} * a^{(\ell)} = 0,$$

which gives the primal-dual relation,

$$\gamma = -\frac{\lambda}{\rho}\sum_{\ell=1}^{r-1} W^{(\ell)} * a^{(\ell)} + y^{(t)}.$$

Using this gives

$$\begin{aligned}
&\min_{\gamma}\left\{\frac{1}{2}\left\|\gamma - y^{(t)}\right\|_2^2 + \frac{\lambda}{\rho}\sum_{\ell=1}^{r-1}\left\|\left(W^{(\ell)} * \gamma\right)_{g_{r,\ell}}\right\|_2\right\} \\
&= \max_{a^{(\ell)}}\left\{\frac{1}{2}\left\|-\frac{\lambda}{\rho}\sum_{\ell=1}^{r-1} W^{(\ell)} * a^{(\ell)}\right\|_2^2 + \frac{\lambda}{\rho}\sum_{\ell=1}^{r-1}\left\langle W^{(\ell)} * a^{(\ell)}, -\frac{\lambda}{\rho}\sum_{\ell=1}^{r-1} W^{(\ell)} * a^{(\ell)} + y^{(t)}\right\rangle\right. \\
&\qquad \left.\text{s.t.} \quad \left\|\left(a^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 \le 1, \quad \left(a^{(\ell)}\right)_{g^c_{r,\ell}} = 0\right\} \\
&= \min_{a^{(\ell)}}\left\{\left\|y^{(t)} - \frac{\lambda}{\rho}\sum_{\ell=1}^{r-1} W^{(\ell)} * a^{(\ell)}\right\|_2^2 \quad \text{s.t.} \quad \left\|\left(a^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 \le 1, \quad \left(a^{(\ell)}\right)_{g^c_{r,\ell}} = 0\right\}.
\end{aligned}$$

■

Algorithm 3 BCD on the dual problem (28)

1: Let $y^{(t)} = \beta^{(t)} + \frac{1}{\rho}u^{(t-1)}$
2: Initialize $\hat{a}^{(\ell)} \leftarrow 0$ for all $\ell = 1, \cdots, r-1$
3: **for** $\ell = 1, \cdots, r-1$ **do**
4: $\quad \hat{z}^{(\ell)} \leftarrow y^{(t)} - \frac{\lambda}{\rho}\sum_{k=1}^{r-1} W^{(k)} * \hat{a}^{(k)}$ Find a root $\hat{\nu}_\ell$ that satisfies

$$h_\ell(\nu) := \sum_{m=1}^{\ell} \frac{w_{\ell m}^2}{\left(w_{\ell m}^2 + \nu\right)^2}\left(\hat{z}_m^{(\ell)}\right)^2 = \frac{\lambda^2}{\rho^2} \tag{30}$$

5: $\quad$ **for** $m = 1, \cdots, \ell$ **do**
6: $\quad\quad \hat{a}_m^{(\ell)} \leftarrow \frac{w_{\ell m}}{\frac{\lambda}{\rho}\left(w_{\ell m}^2 + [\hat{\nu}_\ell]_+\right)}\hat{z}_m^{(\ell)}$
7: **return** $\left\{\hat{a}^{(\ell)}\right\}$ as a solution to (28)
8: **return** $\gamma^{(t)} = y^{(t)} - \frac{\lambda}{\rho}\sum_{\ell=1}^{r-1} W^{(\ell)} * \hat{a}^{(\ell)}$ as a solution to (10)

Appendix D. Elliptical Projection

We adapt the same procedure as in Appendix B of Bien et al. (2016) to update one $a^{(\ell)}$ in Algorithm (3). By (10) we need to solve a problem of the form

$$\min_{a\in\mathbb{R}^\ell}\left\|\hat{z}^{(\ell)} - \tau D a\right\|_2^2 \quad \text{s.t.} \quad \|a\|_2 \le 1,$$

where $\tau = \frac{\lambda}{\rho}$ and $D = \operatorname{diag}(w_{\ell m})_{m\le\ell} \in \mathbb{R}^{\ell\times\ell}$. If $\left\|D^{-1}\hat{z}^{(\ell)}\right\|_2 \le \tau$, then clearly $\hat{a} = \frac{1}{\tau}D^{-1}\hat{z}^{(\ell)}$. Otherwise, we use the Lagrangian multiplier method to solve the constrained minimization problem above. Specifically, we find a stationary point of

$$\mathcal{L}(a,\nu) = \left\|\hat{z}^{(\ell)} - \tau D a\right\|_2^2 + \nu\tau^2\left(\|a\|_2^2 - 1\right).$$

Taking the derivative with respect to a and set it equal to zero, we have

$$\hat{a}_m = \frac{w_{\ell m}}{\tau(w_{\ell m}^2 + \hat{\nu})}\hat{z}_m^{(\ell)},$$

for each $m \le \ell$, and $\hat{\nu}$ is such that $\|\hat{a}\|_2 = 1$, which means it satisfies (30). By observing that $h_\ell(\nu)$ is a decreasing function of ν and $w_{\ell\ell} = \max_{m\le\ell} w_{\ell m}$, following Appendix B of Bien et al. (2016), we obtain lower and upper bounds for $\hat{\nu}$:

$$\left[\frac{1}{\tau}\left\|D\hat{z}^{(\ell)}\right\|_2 - w_{\ell\ell}^2\right]_+ \le \hat{\nu} \le \frac{1}{\tau}\left\|D\hat{z}^{(\ell)}\right\|_2,$$

which can be used as an initial interval for finding $\hat{\nu}$ using Newton's method. In practice, we usually find $\hat{\nu}$ from the equation $\frac{1}{h(\nu)} = \tau^{-2}$ for better numerical stability.

We end this section with a characterization of the solution to (10), which says that the solution can be written as $\gamma^{(t)} = y^{(t)} * \hat{t}$, where $\hat{t}$ is some data-dependent vector in $\mathbb{R}^r$.

Theorem 8 *A solution to* (10) *can be written as* $\gamma^{(t)} = y^{(t)} * \hat{g}$*, where the data-dependent vector* $\hat{g} \in \mathbb{R}^r$ *is given by*

$$\hat{g}_m = \prod_{\ell=m}^{r-1}\frac{[\hat{\nu}_\ell]_+}{w_{\ell m}^2 + [\hat{\nu}_\ell]_+}$$

and $\hat{g}_r = 1$*, where* $\hat{\nu}_\ell$ *satisfies* $\tau^2 = \sum_{m=1}^{\ell}\frac{w_{\ell m}^2}{\left(w_{\ell m}^2+\nu\right)^2}\left(\hat{z}_m^{(\ell)}\right)^2$.

Proof By Jenatton et al. (2011), we can get a solution to (10) in a single pass as described in Algorithm 3. If we start from $\hat{z}^{(1)} = y^{(t)}$, then for $\ell = 1, \cdots, r-1$ and each $m \le \ell$,

$$\hat{z}_m^{(\ell+1)} = \hat{z}_m^{(\ell)} - \tau w_{\ell m}\hat{a}_m^{(\ell)} = \frac{[\hat{\nu}_\ell]_+}{w_{\ell m}^2 + [\hat{\nu}_\ell]_+}\hat{z}_m^{(\ell)}.$$

By (29), $\gamma^{(t)} = \hat{z}^{(r-1)}$, and the result follows. ■

A key observation from this characterization is that a banded sparsity pattern is induced in solving (10), which in turn implies the same property of the output of Algorithm 1.

Corollary 9 *A solution* $\gamma^{(t)}$ *to* (10) *has banded sparsity, i.e.,* $\left(\gamma^{(t)}\right)_{1:\hat{J}} = 0$ *for* $\hat{J} = \max\{\ell : \hat{\nu}_\ell \le 0\}$.

Appendix E. Uniqueness of the Sparse Row Estimator

Lemma 10 *(Optimality condition) For any $\lambda > 0$ and a n-by-p sample matrix $\mathbf{X}$, $\hat{\beta}$ is a solution to the problem*

$$\min_{\beta \in \mathbb{R}^r} \left\{ -2\log \beta_r + \frac{1}{n} \|\mathbf{X}_{1:r}\beta\|_2^2 + \lambda \sum_{\ell=1}^{r-1} \sqrt{\sum_{m=1}^{\ell} w_{\ell m}^2 \beta_m^2} \right\}$$

if and only if there exist $\hat{a}^{(\ell)} \in \mathbb{R}^r$ for $\ell = 1, \ldots, r-1$ such that

$$-\frac{2}{\hat{\beta}_r}\mathbf{e}_r + \frac{2}{n}\mathbf{X}_{1:r}^T\mathbf{X}_{1:r}\hat{\beta} + \lambda \sum_{\ell=1}^{r-1} W^{(\ell)} * \hat{a}^{(\ell)} = 0 \tag{31}$$

with $\left(\hat{a}^{(\ell)}\right)_{g_{r,\ell}^c} = 0$, $\left(\hat{a}^{(\ell)}\right)_{g_{r,\ell}} = \frac{\left(W^{(\ell)} * \hat{\beta}\right)_{g_{r,\ell}}}{\left\|\left(W^{(\ell)} * \hat{\beta}\right)_{g_{r,\ell}}\right\|_2}$ for $\hat{\beta}_{g_{r,\ell}} \neq 0$ and $\left\|\left(\hat{a}^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 \leq 1$ for $\hat{\beta}_{g_{r,\ell}} = 0$.

Lemma 11 *Take $\hat{\beta}$ and $\hat{a}^{(\ell)}$ as in the previous lemma. Suppose that*

$$\left\|\left(\hat{a}^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 < 1 \qquad \text{for} \quad \ell = 1, \ldots, J(\hat{\beta})$$

then for any other solution $\tilde{\beta}$ to (8), it is as sparse as $\hat{\beta}$ if not more. In other words,

$$K(\tilde{\beta}) \leq \hat{K}_r.$$

Lemma 12 *(Uniqueness) Under the conditions of the previous lemma, let $\hat{\mathcal{S}} = \left\{ i : \hat{\beta}_i \neq 0 \right\}$. If $\mathbf{X}_{\hat{\mathcal{S}}}$ has full column rank (i.e., $\operatorname{rank}\left(\mathbf{X}_{\hat{\mathcal{S}}}\right) = |\hat{\mathcal{S}}|$) then $\hat{\beta}$ is unique.*

Proof See Appendices J, K, and L. ■

Appendix F. Proof of Theorem 1

We start with introducing notation. **From now on we suppress the dependence on r in notation for simplicity**. We denote the group structure $g_\ell = \{1, \cdots, \ell\}$ for $\ell \leq r$ for each $r = 1, \ldots, p$. For any vector $\beta \in \mathbb{R}^r$, we let $\beta_{g_\ell} \in \mathbb{R}^\ell$ be the vector with elements $\{\beta_m : m \leq \ell\}$. We also introduce the weight vector $W^{(\ell)} \in \mathbb{R}^p$ with $\left(W^{(\ell)}\right)_m = w_{\ell m}$ where $w_{\ell m}$ can be defined as in (7) or $w_{\ell m} = 1$. Finally recalling from Section 4 the definition of $\mathcal{I}$, we denote $\mathcal{S} = \mathcal{I} \cup \{r\} = \{J+1, \ldots, r\}$ and $\mathcal{S}^c = \{1, 2, \ldots, J\}$.

The general idea of the proof depends on the primal-dual witness procedure in Wainwright (2009) and Ravikumar et al. (2011). Considering the original problem (8) for any $r = 2, \ldots, p$, we construct the primal-dual witness solution pairs $\left(\tilde{\beta}, \sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)$ as follows:

(a) Solve the restricted subproblem with the true bandwidth $K = r - 1 - J$:

$$\tilde{\beta} = \underset{\substack{\beta_r > 0 \\ \beta_{\mathcal{S}^c} = 0}}{\arg\min} \left\{ -2\log \beta_r + \frac{1}{n}\|\mathbf{X}_{1:r}\beta\|_2^2 + \lambda \sum_{\ell=1}^{r-1} \left\| \left(W^{(\ell)} * \beta \right)_{g_\ell} \right\|_2 \right\}.$$

The solution above can be written as

$$\tilde{\beta} = \begin{pmatrix} \mathbf{0}_J \\ \tilde{\gamma} \end{pmatrix},$$

where

$$\tilde{\gamma} = \underset{\gamma \in \mathbb{R}^{K+1}}{\arg\min} \left\{ -2\log \gamma_{K+1} + \frac{1}{n}\|\mathbf{X}_{\mathcal{S}}\gamma\|_2^2 + \lambda \sum_{\ell=1}^{K} \left\| \left(\tilde{W}^{(\ell)} * \gamma \right)_{g_\ell} \right\|_2 \right\},$$

with

$$\tilde{W}^{(\ell)} = \left(W^{(\ell+J)} \right)_{\mathcal{S}} \quad \Longleftrightarrow \quad \sum_{\ell=1}^{K} \left\| \left(\tilde{W}^{(\ell)} * \gamma \right)_{g_\ell} \right\|_2 = \sum_{\ell=J+1}^{r-1} \sqrt{\sum_{m=J+1}^{r-1} w_{\ell m}^2 \gamma_{m-J}^2}.$$

(b) By Lemma 10, there exist $\tilde{b}^{(\ell)} \in \mathbb{R}^{K+1}$ for $\ell = 1, \ldots, K$, such that $\left(\tilde{b}^{(\ell)}\right)_{g_\ell^c} = 0$ and

$$\left(\tilde{b}^{(\ell)}\right)_{g_{r\ell}} = \frac{\left(\tilde{W}^{(\ell)} * \tilde{\gamma}\right)_{g_\ell}}{\left\| \left(\tilde{W}^{(\ell)} * \tilde{\gamma}\right)_{g_\ell} \right\|_2},$$

satisfying

$$-\frac{2}{\tilde{\gamma}_{K+1}}\mathbf{e}_{K+1} + \frac{2}{n}\mathbf{X}_{\mathcal{S}}^T\mathbf{X}_{\mathcal{S}}\tilde{\gamma} + \lambda \sum_{\ell=1}^{K} \tilde{W}^{(\ell)} * \tilde{b}^{(\ell)} = 0.$$

(c) For $\ell = J+1, \ldots, r-1$, we let

$$\tilde{a}^{(\ell)} = \begin{pmatrix} \mathbf{0}_J \\ \tilde{b}^{(\ell-J)} \end{pmatrix}.$$

Then we have $\left(\tilde{a}^{(\ell)}\right)_{g_\ell^c} = 0$, $\left\|\left(\tilde{a}^{(\ell)}\right)_{g_\ell}\right\|_2 \leq 1$, $\left(\tilde{a}^{(\ell)}\right)_{g_\ell} = \frac{\left(W^{(\ell)} * \tilde{\beta}\right)_{g_\ell}}{\left\|\left(W^{(\ell)} * \tilde{\beta}\right)_{g_\ell}\right\|_2}$ for $\tilde{\beta}_{g_\ell} \neq 0$.

(d) For each $\ell = 1, ..., J$, we choose $\tilde{a}^{(\ell)} \in \mathbb{R}^r$ satisfying

$$\left(\tilde{a}^{(\ell)}\right)_{\ell'} = 0 \quad \text{for any } \ell' \neq \ell \quad \text{and} \quad \left(\tilde{a}^{(\ell)}\right)_\ell = -\frac{2}{\lambda w_{\ell\ell}}\left(S\tilde{\beta}\right)_\ell = -\frac{2}{n\lambda}\mathbf{X}_\ell^T\mathbf{X}_{\mathcal{S}}\tilde{\beta}_{\mathcal{S}}.$$

By construction and the fact that $w_{\ell\ell} = 1$,

$$\lambda\left(W^{(\ell)} * \tilde{a}^{(\ell)}\right)_\ell = \lambda w_{\ell\ell}\left(\tilde{a}^{(\ell)}\right)_\ell = -2\left(S\tilde{\beta}\right)_\ell.$$

By Lemma 10, $\{\tilde{a}^{(\ell)}\}$ satisfies the optimality condition (31):

$$-\frac{2}{\tilde{\beta}_r}\mathbf{e}_r + \frac{2}{n}\mathbf{X}_{1:r}^T\mathbf{X}_{1:r}\tilde{\beta} + \lambda\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)} = 0 \tag{32}$$

(e) Verify the strict dual feasibility condition for $\ell = 1, ..., J$

$$\left|\frac{2}{n\lambda}\mathbf{X}_\ell^T\mathbf{X}_{\mathcal{S}}\tilde{\beta}_{\mathcal{S}}\right| = \left|\left(\tilde{a}^{(\ell)}\right)_\ell\right| = \left\|\left(\tilde{a}^{(\ell)}\right)_{g_\ell}\right\|_2 < 1. \tag{33}$$

At a high level, steps (a) through (d) construct a pair $\left(\tilde{\beta}, \{\tilde{a}^{(\ell)}\}\right)$ that satisfies the optimality condition (31), but the $\{\tilde{a}^{(\ell)}\}$ is not necessarily guaranteed to be a member of $\partial\left(P(\tilde{\beta})\right)$. Step (e) does more than verifying the necessary conditions for it to belong to $\partial\left(P(\tilde{\beta})\right)$. The strict dual feasibility condition, once verified, ensures the uniqueness of the solution. Note that by construction in Step (b), $\{\tilde{a}^{(\ell)}\}$ satisfies dual feasibility conditions for $\ell = J+1, ..., r-1$ since $\left\{\tilde{b}^{(\ell)}\right\}$ does, so it remains to verify for $\ell = 1, ..., J$ (see Step (c)).

For each $\ell = 1, ..., J$, by the construction in Step (d), $\left(\tilde{a}^{(\ell)}\right)_{g_\ell^c} = 0$. Note that $\tilde{\beta}_{g_J} = 0$ implies $\tilde{\beta}_{g_\ell} = 0$. Thus, for $\tilde{a}^{(\ell)}$ to satisfy conditions in Lemma 10, it suffices to show (33).

If the primal-dual witness procedure succeeds, then by construction, the solution $\tilde{\beta}$, whose support is contained in the support of the true $L_{r\cdot}$, is a solution to (8). Moreover, by strict dual feasibility and Lemma 12, we know that $\tilde{\beta}$ is the unique solution $\hat{\beta}$ to the unconstrained problem (8). Therefore, the support of $\hat{\beta}$ is contained in the support of $L_{r\cdot}$.

In the following we adapt the same proof technique as Wainwright (2009) to show that the primal-dual witness succeeds with high probability, from which we first conclude that $K(\hat{\beta}) \leq K$.

F.1 Proof of Property 1 in Theorem 1

Proof We need to verify the strict dual feasibility (33). By (32),

$$-\frac{2}{\tilde{\beta}_r} + \frac{2}{n}\mathbf{X}_r^T\mathbf{X}_r\tilde{\beta}_r + \frac{2}{n}\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\tilde{\beta}_{\mathcal{I}} = 0, \tag{34}$$

$$\frac{2}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_r\tilde{\beta}_r + \frac{2}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\tilde{\beta}_{\mathcal{I}} + \lambda\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}} = 0. \tag{35}$$

From (35),

$$\tilde{\beta}_{\mathcal{I}} = -\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left[\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_r\tilde{\beta}_r + \frac{\lambda n}{2}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}}\right]. \tag{36}$$

Plugging (36) back into (34) and denoting $\mathbf{C}_{\mathcal{I}} = \mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}}$ and $\mathbf{O}_{\mathcal{I}} = \mathbf{I} - \mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathbf{X}_{\mathcal{I}}^T$ as the orthogonal projection matrix onto the orthogonal

complement of the column space of $\mathbf{X}_{\mathcal{I}}$, we have

$$-\frac{2}{\tilde{\beta}_r} + \frac{2}{n}\mathbf{X}_r^T\mathbf{O}_{\mathcal{I}}\mathbf{X}_r\tilde{\beta}_r - \lambda\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}} = 0,$$

which implies that

$$\tilde{\beta}_r = \frac{\frac{\lambda}{2}\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}} + \sqrt{\frac{\lambda^2}{4}\left(\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}}\right)^2 + \frac{4}{n}\mathbf{X}_r^T\mathbf{O}_{\mathcal{I}}\mathbf{X}_r}}{\frac{2}{n}\mathbf{X}_r^T\mathbf{O}_{\mathcal{I}}\mathbf{X}_r} \tag{37}$$

and that

$$\begin{aligned}
\left(\tilde{a}^{(\ell)}\right)_\ell &= -\frac{2}{n\lambda}\mathbf{X}_\ell^T\mathbf{X}_{\mathcal{S}}\tilde{\beta}_{\mathcal{S}} = -\frac{2}{n\lambda}\mathbf{X}_\ell^T\mathbf{X}_r\tilde{\beta}_r - \frac{2}{n\lambda}\mathbf{X}_\ell^T\mathbf{X}_{\mathcal{I}}\tilde{\beta}_{\mathcal{I}} \\
&= -\frac{2}{n\lambda}\mathbf{X}_\ell^T\mathbf{X}_r\tilde{\beta}_r + \frac{2}{n\lambda}\mathbf{X}_\ell^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left[\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_r\tilde{\beta}_r + \frac{\lambda n}{2}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}}\right] \\
&= -\frac{2}{n\lambda}\mathbf{X}_\ell^T\left[\mathbf{I} - \mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathbf{X}_{\mathcal{I}}^T\right]\mathbf{X}_r\tilde{\beta}_r + \mathbf{X}_\ell^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}} \\
&= \mathbf{X}_\ell^T\left[\mathbf{C}_{\mathcal{I}} - \mathbf{O}_{\mathcal{I}}\left(\frac{2}{n\lambda}\mathbf{X}_r\tilde{\beta}_r\right)\right].
\end{aligned} \tag{38}$$

Conditioning on $\mathbf{X}_{\mathcal{I}}$, we can decompose $\mathbf{X}_r$ and $\mathbf{X}_\ell$ as

$$\begin{aligned}
\mathbf{X}_r^T &= \Sigma_{r\mathcal{I}}\left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1}\mathbf{X}_{\mathcal{I}}^T + E_r^T, \\
\mathbf{X}_\ell^T &= \Sigma_{\ell\mathcal{I}}\left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1}\mathbf{X}_{\mathcal{I}}^T + E_\ell^T,
\end{aligned} \tag{39}$$

where $E_r \sim N\left(\mathbf{0}_n, \theta_r^{(r)}\mathbf{I}_{n\times n}\right)$ and $E_\ell \sim N\left(\mathbf{0}_n, \theta_r^{(\ell)}\mathbf{I}_{n\times n}\right)$, and $\theta_r^{(\ell)}$ and $\theta_r^{(r)}$ are defined in Section 4. Then

$$\mathbf{X}_\ell^T\mathbf{O}_{\mathcal{I}} = E_\ell^T\mathbf{O}_{\mathcal{I}} \quad \text{and} \quad \mathbf{O}_{\mathcal{I}}\mathbf{X}_r = \mathbf{O}_{\mathcal{I}}E_r,$$

and from (38)

$$\begin{aligned}
\left(\tilde{a}^{(\ell)}\right)_\ell &= E_\ell^T\left[\mathbf{C}_{\mathcal{I}} - \mathbf{O}_{\mathcal{I}}\left(\frac{2}{n\lambda}E_r\tilde{\beta}_r\right)\right] + \Sigma_{\ell\mathcal{I}}\left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}} \\
&:= R^{(\ell)} + F^{(\ell)}.
\end{aligned} \tag{40}$$

We first bound $\max_\ell \left|F^{(\ell)}\right|$. Note that

$$\begin{aligned}
&\left\|\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}}\right\|_\infty = \left\|\left(\sum_{\ell=J+1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}}\right\|_\infty = \max_{m\in\mathcal{I}}\left|\sum_{\ell=m}^{r-1} w_{\ell m}\left(\tilde{a}^{(\ell)}\right)_m\right| \\
\leq &\max_{m\in\mathcal{I}}\sum_{\ell=m}^{r-1} w_{\ell m}\left|\left(\tilde{a}^{(\ell)}\right)_m\right| \leq \max_{m\in\mathcal{I}}\sum_{\ell=m}^{r-1}\frac{1}{(\ell-m+1)^2} \leq \sum_{k=1}^{\infty}\frac{1}{k^2} = \frac{\pi^2}{6},
\end{aligned} \tag{41}$$

where we used $\left\|\tilde{a}^{(\ell)}\right\|_\infty \le \left\|\tilde{a}^{(\ell)}\right\|_2 \le 1$. Therefore, by Assumption **A3**,

$$\max_{1\le\ell\le J}\left|\Sigma_{\ell\mathcal{I}}\left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1}W^{(\ell)}*\tilde{a}^{(\ell)}\right)_{\mathcal{I}}\right| \le 1-\alpha.$$

To give a bound on the random quantity $\left|R^{(\ell)}\right|$, we first state a general result that will be used multiple times later in the proof.

Lemma 13 *Consider the term $E_j^T\eta$ where $\eta\in\mathbb{R}^n$ is a random vector depending on $\mathbf{X}_\mathcal{I}$ and $\mathbf{X}_r$ and $E_j\sim N\left(\mathbf{0}_n,\theta_r^{(j)}\mathbf{I}_{n\times n}\right)$ for $j=1,\ldots,J,r$. If for some $\bar{Q}\ge 0$*

$$\mathrm{P}\left[\mathrm{Var}\left(E_j^T\eta\middle|\mathbf{X}_\mathcal{I},\mathbf{X}_r\right)\ge\bar{Q}\right]\le\bar{p}$$

then for any $a>0$,

$$\mathrm{P}\left[\left|E_j^T\eta\right|\ge a\right]\le 2\exp\left(-\frac{a^2}{2\bar{Q}}\right)+\bar{p}$$

Proof Define the event

$$\bar{\mathcal{B}}=\left\{\mathrm{Var}\left(E_j^T\eta\middle|\mathbf{X}_\mathcal{I}\right)\ge\bar{Q}\right\}.$$

Now for any a and conditioned on $\mathbf{X}_\mathcal{I}$ and $\mathbf{X}_r$,

$$\mathrm{P}\left[E_j^T\eta\ge a\right]\le\mathrm{P}\left[E_j^T\eta\ge a\middle|\bar{\mathcal{B}}^c\right]+\mathrm{P}\left[\bar{\mathcal{B}}\right]\le\mathrm{P}\left[E_j^T\eta\ge a\middle|\bar{\mathcal{B}}^c\right]+\bar{p}.$$

Conditioned on $\bar{\mathcal{B}}^c$, the variance of $E_j^T\eta$ is at most $\bar{Q}$. So by standard Gaussian tail bounds, we have

$$\mathrm{P}\left[E_j^T\eta\ge a\middle|\bar{\mathcal{B}}^c\right]=\mathrm{E}\left[\mathrm{P}\left(E_j^T\eta\ge a\middle|\mathbf{X}_\mathcal{I},\mathbf{X}_r\right)\middle|\bar{\mathcal{B}}^c\right]\le\mathrm{E}\left[2\exp\left(-\frac{a^2}{2\bar{Q}}\right)\middle|\bar{\mathcal{B}}^c\right]\le 2\exp\left(-\frac{a^2}{2\bar{Q}}\right).$$

■

Then note that $\mathrm{Var}\left(E_{i\ell}\right)=\theta_r^{(\ell)}\le\theta_r$ for $i=1,\ldots,n$. Now conditioned on both $\mathbf{X}_\mathcal{I}$ and $\mathbf{X}_r$, $R^{(\ell)}$ is zero-mean with variance at most

$$\begin{aligned}
&\mathrm{Var}\left(R^{(\ell)}\middle|\mathbf{X}_\mathcal{I}\right)\\
\le&\theta_r\left\|\mathbf{C}_\mathcal{I}-\mathbf{O}_\mathcal{I}\left(\frac{2}{n\lambda}E_r\tilde{\beta}_r\right)\right\|_2^2=\theta_r\left\{\mathbf{C}_\mathcal{I}^T\mathbf{C}_\mathcal{I}+\left\|\mathbf{O}_\mathcal{I}\left(\frac{2}{n\lambda}E_r\tilde{\beta}_r\right)\right\|_2^2\right\}\\
=&\theta_r\left\{\frac{1}{n}\left(\sum_{\ell=1}^{r-1}W^{(\ell)}*\tilde{a}^{(\ell)}\right)_\mathcal{I}^T\left(\frac{1}{n}\mathbf{X}_\mathcal{I}^T\mathbf{X}_\mathcal{I}\right)^{-1}\left(\sum_{\ell=1}^{r-1}W^{(\ell)}*\tilde{a}^{(\ell)}\right)_\mathcal{I}+\frac{4\tilde{\beta}_r^2\left\|\mathbf{O}_\mathcal{I}E_r\right\|_2^2}{n^2\lambda^2}\right\}\\
:=&\theta_r M_n,
\end{aligned}$$

where the first equality holds from Pythagorean identity. The next lemma bounds the random scaling M_n.

Lemma 14 *For $\varepsilon\in\left(0,\frac{1}{2}\right)$, denote*

$$\bar{M}_n(\varepsilon):=\frac{3\kappa^2\pi^2}{2}\frac{K}{n}+\frac{1}{\theta_r^{(r)}(n-K)(1-\varepsilon)}+\frac{16}{n\lambda^2},$$

then

$$\mathrm{P}\left[M_n\ge\bar{M}_n(\varepsilon)\middle|\mathbf{X}_\mathcal{I}\right]\le 7\exp\left(-n\min\left\{\frac{\alpha^2}{3\theta_r^{(r)}\kappa^2\pi^2K},\frac{\varepsilon^2}{4}\left(1-\frac{K}{n}\right)\right\}\right).$$

Proof See Appendix M. ■

Now by Lemma 13 and the union bound,

$$\mathrm{P}\left[\max_{1\le\ell\le J}\left|R^{(\ell)}\right|\ge\alpha\right]\le 2J\exp\left(-\frac{\alpha^2}{2\theta_r\bar{M}_n(\varepsilon)}\right)+7\exp\left(-c_3n\right),\tag{42}$$

for some constant c_3 independent of n and J. By the assumption that $\frac{K}{n}=o(1)$, we have that $\frac{K}{n}\le 1-\varepsilon$ for n large enough, thus

$$\bar{M}_n(\varepsilon)\le\frac{K}{n}\left(\frac{3\kappa^2\pi^2}{2}+\frac{1}{K\theta_r^{(r)}(1-\varepsilon)^2}+\frac{16}{K\lambda^2}\right)\le\frac{K}{n}\left(\frac{3\kappa^2\pi^2}{2}+\frac{4}{K\theta_r^{(r)}}+\frac{16}{K\lambda^2}\right).$$

For the exponential term in (42) to have faster decaying rate than the J term, we need

$$\frac{n}{K\log J}>\frac{\theta_r}{\alpha^2}\left(3\kappa^2\pi^2+\frac{8}{K\theta_r^{(r)}}+\frac{32}{K\lambda^2}\right).$$

■

F.2 Proof of Property 2 in Theorem 1

Next we study the ℓ_∞ error bound. The following theorem gives an ℓ_∞ error bound of $\tilde{\beta}$.

Proof Let $\delta = \tilde{\beta} - \beta^* = \tilde{\beta} - \left(L^T\right)_{1:r,r}$ and $\mathcal{W} = SL^T - (L)^{-1}$, then from (35) and the fact that L^{-1} is lower-triangular,

$$\begin{aligned}
\delta_{\mathcal{I}} =& -\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left[\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_r\tilde{\beta}_r + \left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)(L)_{\mathcal{I},r}^T\right] - \frac{n\lambda}{2}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}} \\
=& -\left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left[\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_r\left(\delta_r + \beta_r^*\right) + \left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)(L)_{\mathcal{I},r}^T\right] \\
&-\frac{\lambda}{2}\left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}} \\
=& -\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_r\delta_r - \left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left(SL^T\right)_{\mathcal{I},r} \\
&-\frac{\lambda}{2}\left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}} \\
=& -\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_r\delta_r - \left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r} - \frac{\lambda}{2}\left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}}.
\end{aligned} \tag{43}$$

From (34) and the fact that $\left(L^{-1}\right)_{rr} = \frac{1}{L_{rr}}$,

$$\begin{aligned}
&-\frac{1}{\tilde{\beta}_r} + \frac{1}{n}\mathbf{X}_r^T\mathbf{X}_r\delta_r + \frac{1}{n}\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\delta_{\mathcal{I}} + \frac{1}{n}\mathbf{X}_r^T\mathbf{X}_r\beta_r^* + \frac{1}{n}\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\beta_{\mathcal{I}}^* \\
=& -\frac{1}{\tilde{\beta}_r} + \frac{1}{n}\mathbf{X}_r^T\mathbf{X}_r\delta_r + \frac{1}{n}\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\delta_{\mathcal{I}} + \left(SL^T\right)_{rr} \\
=& \left(L^{-1}\right)_{rr} - \frac{1}{\tilde{\beta}_r} + \frac{1}{n}\mathbf{X}_r^T\mathbf{X}_r\delta_r + \frac{1}{n}\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\delta_{\mathcal{I}} + \mathcal{W}_{rr} \\
=& \frac{\delta_r}{L_{rr}\tilde{\beta}_r} + \frac{1}{n}\mathbf{X}_r^T\mathbf{X}_r\delta_r + \frac{1}{n}\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\delta_{\mathcal{I}} + \mathcal{W}_{rr} = 0.
\end{aligned} \tag{44}$$

Plugging (43) into (44), we have

$$\frac{\delta_r}{L_{rr}\tilde{\beta}_r} + \frac{1}{n}\mathbf{X}_r^T\mathbf{O}_{\mathcal{I}}\mathbf{X}_r\delta_r = \mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r} + \frac{\lambda}{2}\mathbf{X}_r\mathbf{C}_{\mathcal{I}} - \mathcal{W}_{rr},$$

which implies

$$\delta_r = \left(\frac{1}{L_{rr}\tilde{\beta}_r} + \frac{1}{n}\mathbf{X}_r^T\mathbf{O}_{\mathcal{I}}\mathbf{X}_r\right)^{-1}\left[\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r} + \frac{\lambda}{2}\mathbf{X}_r\mathbf{C}_{\mathcal{I}} - \mathcal{W}_{rr}\right].$$

Since $L_{rr} > 0$ and $\tilde{\beta}_r > 0$,

$$\begin{aligned}
|\delta_r| \le& \left|\left(\frac{1}{L_{rr}\tilde{\beta}_r} + \frac{1}{n}\mathbf{X}_r^T\mathbf{O}_{\mathcal{I}}\mathbf{X}_r\right)^{-1}\right|\left(\left|\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r}\right| + \left|\frac{\lambda}{2}\mathbf{X}_r\mathbf{C}_{\mathcal{I}}\right| + |\mathcal{W}_{rr}|\right) \\
\le& \left|\left(\frac{1}{n}\mathbf{X}_r^T\mathbf{O}_{\mathcal{I}}\mathbf{X}_r\right)^{-1}\right|\left(\left|\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r}\right| + \left|\frac{\lambda}{2}\mathbf{X}_r\mathbf{C}_{\mathcal{I}}\right| + |\mathcal{W}_{rr}|\right).
\end{aligned}$$

Now conditioned on $\mathbf{X}_{\mathcal{I}}$, by the decomposition (39), $\left(\frac{1}{n}\mathbf{X}_r^T\mathbf{O}_{\mathcal{I}}\mathbf{X}_r\right)^{-1} = \left(\frac{1}{n}E_r^T\mathbf{O}_{\mathcal{I}}E_r\right)^{-1} = \frac{n}{\|\mathbf{O}_{\mathcal{I}}E_r\|_2^2}$. From Lemma 20, it follows that

$$\mathrm{P}\left[\left(\frac{1}{n}\mathbf{X}_r^T\mathbf{O}_{\mathcal{I}}\mathbf{X}_r\right)^{-1} \ge \frac{1}{\theta_r^{(r)}}\frac{n}{n-K}\frac{1}{1-\varepsilon}\right] \le \exp\left(-\frac{1}{4}(n-K)\varepsilon^2\right).$$

Also, by Lemma 19,

$$\mathrm{P}\left[\left|\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}}\right| \ge 1\right] \le 2\exp\left(-\frac{n\alpha^2}{3\theta_r^{(r)}\kappa^2\pi^2K}\right) + 2\exp\left(-\frac{n}{2}\right).$$

To deal with the rest of terms in (44) that involve $\mathcal{W}$, we introduce the following concentration inequality to control its element-wise infinity norm.

Lemma 15 *Let $\mathcal{W} = SL^T - L^{-1}$. Under Assumptions **A4** and **A5**, there exist constants $C_1, C_2, C_3 > 0$ such that for any $0 < t \le 2\kappa$,*

$$\mathrm{P}\left[\|\mathcal{W}\|_\infty > t\right] \le 2p^2\exp\left(-\frac{C_3nt^2}{\kappa^2}\right) + 4p\exp\left(-\frac{C_1nt}{\kappa^2}\right) + 4p\exp\left(-C_2nt\right).$$

Proof See Appendix N. ■

In terms of the event

$$\mathcal{A} = \left\{\|\mathcal{W}\|_\infty \le \lambda\right\},$$

Lemma 15 states that

$$\mathrm{P}\left[\mathcal{A}^c\right] \le 2p^2\exp\left(-\frac{C_3n\lambda^2}{\kappa^2}\right) + 4p\exp\left(-\frac{C_1n\lambda}{\kappa^2}\right) + 4p\exp\left(-C_2n\lambda\right).$$

The next lemma shows that, on the event $\mathcal{A}$ and with the assumption that $\frac{\lambda^2}{n} = o(1)$, the term $\left|\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r}\right|$ can be bounded by λ with high probability.

Lemma 16 *Using the general weigthing scheme* (7)*, we have*

$$\mathrm{P}\left[\left|\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r}\right| \ge \lambda\middle|\mathcal{A}\right] \le 2\exp\left(-\frac{2n\alpha^2}{9\theta_r^{(r)}\kappa^2K\lambda^2}\right) + 2\exp\left(-\frac{n}{2}\right).$$

Proof Recall that by conditioning on $\mathbf{X}_{\mathcal{I}}$, the decomposition (39) gives

$$\mathbf{X}_r^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r} = \Sigma_{r\mathcal{I}}\left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r} + E_r^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r}.$$

On the event $\mathcal{A}$, by **A3** and (41),

$$\left|\Sigma_{r\mathcal{I}}\left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1}\mathcal{W}_{\mathcal{I},r}\right| \le \left\|\Sigma_{r\mathcal{I}}\left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1}\right\|_\infty\|\mathcal{W}_{\mathcal{I},r}\|_\infty \le \lambda.$$

Note that $\mathrm{Var}(E_{ir}) = \theta_r^{(r)}$ for $i = 1, \ldots, n$. Let $B^{(r)} := E_r^T \mathbf{X}_{\mathcal{I}} \left(\mathbf{X}_{\mathcal{I}}^T \mathbf{X}_{\mathcal{I}}\right)^{-1} \mathcal{W}_{\mathcal{I},r}$, then $B^{(r)}$ has mean zero and variance at most

$$\mathrm{Var}\left(B^{(r)} \middle| \mathbf{X}_{\mathcal{I}}\right) = \frac{\theta_r^{(r)}}{n} \mathcal{W}_{\mathcal{I},r}^T \left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T \mathbf{X}_{\mathcal{I}}\right)^{-1} \mathcal{W}_{\mathcal{I},r} \leq \frac{9\theta_r^{(r)} \kappa^2 K \lambda^2}{n},$$

with probability greater than $1 - 2\exp\left(\frac{n}{2}\right)$. The result follows from Lemma 13. ■

Putting everything together and choosing the tuning parameter from (13), with a union bound argument and some algebra, we have shown that conditioned on $\mathbf{X}_{\mathcal{I}}$,

$$\begin{aligned}
&\mathrm{P}\left[|\delta_r| \geq \frac{1}{\theta_r^{(r)}} \frac{n}{n-K} \frac{1}{1-\varepsilon} \frac{5}{2}\lambda\right] \leq \mathrm{P}\left[|\delta_r| \geq \frac{5}{2\theta_r^{(r)}}\lambda\right] \leq \mathrm{P}\left[|\delta_r| \geq \frac{5}{2\theta_r^{(r)}}\lambda \middle| \mathcal{A}\right] + \mathrm{P}\left[\mathcal{A}^c\right] \\
&\leq \exp\left(-\frac{1}{4n}\left(1 - \frac{K}{n}\right)\varepsilon^2\right) + 2\exp\left(-\frac{n\alpha^2}{3\theta_r \kappa^2 \pi^2 K}\right) + 2\exp\left(-\frac{2n\alpha^2}{9\theta_r \kappa^2 K \lambda^2}\right) + 4\exp\left(-\frac{n}{2}\right) \\
&+ 2p^2 \exp\left(-\frac{C_3 n \lambda^2}{\kappa^2}\right) + 4p\exp\left(-\frac{C_1 n \lambda}{\kappa^2}\right) + 4p\exp\left(-C_2 n \lambda\right) \\
&\leq c_4 \exp\left(-c_5 n\right) + \frac{c_6}{p}, \qquad (45)
\end{aligned}$$

for some constants $c_4, c_5, c_6, x > 0$ that do not depend on n and p.

We now consider a bound for $\delta_{\mathcal{I}}$. Recall from (43) that

$$\delta_{\mathcal{I}} = F_1 + F_2$$

where

$$\begin{aligned}
F_1 &= -\left(\mathbf{X}_{\mathcal{I}}^T \mathbf{X}_{\mathcal{I}}\right)^{-1} \mathbf{X}_{\mathcal{I}}^T \mathbf{X}_r \delta_r, \\
F_2 &= -\left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T \mathbf{X}_{\mathcal{I}}\right)^{-1} \left(\mathcal{W}_{\mathcal{I},r} + \frac{\lambda}{2}\mathbf{D}\right) \quad \text{with} \quad \mathbf{D} = \left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}}.
\end{aligned}$$

An ℓ_∞ bound of F_2 is given by

$$\|F_2\|_\infty \leq \left\| \left(\left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T \mathbf{X}_{\mathcal{I}}\right)^{-1} - \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1} \right) \left(\mathcal{W}_{\mathcal{I},r} + \frac{\lambda}{2}\mathbf{D}\right) \right\|_\infty + \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1} \left(\mathcal{W}_{\mathcal{I},r} + \frac{\lambda}{2}\mathbf{D}\right) \right\|_\infty. \qquad (46)$$

On the event $\mathcal{A}$, by (41),

$$\begin{aligned}
&\left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1} \left(\mathcal{W}_{\mathcal{I},r} + \frac{\lambda}{2}\mathbf{D}\right) \right\|_\infty \leq \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1} \right\|_\infty \left(\|\mathcal{W}_{\mathcal{I},r}\|_\infty + \frac{\lambda}{2} \|\mathbf{D}\|_\infty \right) \\
&\leq \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1} \right\|_\infty \left(1 + \frac{\pi^2}{12}\right) \lambda \leq 2\lambda \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \right\|_\infty^2
\end{aligned}$$

To deal with the first term in (46), note that $X_{\mathcal{I}} = W_{\mathcal{I}} \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{1/2}$, where $W_{\mathcal{I}} \in \mathbb{R}^{n \times K}$ is a standard Gaussian random matrix, i.e., $(W_{\mathcal{I}})_{ij} \sim N(0,1)$. Thus we can write it as

$$\left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \left[\left(\frac{1}{n} W_{\mathcal{I}}^T W_{\mathcal{I}}\right)^{-1} - \mathbf{I}_K \right] \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \left(\mathcal{W}_{\mathcal{I},r} + \frac{\lambda}{2}\mathbf{D}\right) \right\|_\infty \leq \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \right\|_\infty G,$$

where

$$G = \left\| \left[\left(\frac{1}{n} W_{\mathcal{I}}^T W_{\mathcal{I}}\right)^{-1} - \mathbf{I}_K \right] \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \left(\mathcal{W}_{\mathcal{I},r} + \frac{\lambda}{2}\mathbf{D}\right) \right\|_\infty.$$

By Lemma 5 in Wainwright (2009), we have, for some constant $c_7 > 0$.

$$\mathrm{P}\left[G \geq \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \left(\mathcal{W}_{\mathcal{I},r} + \frac{\lambda}{2}\mathbf{D}\right) \right\|_\infty \middle| \mathbf{X}_{\mathcal{I}}\right] \leq 4\exp\left(-c_7 \min\left\{K, \log J\right\}\right)$$

Note that conditioning on $\mathcal{A}$, $\left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \left(\mathcal{W}_{\mathcal{I},r} + \frac{\lambda}{2}\mathbf{D}\right) \right\|_\infty$ is upper bounded by $2\lambda \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \right\|_\infty$. Thus,

$$\mathrm{P}\left[G \geq 2\lambda \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \right\|_\infty^2 \middle| \mathcal{A}\right] \leq 4\exp\left(-c_7 \min\left\{K, \log J\right\}\right),$$

and

$$\begin{aligned}
&\mathrm{P}\left[\|F_2\|_\infty \geq 4\lambda \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \right\|_\infty^2\right] \leq \mathrm{P}\left[\|F_2\|_\infty \geq 4\lambda \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2} \right\|_\infty^2 \middle| \mathcal{A}\right] + \mathrm{P}\left[\mathcal{A}^c\right] \\
&\leq 4\exp\left(-c_7 \min\left\{K, \log J\right\}\right) + \frac{c_6}{p}. \qquad (47)
\end{aligned}$$

Turning to F_1, conditioned on $\mathbf{X}_{\mathcal{I}}$, by decomposition (39), we have that

$$\|F_1\|_\infty \leq \left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1} \Sigma_{\mathcal{I}r} \right\|_\infty |\delta_r| + \left\| \left(\mathbf{X}_{\mathcal{I}}^T \mathbf{X}_{\mathcal{I}}\right)^{-1} \mathbf{X}_{\mathcal{I}}^T E_r \delta_r \right\|_\infty.$$

By (45) and **A3**,

$$\mathrm{P}\left[\left\| \left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1} \Sigma_{\mathcal{I}r} \right\|_\infty |\delta_r| \geq \frac{5}{2\theta_r^{(r)}}\lambda\right] \leq c_4 \exp\left(-c_5 n\right) + \frac{c_6}{p}.$$

Consider each coordinate $j \in \mathcal{I}$ of the random term whose variance is bounded by

$$\mathrm{Var}\left[\mathbf{e}_j^T \left(\mathbf{X}_{\mathcal{I}}^T \mathbf{X}_{\mathcal{I}}\right)^{-1} \mathbf{X}_{\mathcal{I}}^T E_r \delta_r \middle| \mathbf{X}_{\mathcal{I}}\right] \leq \theta_r \left\| \left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T \mathbf{X}_{\mathcal{I}}\right)^{-1} \right\|_2 \frac{\delta_r^2}{n}.$$

By Lemma 18 and (45),

$$\mathrm{P}\left[\mathrm{Var}\left[\mathbf{e}_j^T \left(\mathbf{X}_{\mathcal{I}}^T \mathbf{X}_{\mathcal{I}}\right)^{-1} \mathbf{X}_{\mathcal{I}}^T E_r \delta_r \middle| \mathbf{X}_{\mathcal{I}}\right] \geq \frac{235}{4} \frac{\kappa^2}{\theta_r} \frac{\lambda^2}{n}\right] \leq 2\exp\left(-\frac{n}{2}\right) + c_4 \exp\left(-c_5 n\right) + \frac{c_6}{p}.$$

Thus by Lemma 13,

$$\mathrm{P}\left[\left\|\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\mathbf{X}_{\mathcal{I}}^T E_r\delta_r\right\|_\infty \geq \frac{5}{2\theta_r^{(r)}}\lambda\right] \leq 2\exp\left(-\frac{n}{18\theta_r\kappa^2}\right) + 2\exp\left(-\frac{n}{2}\right) + c_4\exp\left(-c_5 n\right) + \frac{c_6}{p},$$

and

$$\mathrm{P}\left[\|F_1\|_\infty \geq \frac{5}{\theta_r^{(r)}}\lambda\right] \leq 2\exp\left(-\frac{n}{18\theta_r\kappa^2}\right) + 2\exp\left(-\frac{n}{2}\right) + c_4\exp\left(-c_5 n\right) + \frac{c_6}{p}.$$

Combining with (45) and (47), we have

$$\mathrm{P}\left[\|\delta\|_\infty \geq 4\lambda\left\|\left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1/2}\right\|_\infty^2 + \frac{5}{\theta_r^{(r)}}\lambda\right] \leq c_8\exp\left(-c_9 n\right) + 2\frac{c_6}{p} + 4\exp\left(-c_7\min\left\{K, \log J\right\}\right),$$

for some constants $c_8, c_9 > 0$ that do not depend on n and J. ■

F.3 Proof of Property 3 in Theorem 1

Finally we establish a $\beta_{\min}$ condition, which, combined with the ℓ_∞ rate, gives the other direction of the support recovery, i.e., $K(\hat{\beta}) \geq K$.

By the triangle inequality

$$\left|\tilde{\beta}_j\right| \geq \left|\beta_j\right| - \left|\tilde{\beta}_j - \beta_j\right|.$$

So if we have

$$\max_{j\geq J+1}\left\{\left|\beta_j\right| - \left|\tilde{\beta}_j - \beta_j\right|\right\} > 0,$$

then $K(\tilde{\beta}) \geq K$.

Appendix G. Proof of Theorem 3

Proof The overall proof techniques are the same as the proof of Theorem 1. The first part of the theorem holds if $\max_{2\leq r\leq p}\max_{1\leq\ell\leq J_r}|\tilde{a}^{(r\ell)}| < 1$. Now for each $r = 2, \ldots, p$ we proceed with the same primal-dual witness procedure and end up with the same decomposition (40).

Assumption **A3** ensures that $\max_{2\leq r\leq p}\max_{1\leq\ell\leq J_r}|F^{(r\ell)}| \leq 1-\alpha$. Following the same line of proof to deal with random term $R^{(r\ell)}$, we have that $R^{(r\ell)}$ is zero-mean Gaussian with conditional variance bounded above by the scaling

$$\theta_r\bar{M}_n^{(r)}(\varepsilon) = \frac{3\kappa^2\pi^2\theta_r}{2}\frac{K_r^*}{n} + \frac{\theta_r}{\theta_r^{(r)}}\frac{1}{(n-K_r^*)(1-\varepsilon)} + \frac{16\theta_r}{n\lambda^2}$$
$$\leq \frac{3\kappa^2\pi^2\theta_r}{2}\left(\frac{K}{n} + \frac{\kappa^2}{n\theta_r^{(r)}(1-\varepsilon)^2} + \frac{16}{n\lambda^2}\right),$$

for $\varepsilon \in \left(0, \frac{1}{2}\right)$ with high probability, where we use the fact that $K = o(n)$ implies that $\frac{K}{n} \leq \varepsilon$ for n large. And

$$\mathrm{P}\left[\left|R^{(r\ell)}\right| \geq \alpha\right] \leq 2\exp\left(-\frac{\alpha^2}{2\theta_r\bar{M}_n^{(r)}(\varepsilon)}\right) + 7\exp\left(-c_3 n\right).$$

Thus,

$$\mathrm{P}\left[\max_{2\leq r\leq p}\max_{1\leq\ell\leq J_r}\left|R^{(r\ell)}\right| \geq \alpha\right] \leq 2\sum_{r=2}^p J_r\exp\left(-\frac{\alpha^2}{2\theta_r\bar{M}_n^{(r)}(\varepsilon)}\right) + 7\sum_{r=2}^p J_r\exp\left(-c_3 n\right)$$
$$\leq p^2\exp\left(-\frac{\alpha^2}{3\kappa^2\pi^2\theta\frac{K}{n} + \frac{8\theta\kappa^2}{n} + \frac{32\theta}{n\lambda^2}}\right) + \frac{7}{2}p^2\exp\left(-c_3 n\right).$$

For the exponential term to decay faster than p^2, we need

$$\frac{n}{\log p} > \max\left\{\frac{2}{\alpha^2}\left(3\kappa^2\pi^2\theta K + 8\kappa^2\theta + \frac{32\theta}{\lambda^2}\right), \frac{2}{c_3}\right\}.$$

■

Appendix H. Proof of Theorem 4

Lemma 17 *Using the notation and conditions in Theorem 4, the following deviation bounds hold with high probability:*

$$\left\|\hat{L} - L\right\|_\infty \leq \zeta_\Gamma(K+1)\sqrt{\frac{\log p}{n}},$$
$$\left\|\hat{L} - L\right\|_1 \leq \zeta_\Gamma(K+1)\sqrt{\frac{\log p}{n}},$$
$$\left\|\hat{L} - L\right\|_2 \leq \zeta_\Gamma(K+1)\sqrt{\frac{\log p}{n}},$$
$$\left\|\hat{L} - L\right\|_F \leq \zeta_\Gamma\sqrt{\frac{(s+p)\log p}{n}}.$$

Proof By Theorem 3, with high probability, the support of $\hat{L}$ is contained in the true support and

$$\left\|\hat{L} - L\right\|_\infty \leq \zeta_\Gamma\sqrt{\frac{\log p}{n}}.$$

Note that

$$\left\|\hat{L} - L\right\|_\infty = \max_{2\leq r\leq p}\sum_{c=1}^r\left|\hat{L}_{rc} - L_{rc}\right| \leq \max_{2\leq r\leq p}(K_r+1)\left\|\hat{L} - L\right\|_\infty \leq (K+1)\left\|\hat{L} - L\right\|_\infty.$$

Denote $D = \max_{1\le c\le p-1} D_c$ where $D_c = |\{r = c, \ldots, p : L_{rc} \neq 0\}|$. Observing that $D \le K$, we have

$$\begin{aligned}\left|\left|\left|\hat{L} - L\right|\right|\right|_1 &= \max_{1\le c\le p-1} \sum_{r=1}^{c} \left|\hat{L}_{rc} - L_{rc}\right| \le \max_{1\le c\le p-1} (D_c + 1) \left\|\hat{L} - L\right\|_\infty \\ &\le (D+1)\left\|\hat{L} - L\right\|_\infty \le (K+1)\left\|\hat{L} - L\right\|_\infty .\end{aligned}$$

By Hölder's inequality

$$\left|\left|\left|\hat{L} - L\right|\right|\right|_2 \le \sqrt{\left|\left|\left|\hat{L} - L\right|\right|\right|_1 \left|\left|\left|\hat{L} - L\right|\right|\right|_\infty}.$$

Finally for Frobenius norm,

$$\left\|\hat{L} - L\right\|_F^2 = \sum_{r=2}^{p}\sum_{c=J_r+1}^{r} \left(\hat{L}_{rc} - L_{rc}\right)^2 \le \sum_{r=2}^{p}\sum_{c=J_r+1}^{r} \left\|\hat{L} - L\right\|_\infty^2 \le \zeta_\Gamma^2 \left(\sum_r K_r + p\right)\frac{\log p}{n}.$$

■

Proof [of Theorem 4] First note that

$$\begin{aligned}\hat{L}^T\hat{L} - L^TL &= \left(\hat{L} - L\right)^T\left(\hat{L} - L\right) + \hat{L}^TL + L^T\hat{L} - 2L^TL \\ &= \left(\hat{L} - L\right)^T\left(\hat{L} - L\right) + \left(\hat{L} - L\right)^TL + L^T\left(\hat{L} - L\right).\end{aligned}$$

Thus,

$$\begin{aligned}\left\|\hat{L}^T\hat{L} - L^TL\right\|_\infty &\le \left|\left|\left|\hat{L} - L\right|\right|\right|_\infty \left\|\hat{L} - L\right\|_\infty + 2|||L|||_\infty \left\|\hat{L} - L\right\|_\infty, \\ \left|\left|\left|\hat{L}^T\hat{L} - L^TL\right|\right|\right|_1 &= \left|\left|\left|\hat{L}^T\hat{L} - L^TL\right|\right|\right|_\infty \le 2|||L|||_\infty \left|\left|\left|\hat{L} - L\right|\right|\right|_\infty + \left\|\hat{L} - L\right\|_\infty^2.\end{aligned}$$

By Hölder's inequality

$$\left|\left|\left|\hat{L}^T\hat{L} - L^TL\right|\right|\right|_2 \le \sqrt{\left|\left|\left|\hat{L}^T\hat{L} - L^TL\right|\right|\right|_1 \left|\left|\left|\hat{L}^T\hat{L} - L^TL\right|\right|\right|_\infty}.$$

Finally, for Frobenius norm, observe that

$$\begin{aligned}\left\|L^T\left(\hat{L} - L\right)\right\|_F &= \left\|\mathrm{vec}\left(L^T\left(\hat{L} - L\right)\right)\right\|_2 = \left\|\left(I_p \otimes L^T\right)\mathrm{vec}\left(\hat{L} - L\right)\right\|_2 \\ &\le \left|\left|\left|I_p \otimes L^T\right|\right|\right|_2 \left\|\hat{L} - L\right\|_F = |||L|||_2 \left\|\hat{L} - L\right\|_F.\end{aligned}$$

Applying the same strategy to $\left\|\left(\hat{L} - L\right)\left(\hat{L} - L\right)\right\|_F$, we have

$$\left\|\hat{L}^T\hat{L} - L^TL\right\|_F \le \left(\left|\left|\left|\hat{L} - L\right|\right|\right|_2 + 2|||L|||_2\right)\left\|\hat{L} - L\right\|_F,$$

then the results follow from Corollary 17. ■

Appendix I. Proof of Theorem 6

Proof We adapt the proof technique of Rothman et al. (2008). Let

$$\begin{aligned}G(\Delta) = &-2\log\det(L+\Delta) + \mathrm{tr}\left(S(L+\Delta)^T(L+\Delta)\right) + \lambda\|(\Delta+L)\|_{2,1}^* \\ &+ 2\log\det L - \mathrm{tr}\left(SL^TL\right) - \lambda\|L\|_{2,1}^*,\end{aligned} \tag{48}$$

where L is the inverse of the Cholesky factor of the true covariance matrix, and the penalty is defined above as

$$\|L\|_{2,1}^* = \sum_{r=2}^{p}\sum_{\ell=1}^{r-1}\sqrt{\sum_{m=1}^{\ell} w_{\ell m}^2 L_{rm}^2}.$$

Since the estimator $\hat{L}$ is defined as

$$\hat{L} = \underset{L_{jk}=0:j<k}{\arg\min}\left\{-2\log\det L + \mathrm{tr}\left(SL^TL\right) + \lambda\|L\|_{2,1}^*\right\},$$

it follows that $G(\Delta)$ is minimized at $\hat{\Delta} = \hat{L} - L$. Consider the value of $G(\Delta)$ on the set defined as

$$\Theta_n(M) = \left\{\Delta :\ \Delta_{jk} = 0 \text{ for all } k > j,\ \ (\Delta+L)_{jj} > 0 \text{ for all } j\ , \|\Delta\|_F = Mr_n\right\},$$

where $M > 0$ and

$$r_n = \sqrt{\frac{(\sum_{r=2}^{p} K_r + p)\log p}{n}}.$$

The assumed scaling implies that $r_n \to 0$. We aim at showing that $\inf\{G(\Delta) :\ \Delta \in \Theta_n(M)\} > 0$. If it holds, then the convexity of $G(\Delta)$ and the fact that $G(\hat{\Delta}) \le G(\mathbf{0}) = 0$ implies

$$\|\hat{\Delta}\|_F = \|\hat{L} - L\|_F \le Mr_n.$$

We start with analyzing the logarithm terms in (48). First let $f(t) = \log\det(L + t\Delta)$. Using a Taylor expansion of $f(t)$ at $t = 0$ with $f'(t) = \mathrm{tr}[(L+t\Delta)^{-1}\Delta]$ and $f''(t) = -\mathrm{vec}\,\Delta^T(L+t\Delta)^{-1} \otimes (L+t\Delta)^{-1}\,\mathrm{vec}\,\Delta$, we have

$$\begin{aligned}&\log\det(L+\Delta) - \log\det(L) \\ =&\,\mathrm{tr}(L^{-1}\Delta) - (\mathrm{vec}\,\Delta)^T\left[\int_0^1 (1-\nu)(L+\nu\Delta)^{-1}\otimes(L+\nu\Delta)^{-1}d\nu\right](\mathrm{vec}\,\Delta).\end{aligned}$$

The trace term in (48) can be written as

$$\begin{aligned}\mathrm{tr}\left(S(L+\Delta)^T(L+\Delta)\right) - \mathrm{tr}\left(SL^TL\right) &= \mathrm{tr}\left(SL^T\Delta + S\Delta^TL + S\Delta^T\Delta\right) \\ &= 2\,\mathrm{tr}\left(SL^T\Delta\right) + \mathrm{tr}\left(S\Delta^T\Delta\right) \\ &\ge 2\,\mathrm{tr}\left(SL^T\Delta\right),\end{aligned}$$

where the last inequality comes from the fact that the sample covariance matrix S is positive semidefinite. Combining these with (48) gives

$$\begin{aligned} G(\Delta) \geq & 2(\operatorname{vec}\Delta)^T \left[\int_0^1 (1-\nu)(L+\nu\Delta)^{-1} \otimes (L+\nu\Delta)^{-1} d\nu\right] (\operatorname{vec}\Delta) \\ & + 2\operatorname{tr}[(SL^T - L^{-1})\Delta] + \lambda\left(\|L+\Delta\|_{2,1}^* - \|L\|_{2,1}^*\right) \\ \equiv & (a) + (b) + (c). \end{aligned} \tag{49}$$

The integral term (a) above has a positive lower bound. Recalling that $\sigma_{\min}(M) = \min_{\|x\|=1} x^T M x$ is a concave function of M (the minimum of linear functions of M is concave), we have

$$\begin{aligned} (a) &= 2\|\operatorname{vec}\Delta\|^2 \frac{\operatorname{vec}\Delta^T}{\|\operatorname{vec}\Delta\|}\left[\int_0^1 (1-\nu)(L+\nu\Delta)^{-1} \otimes (L+\nu\Delta)^{-1} d\nu\right] \frac{\operatorname{vec}\Delta}{\|\operatorname{vec}\Delta\|} \\ &\geq 2\|\Delta\|_F^2 \sigma_{\min}\left[\int_0^1 (1-\nu)(L+\nu\Delta)^{-1} \otimes (L+\nu\Delta)^{-1} d\nu\right] \\ &\geq 2\|\Delta\|_F^2 \left[\int_0^1 (1-\nu)\sigma_{\min}\left((L+\nu\Delta)^{-1} \otimes (L+\nu\Delta)^{-1}\right) d\nu\right] \\ &\geq 2\|\Delta\|_F^2 \int_0^1 (1-\nu)\sigma_{\min}^2(L+\nu\Delta)^{-1} d\nu \\ &\geq \|\Delta\|_F^2 \min_{0\leq\nu\leq 1} \sigma_{\min}^2(L+\nu\Delta)^{-1} \\ &\geq \|\Delta\|_F^2 \min\left\{\sigma_{\min}^2(L+\tilde{\Delta})^{-1} : \ \|\tilde{\Delta}\|_F \leq M r_n\right\}. \end{aligned} \tag{50}$$

The second inequality uses Jenson's inequality of the concave function $\sigma_{\min}(\cdot)$, and the third inequality uses the fact that $\sigma_{\min}(A \otimes A) = \sigma_{\min}(A)^2$ for any positive (semi)definite matrix A. Using triangle inequality on the matrix operator norm, we have

$$\sigma_{\min}^2(L+\tilde{\Delta})^{-1} = \sigma_{\max}^{-2}(L+\tilde{\Delta}) \geq \left(\|L\|_2 + \left\|\left\|\tilde{\Delta}\right\|\right\|_2\right)^{-2} \geq \frac{1}{2\|\|L\|\|_2^2} \geq \frac{\kappa^2}{2},$$

where the second inequality holds with high probability since $\left\|\left\|\tilde{\Delta}\right\|\right\|_2 \leq \|\tilde{\Delta}\|_F \leq M r_n \leq \|\|L\|\|_2$ as $r_n \to 0$ and the last inequality follows from Assumption **A4**. This gives the lower bound for the first term in (49):

$$(a) \geq \frac{1}{2}\kappa^2 \|\Delta\|_F^2 = \frac{1}{2}\kappa^2 M^2 r_n^2. \tag{51}$$

To deal with (b), we start by recalling some notation. We let $\mathcal{S} = \{(r,j) : L_{rj} \neq 0\}$ denote the support of L, and $s = \sum_{r=2}^p K_r$ be the number of non-zero off-diagonal elements. We also define

$$\|L\|_{2,1} = \sum_{r=2}^p \sum_{\ell=1}^{r-1} w_{\ell\ell}|L_{r\ell}| = \sum_{r=2}^p \sum_{\ell=1}^{r-1} |L_{r\ell}|,$$

where the last equality holds since $w_{\ell\ell} = 1$ by (7). Then, by the Cauchy-Schwarz inequality,

$$\begin{aligned} \left|\operatorname{tr}[(SL^T - L^{-1})\Delta]\right| &= \left|\sum_{r=1}^p \sum_{j=1}^r \left(SL^T - L^{-1}\right)_{rj} \Delta_{rj}\right| \\ &\leq \left|\sum_{r=1}^p \sum_{j\in\mathcal{I}_r} (SL^T - L^{-1})_{rj}\Delta_{rj}\right| + \left|\sum_{r=1}^p \sum_{j\notin\mathcal{I}_r} (SL^T - L^{-1})_{rj}\Delta_{rj}\right| \\ &\leq \sqrt{s+p}\left\|SL^T - L^{-1}\right\|_\infty \|\Delta_{\mathcal{S}}\|_F + \left\|SL^T - L^{-1}\right\|_\infty \|\Delta_{\mathcal{S}^c}\|_{2,1} \\ &\leq C_1\sqrt{s+p}\sqrt{\frac{\log p}{n}}\|\Delta_{\mathcal{S}}\|_F + C_1\sqrt{\frac{\log p}{n}}\|\Delta_{\mathcal{S}^c}\|_{2,1}, \end{aligned} \tag{52}$$

where the last inequality comes from Lemma 15 with probability tending to 1. To bound the penalty terms, we note that

$$\begin{aligned} &\|L+\Delta\|_{2,1}^* - \|L\|_{2,1}^* \\ =& \sum_{r=2}^p \sum_{\ell=1}^{r-1} \sqrt{\sum_{m=1}^{\ell} w_{\ell m}^2 (L_{rm}+\Delta_{rm})^2} - \|L_{\mathcal{S}}\|_{2,1}^* \\ =& \sum_{r=2}^p \sum_{\ell=1}^{r-1} \sqrt{\sum_{m:(r,m)\in\mathcal{S}} w_{\ell m}^2 (L_{rm}+\Delta_{rm})^2 + \sum_{m:(r,m)\notin\mathcal{S}} w_{\ell m}^2 (L_{rm}+\Delta_{rm})^2} - \|L_{\mathcal{S}}\|_{2,1}^* \\ \geq& \sum_{r=2}^p \sum_{\ell=1}^{r-1} \sqrt{\sum_{m:(r,m)\in\mathcal{S}} w_{\ell m}^2 (L_{rm}+\Delta_{rm})^2} + \sum_{r=2}^p \sum_{\ell:(r,\ell)\notin\mathcal{S}} |L_{r\ell}+\Delta_{r\ell}| - \|L_{\mathcal{S}}\|_{2,1}^* \\ =& \|L_{\mathcal{S}}+\Delta_{\mathcal{S}}\|_{2,1}^* + \|L_{\mathcal{S}^c}+\Delta_{\mathcal{S}^c}\|_{2,1} - \|L_{\mathcal{S}}\|_{2,1}^* \\ =& \|L_{\mathcal{S}}+\Delta_{\mathcal{S}}\|_{2,1}^* + \|\Delta_{\mathcal{S}^c}\|_{2,1} - \|L_{\mathcal{S}}\|_{2,1}^* \\ \geq& \|\Delta_{\mathcal{S}^c}\|_{2,1} - \|\Delta_{\mathcal{S}}\|_{2,1}^*, \end{aligned}$$

where the last inequality comes from triangle inequality. To give an upper bound on $\|L_{\mathcal{S}}\|_{2,1}^*$, we observe that $2\lambda b \leq a\lambda^2 + b^2/a$ holds for any $a > 0$, and obtain

$$\begin{aligned} 2\lambda\|\Delta_{\mathcal{S}}\|_{2,1}^* &= \sum_{r=2}^p 2\lambda \sum_{\ell=J_r+1}^{r-1} \sqrt{\sum_{m=J_r+1}^{\ell} w_{\ell m}^2 \Delta_{rm}^2} \\ &\leq \left(\sum_{r=2}^p K_r\right)\lambda^2 a + \sum_{r=2}^p \sum_{\ell=J_r+1}^{r-1} \sum_{m=J_r+1}^{\ell} w_{\ell m}^2 \Delta_{rm}^2/a \\ &= \left(\sum_{r=2}^p K_r\right)\lambda^2 a + \sum_{r=2}^p \sum_{m=J_r+1}^{r-1} \left(\sum_{\ell=m}^{r-1} w_{\ell m}^2\right)\Delta_{rm}^2/a. \end{aligned}$$

Now let

$$a = \frac{4}{\kappa^2}\max_r \max_{J_r+1\le m\le r-1}\sum_{\ell=m}^{r-1} w_{\ell m}^2$$
$$= \frac{4}{\kappa^2}\max_r \max_{J_r+1\le m\le r-1}\sum_{\ell=m}^{r-1}\frac{1}{(\ell-m+1)^4} \le \sum_{k=1}^{\infty}\frac{4}{k^4\kappa^2} \le \frac{C_2}{\kappa^2},$$

for some constant $C_2 > 0$, it follows that

$$\lambda\left\|\Delta_{\mathcal{S}}\right\|_{2,1}^* \le \frac{C_2}{\kappa^2}s\lambda^2 + \left\|\Delta_{\mathcal{S}}\right\|_F^2\frac{\kappa^2}{4} \le \frac{C_2}{\kappa^2}s\lambda^2 + \left\|\Delta\right\|_F^2\frac{\kappa^2}{4}.$$

Therefore,

$$\lambda\left(\left\|L+\Delta\right\|_{2,1}^* - \left\|L\right\|_{2,1}^*\right) \ge \lambda\left\|\Delta_{\mathcal{S}^c}\right\|_{2,1} - \frac{C_2}{\kappa^2}s\lambda^2 - \frac{\kappa^2}{4}\left\|\Delta\right\|_F^2. \tag{53}$$

Finally, combining (51), (52), and (53), we have

$$G(\Delta) \ge \frac{\kappa^2}{4}\left\|\Delta\right\|_F^2 - C_1\sqrt{\frac{(s+p)\log p}{n}}\left\|\Delta\right\|_F + \left(\lambda - C_1\sqrt{\frac{\log p}{n}}\right)\left\|\Delta_{\mathcal{S}^c}\right\|_{2,1} - \frac{C_2}{\kappa^2}s\lambda^2.$$

For any $\varepsilon < 1$, choose

$$\lambda = \frac{C_1}{\varepsilon}\sqrt{\frac{\log p}{n}}.$$

Since $\left\|\Delta\right\|_F = Mr_n$, we have

$$G(\Delta) \ge \frac{\kappa^2}{4}M^2r_n^2 - C_1Mr_n^2 + C_1\sqrt{\frac{\log p}{n}}\left(\frac{1}{\varepsilon}-1\right)\left\|\Delta_{\mathcal{S}^c}\right\|_{2,1} - \frac{C_2C_1^2}{\kappa^2\varepsilon^2}\frac{s\log p}{n}$$
$$\ge \left(\frac{\kappa^2}{4}M^2 - C_1M - \frac{C_2C_1^2}{\kappa^2\varepsilon^2}\right)r_n^2 > 0,$$

for M sufficiently large. ■

Appendix J. Proof of Lemma 10

Proof Denote

$$\mathcal{L}\left(\tau, z, \beta; \nu, \phi, a^{(\ell)}\right)$$
$$= -2\log\tau + \frac{1}{n}\left\|z\right\|_2^2 + \nu\left(\tau - \beta_r\right) + \frac{1}{n}\left\langle\phi, z - \mathbf{X}_{1:r}\beta\right\rangle + \lambda\sum_{\ell=1}^{r-1}\left\langle W^{(\ell)} * a^{(\ell)}, \beta\right\rangle.$$

Then the primal (8) can be written equivalently as

$$\min_{\tau,z,\beta}\left\{\max_{\nu,\phi,a^{(\ell)}}\left\{\mathcal{L}\left(\tau, z, \beta; \nu, \phi, a^{(\ell)}\right) : \left\|\left(a^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 \le 1, \left(a^{(\ell)}\right)_{g_{r,\ell}^c} = 0\right\}\right\}.$$

The dual function can then be written as

$$g\left(\nu, \phi, a^{(\ell)}\right) = \inf_{\tau,z,\beta}\mathcal{L}\left(\tau, z, \beta; \nu, \phi, a^{(\ell)}\right)$$
$$= \inf_{\tau}\left\{-2\log\tau + \nu\tau\right\} + \inf_{z}\left\{\frac{1}{n}\left\|z\right\|_2^2 + \frac{1}{n}\left\langle\phi, z\right\rangle\right\}$$
$$+ \inf_{\beta}\left\{-\nu\beta_r - \frac{1}{n}\left\langle\mathbf{X}_{1:r}^T\phi, \beta\right\rangle + \lambda\sum_{\ell=1}^{r-1}\left\langle W^{(\ell)} * a^{(\ell)}, \beta\right\rangle\right\}$$
$$= 2\log\nu - 2\log 2 + 2 - \mathbb{1}_\infty\left\{\nu > 0\right\} - \frac{1}{4n}\left\|\phi\right\|_2^2$$
$$- \mathbb{1}_\infty\left\{-\nu\mathbf{e}_r - \frac{1}{n}\mathbf{X}_{1:r}^T\phi + \lambda\sum_{\ell=1}^{r-1}W^{(\ell)} * a^{(\ell)} = 0\right\},$$

where $\mathbf{e}_r \in \mathbb{R}^r$ is such that $(\mathbf{e}_r)_r = 1$ and $(\mathbf{e}_r)_j = 0$ for all $j \ne r$. Thus the dual problem (up to a constant) is

$$\max_{\nu,\phi,a^{(\ell)}} g\left(\nu, \phi, a^{(\ell)}\right)$$
$$= \min_{\nu,\phi,a^{(\ell)}}\left\{-2\log\nu + \frac{1}{4n}\left\|\phi\right\|_2^2 \quad \text{s.t.} \quad \nu > 0, \quad \left\|\left(a^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 \le 1, \left(a^{(\ell)}\right)_{g_{r,\ell}^c} = 0,\right.$$
$$\left.\nu\mathbf{e}_r + \frac{1}{n}\mathbf{X}_{1:r}^T\phi = \lambda\sum_{\ell=1}^{r-1}W^{(\ell)} * a^{(\ell)}\right\}.$$

The primal-dual relation is

$$\hat{\beta}_r = \hat{\tau} = \frac{2}{\hat{\nu}} \qquad \hat{\phi} = -2\hat{z} = -2\mathbf{X}_{1:r}\hat{\beta}.$$

This implies that at optimal points

$$-\frac{2}{\hat{\beta}_r}\mathbf{e}_r + 2S_{1:r,1:r}\hat{\beta} + \lambda\sum_{\ell=1}^{r-1}W^{(\ell)} * \hat{a}^{(\ell)} = 0,$$

with $\left\|(\hat{a}^{(\ell)})_{g_{r,\ell}}\right\|_2 \le 1, (\hat{a}^{(\ell)})_{g_{r,\ell}^c} = 0$.

If we denote the objective function as

$$f\left(\beta\right) = -2\log\beta_r + \left\langle S_{1:r,1:r}, \beta\beta^T\right\rangle + \lambda P(\beta),$$

then from the equality $f(\hat{\beta}) = \mathcal{L}\left(\hat{\tau}, \hat{z}, \hat{\beta}; \hat{\nu}, \hat{\phi}, \hat{a}^{(\ell)}\right)$ together with the primal-dual relation, we have

$$P(\hat{\beta}) = \sum_{\ell=1}^{r-1}\left\langle W^{(\ell)} * \hat{a}^{(\ell)}, \hat{\beta}\right\rangle = \sum_{\ell=1}^{r-1}\left\langle W^{(\ell)} * \hat{\beta}, \hat{a}^{(\ell)}\right\rangle.$$

Suppose there exists some ℓ with $\hat{\beta}_{g_{r,\ell}} \neq 0$ but $\left(\hat{a}^{(\ell)}\right)_{g_{r,\ell}} \neq \frac{\left(W^{(\ell)} * \hat{\beta}\right)_{g_{r,\ell}}}{\left\|\left(W^{(\ell)} * \hat{\beta}\right)_{g_{r,\ell}}\right\|_2}$, then $\left\langle W^{(\ell)} * \hat{\beta}, \hat{a}^{(\ell)} \right\rangle < \left\| \left(W^{(\ell)} * \hat{\beta}\right)_{g_{r,\ell}} \right\|_2$ while for other ℓ' by Cauchy-Schwarz inequality we have $\left\langle W^{(\ell')} * \hat{\beta}, \hat{a}^{(\ell')} \right\rangle \leq \left\| \left(W^{(\ell')} * \hat{\beta}\right)_{g_{r,\ell'}} \right\|_2$. Therefore, summing over all $\ell = 1, \ldots, r-1$ would give

$$P(\hat{\beta}) = \sum_{\ell=1}^{r-1} \left\| \left(W^{(\ell)} * \hat{\beta}\right)_{g_{r,\ell}} \right\|_2 > \sum_{r=2}^{p} \sum_{\ell=1}^{r-1} \left\langle W^{(\ell)} * \hat{\beta}, \hat{a}^{(\ell)} \right\rangle,$$

which leads to a contradiction. Thus $\left(\hat{a}^{(\ell)}\right)_{g_{r,\ell}} = \frac{\left(W^{(\ell)} * \hat{\beta}\right)_{g_{r,\ell}}}{\left\|\left(W^{(\ell)} * \hat{\beta}\right)_{g_{r,\ell}}\right\|_2}$ for $\hat{\beta}_{g_{r,\ell}} \neq 0$ and $\left\|\hat{a}^{(\ell)}_{g_{r,\ell}}\right\|_2 \leq 1$ for $\hat{\beta}_{g_{r,\ell}} = 0$. ■

Appendix K. Proof of Lemma 11

Proof In this proof, we continue to use the notation in Appendix J. Observe that $\mathcal{L}\left(\tau, z, \beta; \nu, \phi, a^{(\ell)}\right)$ is jointly convex in τ, z and β, and it is strictly convex in τ and z. Thus, the minimizers $\hat{z}$ and $\hat{\tau}$ are unique.

To see this in a more general setting, without loss of generality, suppose $f(x, y)$ is convex in y and is strictly convex in x. Then for $x_1 \neq x_2$ and $\theta \in (0, 1)$ we have

$$f\left(\theta x_1 + (1-\theta) x_2, y\right) < \theta f\left(x_1, y\right) + (1-\theta) f\left(x_2, y\right)$$

Now suppose $(\hat{x}_1, \hat{y})$ and $(\hat{x}_2, \hat{y}_2)$ are both minima of f, then taking $\theta = 1/2$ we have $f\left(\frac{\hat{x}_1 + \hat{x}_2}{2}, \hat{y}\right) < f\left(\hat{x}_1, \hat{y}\right) = f\left(\hat{x}_2, \hat{y}\right)$, which leads to a contradiction.

By the primal-dual relation, we know that if $\hat{\beta}$ and $\tilde{\beta}$ are two solutions to (8), then $\hat{\beta}_r = \tilde{\beta}_r$ and $\mathbf{X}_{1:r}\hat{\beta} = \mathbf{X}_{1:r}\tilde{\beta}$. So from the equality $f(\hat{\beta}) = f(\tilde{\beta})$ we know that $P(\tilde{\beta}) = P(\hat{\beta})$. Also by

$$f\left(\hat{\beta}\right) = \mathcal{L}\left(\hat{\tau}, \hat{z}, \hat{\beta}; \hat{\nu}, \hat{\phi}, \hat{a}^{(\ell)}\right) \leq \mathcal{L}\left(\hat{\tau}, \hat{z}, \tilde{\beta}; \hat{\nu}, \hat{\phi}, \hat{a}^{(\ell)}\right) \leq \mathcal{L}\left(\tilde{\tau}, \tilde{z}, \tilde{\beta}; \tilde{\nu}, \tilde{\phi}, \tilde{a}^{(\ell)}\right) = f\left(\tilde{\beta}\right),$$

we have

$$\mathcal{L}\left(\hat{\tau}, \hat{z}, \hat{\beta}; \hat{\nu}, \hat{\phi}, \hat{a}^{(\ell)}\right) = \mathcal{L}\left(\hat{\tau}, \hat{z}, \tilde{\beta}; \hat{\nu}, \hat{\phi}, \hat{a}^{(\ell)}\right),$$

and thus

$$\sum_{\ell=1}^{r-1} \left\langle W^{(\ell)} * \hat{a}^{(\ell)}, \tilde{\beta} \right\rangle = \sum_{\ell=1}^{r-1} \left\langle W^{(\ell)} * \hat{a}^{(\ell)}, \hat{\beta} \right\rangle = P(\hat{\beta}) = P(\tilde{\beta}) = \sum_{\ell=1}^{r-1} \left\| \left(W^{(\ell)} * \tilde{\beta}\right)_{g_{r,\ell}} \right\|_2.$$

Now for any $\ell \leq r-1$ suppose $\left\|\left(\hat{a}^{(\ell)}\right)_{g_{r,\ell}}\right\|_2 < 1$, then for the equality above to hold, we must have $\tilde{\beta}_{g_{r,\ell}} = 0$. Therefore, by Lemma 10, $\hat{\beta}_{g_{r,\ell}} = 0 \implies \tilde{\beta}_{g_{r,\ell}} = 0$, so any other solutions to (8) cannot be less sparse than $\hat{\beta}$. ■

Appendix L. Proof of Lemma 12

Proof By Lemma 11, any other solution β to (8) must have $\beta_{g_{J(\hat{\beta})}} = 0$. Recall that $J(\hat{\beta}) = r - 1 - K(\hat{\beta})$. The original problem (8) can thus be written equivalently as

$$\min_{\gamma \in \mathbb{R}^{K(\hat{\beta})+1}} -2\log\gamma_{K(\hat{\beta})+1} + \frac{1}{n}\left\|\mathbf{X}_{\hat{\mathcal{S}}}\gamma\right\|_2^2 + \lambda \sum_{\ell=1}^{K(\hat{\beta})} \left\| \left(\hat{W}^{(\ell)} * \gamma\right)_{g_{r,\ell}} \right\|_2,$$

where $\hat{W}^{(\ell)} = \left(W^{(\ell+\hat{J})}\right)_{\hat{S}}$.

Note that the penalty term is a convex function of γ. The Hessian matrix of the first term is a diagonal matrix of dimension $|\hat{\mathcal{S}}| = K(\hat{\beta}) + 1$ with non-negative entries in the diagonal. The Hessian matrix of the second term is $2S_{\hat{\mathcal{S}}\hat{\mathcal{S}}}$. Then by Assumption **A1**, the uniqueness follows from strict convexity. ■

Appendix M. Proof of Lemma 14

Proof Recall that

$$M_n = \frac{1}{n}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}}^T \left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1} \left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}} + \frac{4}{n^2\lambda^2}\tilde{\beta}_r^2 \left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2.$$

We cite Lemma 9 (specifically in the form (60)) in Wainwright (2009) here for completeness.

Lemma 18 (Wainwright 2009) *For $k \leq n$, let $\mathbf{X}_{\mathcal{I}} \in \mathbb{R}^{n\times k}$ have i.i.d. rows from a multivariate Gaussian distribution with mean $\mathbf{0}$ and covariance matrix Σ. If Σ has minimum eigenvalue $\kappa > 0$, then*

$$\mathrm{P}\left[\left\|\left(\frac{1}{n}X_{\mathcal{I}}^T X_{\mathcal{I}}\right)^{-1}\right\|_2 \geq \frac{9}{\kappa}\right] \leq 2\exp\left(-\frac{n}{2}\right).$$

By the lemma above, Assumption **A4**, and (41)

$$\begin{aligned}\frac{1}{n}\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}}^T \left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1} \left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}} &\leq \frac{9\kappa^2}{n}\left\|\left(\sum_{\ell=1}^{r-1} W^{(\ell)} * \tilde{a}^{(\ell)}\right)_{\mathcal{I}}\right\|^2 \\ &\leq \frac{3\pi^2\kappa^2}{2}\frac{K}{n},\end{aligned}$$

with probability greater than $1 - 2\exp\left(-\frac{n}{2}\right)$.

Next we deal with the second term in M_n. Recall from (37) that

$$\begin{aligned}\frac{4}{n^2\lambda^2}\tilde{\beta}_r^2\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2 &= \frac{4}{n^2}\left(\frac{\frac{1}{2}\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}}+\sqrt{\frac{1}{4}\left(\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}}\right)^2+\frac{4}{\lambda^2 n}\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2}}{\frac{2}{n}\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2}\right)^2\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2\\ &\leq \frac{4}{n^2}\frac{\frac{1}{4}\left(\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}}\right)^2+\frac{4}{\lambda^2 n}\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2}{\frac{1}{n^2}\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^4}\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2\\ &= \frac{\left(\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}}\right)^2}{\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2}+\frac{16}{\lambda^2 n}.\end{aligned}$$

The next lemma gives us a handle on the numerator of the first term.

Lemma 19 *Using the general weight* (7)*, we have*

$$\mathrm{P}\left[\left|\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}}\right|\geq 1\right]\leq 2\exp\left(-\frac{n\alpha^2}{3\theta\kappa^2\pi^2K}\right)+2\exp\left(-\frac{n}{2}\right).$$

Proof Conditioned on $\mathbf{X}_{\mathcal{I}}$, from the decomposition (39) and the definition of $\mathbf{C}_{\mathcal{I}}$

$$\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}}=\Sigma_{r\mathcal{I}}\left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1}W^{(\ell)}*\tilde{a}^{(\ell)}\right)_{\mathcal{I}}+E_r^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1}W^{(\ell)}*\tilde{a}^{(\ell)}\right)_{\mathcal{I}}.$$

By the irrepresentable assumption (**A3**) and (41),

$$\Sigma_{r\mathcal{I}}\left(\Sigma_{\mathcal{I}\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1}W^{(\ell)}*\tilde{a}^{(\ell)}\right)_{\mathcal{I}}\leq 1-\alpha.$$

Note that $\mathrm{Var}\left(E_{ir}\right)=\theta_r^{(r)}$ for $i=1,\ldots,n$. Let $B^{(r)}=E_r^T\mathbf{X}_{\mathcal{I}}\left(\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1}W^{(\ell)}*\tilde{a}^{(\ell)}\right)_{\mathcal{I}}$. By Lemma 18, $B^{(r)}$ has mean zero and variance at most

$$\mathrm{Var}\left(B^{(r)}\middle|\mathbf{X}_{\mathcal{I}}\right)=\frac{\theta_r^{(r)}}{n}\left(\sum_{\ell=1}^{r-1}W^{(\ell)}*\tilde{a}^{(\ell)}\right)_{\mathcal{I}}^T\left(\frac{1}{n}\mathbf{X}_{\mathcal{I}}^T\mathbf{X}_{\mathcal{I}}\right)^{-1}\left(\sum_{\ell=1}^{r-1}W^{(\ell)}*\tilde{a}^{(\ell)}\right)_{\mathcal{I}}\leq\frac{3\theta_r^{(r)}\kappa^2\pi^2K}{2n},$$

with probability greater than $1-2\exp\left(\frac{n}{2}\right)$. By Lemma 13, we have that

$$\mathrm{P}\left[B^{(r)}\geq\alpha\right]\leq 2\exp\left(-\frac{n\alpha^2}{3\theta_r^{(r)}\kappa^2\pi^2K}\right)+2\exp\left(-\frac{n}{2}\right).$$

■

Since $\frac{\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2}{\theta_r^{(r)}}\sim\chi^2\left(n-K\right)$. To bound it, we cite a concentration inequality from Wainwright (2009) (specifically (54b)) as the following lemma:

Lemma 20 (Tail Bounds for χ^2-variates, Wainwright 2009) *For a centralized χ^2-variate X with d degrees of freedom, for all $\varepsilon\in(0,1/2)$, we have*

$$\mathrm{P}\left[X\leq d(1-\varepsilon)\right]\leq\exp\left(-\frac{1}{4}d\varepsilon^2\right).$$

From Lemma 20 it follows that

$$\mathrm{P}\left[\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2\leq\theta_r^{(r)}\left(n-K\right)\left(1-\varepsilon\right)\right]\leq\exp\left(-\frac{1}{4}\left(n-K\right)\varepsilon^2\right),$$

which together with Lemma 19 implies that

$$\begin{aligned}&\mathrm{P}\left[\frac{\left(\mathbf{X}_r^T\mathbf{C}_{\mathcal{I}}\right)^2}{\left\|\mathbf{O}_{\mathcal{I}}E_r\right\|_2^2}\geq\frac{1}{\theta_r^{(r)}\left(n-K\right)\left(1-\varepsilon\right)}\right]\\ \leq&2\exp\left(-\frac{n\alpha^2}{3\theta_r^{(r)}\kappa^2\pi^2K}\right)+2\exp\left(-\frac{n}{2}\right)+\exp\left(-\frac{1}{4}\left(n-K\right)\varepsilon^2\right).\end{aligned}$$

The result follows from a union bound. ■

Appendix N. Proof of Lemma 15

Proof The proof strategy is based on the proof of Lemma 2 in Bien et al. (2016).

For the design matrix $\mathbf{X}_{n\times p}$ with independent rows, denote $X_i=\left(\mathbf{X}_{i\cdot}\right)^T\in\mathbb{R}^p$. Then X_i are i.i.d with mean 0 and true covariance matrix $\Sigma=\left(L^TL\right)^{-1}$ for $i=1,...,n$. And $\bar{X}=\frac{1}{n}\sum_{i=1}^n X_i$ has mean 0 and true covariance matrix $\frac{1}{n}\Sigma$.

Let $Y_i=LX_i\in\mathbb{R}^p$. Then Y_i are i.i.d with mean 0 and true covariance matrix $L\Sigma L^T=L\left(L^TL\right)^{-1}L^T=\mathbf{I}_p$. And $\bar{Y}=\frac{1}{n}\sum_{i=1}^n Y_i=\frac{1}{n}\sum_{i=1}^n LX_i=L\bar{X}$ has mean zero and covariance matrix $\frac{1}{n}\mathbf{I}_p$. Also the corresponding design matrix $\mathbf{Y}=\mathbf{X}L^T$ has independent rows.

$$\begin{aligned}SL^T&=\frac{1}{n}\sum_{i=1}^n\left(X_i-\bar{X}\right)\left(X_i-\bar{X}\right)^TL^T\\ &=\frac{1}{n}\sum_{i=1}^n\left(X_i-\bar{X}\right)\left(LX_i-L\bar{X}\right)^T=\frac{1}{n}\sum_{i=1}^n\left(X_i-\bar{X}\right)\left(Y_i-\bar{Y}\right)^T.\end{aligned}$$

So we have

$$\left(SL^T\right)_{ij}=n^{-1}\sum_{k=1}^p X_{ki}Y_{kj}-\bar{X}_i\bar{Y}_j.$$

Letting

$$\mathcal{W}=SL^T-L^{-1},$$

we have that

$$\left|\mathcal{W}_{ij}\right|\leq\left|n^{-1}\sum_{k=1}^p X_{ki}Y_{kj}-\left(L^{-1}\right)_{ij}\right|+\left|\bar{X}_i\bar{Y}_j\right|.$$

$$
\begin{aligned}
&\mathrm{P}\left[\max_{ij}|\mathcal{W}|_{ij} > t\right] \\
\leq &\mathrm{P}\left[\max_{ij}\left|n^{-1}\sum_{k=1}^{p} X_{ki}Y_{kj} - \left(L^{-1}\right)_{ij}\right| > \frac{t}{2}\right] + \mathrm{P}\left[\max_{ij}\left|\bar{X}_i\bar{Y}_j\right| > \frac{t}{2}\right] \\
\leq &\mathrm{P}\left[\left|n^{-1}\sum_{k=1}^{p} X_{ki}Y_{kj} - \left(L^{-1}\right)_{ij}\right| > \frac{t}{2} \text{ for some } i,j\right] \\
&+ \mathrm{P}\left[\max_{i}\left|\bar{X}_i\right| > \sqrt{\frac{t}{2}}\right] + \mathrm{P}\left[\max_{j}\left|\bar{Y}_j\right| > \sqrt{\frac{t}{2}}\right] \\
\leq &\sum_{ij}\mathrm{P}\left[\left|n^{-1}\sum_{k=1}^{p} X_{ki}Y_{kj} - \left(L^{-1}\right)_{ij}\right| > \frac{t}{2}\right] + \sum_{i}\mathrm{P}\left[\left|\bar{X}_i\right| > \sqrt{\frac{t}{2}}\right] + \sum_{j}\mathrm{P}\left[\left|\bar{Y}_j\right| > \sqrt{\frac{t}{2}}\right] \\
\leq &p^2\max_{ij}\mathrm{P}\left[\left|n^{-1}\sum_{k=1}^{p} X_{ki}Y_{kj} - \left(L^{-1}\right)_{ij}\right| > \frac{t}{2}\right] \\
&+ p\max_{i}\mathrm{P}\left[\left|\bar{X}_i\right| > \sqrt{\frac{t}{2}}\right] + p\max_{j}\mathrm{P}\left[\left|\bar{Y}_j\right| > \sqrt{\frac{t}{2}}\right] \\
:= &p^2\max_{ij} I_{ij} + p\max_{i} I_i^X + p\max_{j} I_j^Y .
\end{aligned}
$$

Consider I_i^X first. Since X_{ki} are independent sub-Gaussian with variance Σ_{ii} for $k = 1, .., n$, we have

$$
\begin{aligned}
\mathrm{E}\exp\left(t\frac{\bar{X}_i}{\sqrt{\Sigma_{ii}/n}}\right) &= \prod_{k=1}^{n}\mathrm{E}\exp\left(t\frac{X_{ki}}{\sqrt{n\Sigma_{ii}}}\right) \quad \text{by independence} \\
&\leq \prod_{k=1}^{n}\exp\left(\tilde{C}_1 t^2/n\right) = \exp(\tilde{C}_1 t^2) \quad \text{by the definition of sub-Gaussian,}
\end{aligned}
$$

so $\bar{X}_i$ is sub-Gaussian with variance Σ_{ii}/n.

By Lemma 5.5 in Vershynin (2010), we have

$$
\mathrm{P}\left[\left|\bar{X}_i\right|/\sqrt{\Sigma_{ii}^*} > t\right) \leq \exp\left(1 - t^2/K_1^2\right],
$$

where K_1 is a constant that does not depend on i.

Following the same argument we have

$$
\mathrm{E}\exp\left(t\bar{Y}_i/\sqrt{1/n}\right) = \prod_{k=1}^{n}\mathrm{E}\exp\left(tY_{ki}/\sqrt{n}\right) \leq \exp\left(\tilde{C}_2 t^2\right),
$$

thus

$$
\mathrm{P}\left[\left|\bar{Y}_i\right|/\sqrt{1/n} > t\right) \leq \exp\left(1 - t^2/K_2^2\right],
$$

where K_2 is a constant that does not depend on i. And we have

$$
\begin{aligned}
I_i^X + I_i^Y &= \mathrm{P}\left[\left|\bar{X}_i\right| > \sqrt{t/2}\right] + \mathrm{P}\left[\left|\bar{Y}_i\right| > \sqrt{t/2}\right] \\
&= \mathrm{P}\left[\frac{\left|\bar{X}_i\right|}{\sqrt{\Sigma_{ii}/n}} > \frac{\sqrt{t/2}}{\sqrt{\Sigma_{ii}/n}}\right] + \mathrm{P}\left[\left|\frac{\bar{Y}_i}{\sqrt{1/n}}\right| > \frac{\sqrt{t/2}}{\sqrt{1/n}}\right] \\
&\leq \exp\left(1 - \frac{nt}{2K_1^2\Sigma_{ii}^*}\right) + \exp\left(1 - \frac{nt}{2K_2^2}\right).
\end{aligned}
$$

Thus

$$
\max_i\left(I_i^X + I_i^Y\right) \leq 4\exp\left(-\frac{C_1 nt}{\max_i \Sigma_{ii}^*}\right) + 4\exp\left(-C_2 nt\right)
$$

for some constant C_1.

Now consider the term I_{ij}. We have shown that both $\mathbf{X}$ and $\mathbf{Y}$ have independent rows. So for any i, j, $Z_k^{(ij)} = X_{ki}Y_{kj}$ are independent for $k = 1, \ldots, n$. Let $X \sim N(\mathbf{0}, \Sigma)$ and $Y \sim N(0, \mathbf{I}_p)$, then

$$
\mathrm{E}\left(X_{ki}Y_{kj}\right) = \mathrm{Cov}\left(X, LX\right)_{ij} - 0 = \left[\mathrm{Cov}\left(X, X\right)L^T\right]_{ij} = \left(\Sigma L^T\right)_{ij} = \left(L^{-1}\right)_{ij}.
$$

If there exist ν_{ij} and c_{ij} such that

$$
\begin{aligned}
&\sum_{k=1}^{n}\mathrm{E}\left(X_{ki}^2Y_{kj}^2\right) \leq \nu_{ij} \\
&\sum_{k=1}^{n}\mathrm{E}\left\{\left(X_{ki}Y_{kj}\right)_+^q\right\} \leq \frac{q!}{2}\nu_{ij}c_{ij}^{q-2} \quad \text{for some} \quad q \geq 3 \in \mathbb{N},
\end{aligned}
$$

then by Theorem 2.10 (Corollary 2.11) in Boucheron et al. (2013), $\forall t > 0$, we have

$$
\mathrm{P}\left[\left|\sum_{k=1}^{n}\left(X_{ki}Y_{kj} - (L)_{ij}^{-1}\right)\right| > t\right] \leq 2\exp\left(-\frac{t^2}{2\left(\nu_{ij} + c_{ij}t\right)}\right).
$$

The rest of the proof focuses on characterizing ν_{ij} and c_{ij}. First, Lemma 5.5 in Vershynin (2010) shows that, for some constant K_3 that does not depend on j,

$$
\left(\mathrm{E}\left|X_{ij}/\sqrt{\Sigma_{jj}}\right|^q\right)^{1/q} \leq K_3\sqrt{q}
$$

holds for all $q \geq 1$. Thus,

$$
\mathrm{E}\left|X_{ij}\right|^q \leq K_3^q q^{q/2}\left(\Sigma_{jj}\right)^{q/2}.
$$

Following the same argument, there exists some constant K_4 that does not depend on j such that

$$
\mathrm{E}\left|Y_{ij}\right|^q \leq K_4^q q^{q/2}
$$

for all $q \geq 1$.

Therefore,

$$\sum_{k=1}^{n} \mathrm{E}\left(X_{ki}^2 Y_{kj}^2\right) \leq \sum_{k=1}^{n} \sqrt{\mathrm{E}X_{ki}^4 \mathrm{E}Y_{kj}^4} \leq n\sqrt{K_3^4 2^4 K_4^4 2^4 {\Sigma_{ii}}^2} = 16nK_3^2 K_4^2 \Sigma_{ii},$$

and

$$\sum_{k=1}^{n} \mathrm{E}\left\{(X_{ki}Y_{kj})_+^q\right\} \leq \sum_{k=1}^{n} \sqrt{\mathrm{E}X_{ki}^{2q} \mathrm{E}Y_{kj}^{2q}} \leq n\sqrt{K_3^{2q}(2q)^{2q} K_4^{2q} \left(\Sigma_{ii}\right)^2} = nK_3^q K_4^q (2q)^q \left(\Sigma_{ii}\right)^{q/2}.$$

So taking

$$\nu_{ij} = K_5 n \Sigma_{ii}^*,$$
$$c_{ij} = K_5 \sqrt{\Sigma_{ii}^*}$$

for some K_5 large enough and does not depend on i, j.

Now we have

$$I_{ij} \leq 2\exp\left(-\frac{n^2t^2}{4\left(2\nu_{ij} + c_{ij}tn\right)}\right) = 2\exp\left(-\frac{nt^2}{4\left(2K_5\Sigma_{ii}^* + K_5\sqrt{\Sigma_{ii}}t\right)}\right).$$

If $t \leq 2\max_i \sqrt{\Sigma_{ii}^*}$, then with $C_3 = (16K_5)^{-1}$ we have

$$I_{ij} \leq 2\exp\left(-\frac{C_2 nt^2}{\max_i \Sigma_{ii}^*}\right).$$

To sum up, for any $0 < t \leq 2\max_i \sqrt{\Sigma_{ii}^*}$,

$$\mathrm{P}\left[\max_{ij}|\mathcal{W}_{ij}| > t\right] \leq 2p^2\exp\left(-\frac{C_2 nt^2}{\max_i \Sigma_{ii}^*}\right) + 4p\exp\left(-\frac{C_1 nt}{\max_i \Sigma_{ii}^*}\right) + 4p\exp\left(-C_2 nt\right).$$

■

References

Bryon Aragam and Qing Zhou. Concave penalized estimation of sparse gaussian bayesian networks. *Journal of Machine Learning Research*, 16:2273–2328, 2015.

Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *The Journal of Machine Learning Research*, 9:485–516, 2008.

Peter J Bickel and Elizaveta Levina. Regularized estimation of large covariance matrices. *The Annals of Statistics*, 36(1):199–227, 2008.

Jacob Bien, Florentina Bunea, and Luo Xiao. Convex banding of the covariance matrix. *Journal of the American Statistical Association*, 111(514):834–845, 2016.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. OUP Oxford, 2013.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

Tony Cai, Weidong Liu, and Xi Luo. A constrained ℓ_1 minimization approach to sparse precision matrix estimation. *Journal of the American Statistical Association*, 106(494): 594–607, 2011.

International HapMap 3 Consortium et al. Integrating common and rare genetic variation in diverse human populations. *Nature*, 467(7311):52–58, 2010.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning; Data Mining, Inference and Prediction, Second Edition*. Springer Verlag, New York, 2009.

Jianhua Z Huang, Naiping Liu, Mohsen Pourahmadi, and Linxu Liu. Covariance matrix selection and estimation via penalised normal likelihood. *Biometrika*, 93(1):85–98, 2006.

Rodolphe Jenatton, Jean-Yves Audibert, and Francis Bach. Structured variable selection with sparsity-inducing norms. *The Journal of Machine Learning Research*, 12:2777–2824, 2011.

K. Khare, S. Oh, S. Rahman, and B. Rajaratnam. A convex framework for high-dimensional sparse Cholesky based covariance estimation. *ArXiv e-prints*, October 2016.

Kshitij Khare, Sang-Yun Oh, and Bala Rajaratnam. A convex pseudolikelihood framework for high dimensional partial correlation estimation with convergence guarantees. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 77(4):803–825, 2014.

Elizaveta Levina, Adam Rothman, and Ji Zhu. Sparse estimation of large covariance matrices via a nested lasso penalty. *The Annals of Applied Statistics*, 2(1):245–263, 2008.

Han Liu and Lie Wang. Tiger: A tuning-insensitive approach for optimally estimating gaussian graphical models. *arXiv preprint arXiv:1209.2437*, 2012.

Weidong Liu and Xi Luo. High-dimensional sparse precision matrix estimation via sparse column inverse operator. *arXiv preprint arXiv:1203.3896*, 2012.

Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, pages 1436–1462, 2006.

Jie Peng, Pei Wang, Nengfeng Zhou, and Ji Zhu. Partial correlation estimation by joint sparse regression models. *Journal of the American Statistical Association*, 104(486):735–746, 2009.

Mohsen Pourahmadi. Joint mean-covariance models with applications to longitudinal data: Unconstrained parameterisation. *Biometrika*, 86(3):677–690, 1999.

Mohsen Pourahmadi. *High-dimensional covariance estimation: with high-dimensional data*. John Wiley & Sons, 2013.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2016. URL `https://www.R-project.org/`.

Pradeep Ravikumar, Martin J Wainwright, Garvesh Raskutti, and Bin Yu. High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 5:935–980, 2011.

Adam J Rothman, Peter J Bickel, Elizaveta Levina, and Ji Zhu. Sparse permutation invariant covariance estimation. *Electronic Journal of Statistics*, 2:494–515, 2008.

Ali Shojaie and George Michailidis. Penalized likelihood methods for estimation of sparse high-dimensional directed acyclic graphs. *Biometrika*, 97(3):519–538, 2010.

Nicolas Städler, Peter Bühlmann, and Sara van de Geer. l1-penalization for mixture regression models (with discussion). *Test*, 19:209–285, 2010.

Tingni Sun and Cun-Hui Zhang. Comments on: l1-penalization for mixture regression models. *Test*, 19(2):270–275, 2010.

Tingni Sun and Cun-Hui Zhang. Scaled sparse linear regression. *Biometrika*, 99(4):879–898, 2012.

Tingni Sun and Cun-Hui Zhang. Sparse matrix inversion with scaled lasso. *The Journal of Machine Learning Research*, 14(1):3385–3418, 2013.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

Sara A. Van de Geer and Peter Bühlmann. On the conditions used to prove oracle results for the lasso. *Electronic Journal of Statistics*, 3:1360–1392, 2009.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

Martin J Wainwright. Sharp thresholds for high-dimensional and noisy sparsity recovery using-constrained quadratic programming (lasso). *Information Theory, IEEE Transactions on*, 55(5):2183–2202, 2009.

Wei Biao Wu and Mohsen Pourahmadi. Nonparametric estimation of large covariance matrices of longitudinal data. *Biometrika*, 90(4):831–844, 2003.

Xiaohan Yan and Jacob Bien. Hierarchical sparse modeling: A choice of two regularizers. *arXiv preprint arXiv:1512.01631*, 2015.

Ming Yuan. High dimensional inverse covariance matrix estimation via linear programming. *The Journal of Machine Learning Research*, 11:2261–2286, 2010.

Ming Yuan and Yi Lin. Model selection and estimation in the gaussian graphical model. *Biometrika*, 94(1):19–35, 2007.

Teng Zhang and Hui Zou. Sparse precision matrix estimation via lasso penalized d-trace loss. *Biometrika*, 101(1):103–120, 2014.

Peng Zhao and Bin Yu. On model selection consistency of lasso. *The Journal of Machine Learning Research*, 7:2541–2563, 2006.

Peng Zhao, Guilherme Rocha, and Bin Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, 37(6A):3468–3497, 2009.

Dale L Zimmerman and Vicente A Nunez-Anton. *Antedependence models for longitudinal data*. CRC Press, 2009.

Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006.

Journal of Machine Learning Research 18 (2017) 1-29 Submitted 8/16; Revised 12/16; Published 4/17

Bayesian Learning of Dynamic Multilayer Networks

Daniele Durante DURANTE@STAT.UNIPD.IT
Department of Statistical Sciences
University of Padova
Padova, 35121, Italy

Nabanita Mukherjee NM168@DUKE.EDU
Department of Statistical Science
Duke University
Durham, NC 27708-0251, USA

Rebecca C. Steorts RCS46@STAT.DUKE.EDU
Departments of Statistical Science and Computer Science
Duke University
Durham, NC 27708-0251, USA

Editor: Amos Storkey

Abstract

A plethora of networks is being collected in a growing number of fields, including disease transmission, international relations, social interactions, and others. As data streams continue to grow, the complexity associated with these highly multidimensional connectivity data presents novel challenges. In this paper, we focus on the time-varying interconnections among a set of actors in multiple contexts, called layers. Current literature lacks flexible statistical models for dynamic multilayer networks, which can enhance quality in inference and prediction by efficiently borrowing information within each network, across time, and between layers. Motivated by this gap, we develop a Bayesian nonparametric model leveraging latent space representations. Our formulation characterizes the edge probabilities as a function of shared and layer-specific actors positions in a latent space, with these positions changing in time via Gaussian processes. This representation facilitates dimensionality reduction and incorporates different sources of information in the observed data. In addition, we obtain tractable procedures for posterior computation, inference, and prediction. We provide theoretical results on the flexibility of our model. Our methods are tested on simulations and infection studies monitoring dynamic face-to-face contacts among individuals in multiple days, where we perform better than current methods in inference and prediction.

Keywords: Dynamic multilayer network, edge prediction, face-to-face contact network, Gaussian process, latent space model

1. Introduction

Data on social interaction processes are rapidly becoming highly multidimensional, increasingly complex and inherently dynamic, providing a natural venue for machine learning as seen in applications such as disease transmission (Marathe and Ramakrishnan, 2013), international relations (Schein et al., 2016), social interactions (Guo et al., 2015), and others.

©2017 Durante, Mukherjee and Steorts.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-391.html.

While modeling of a single network observation is still an active area of research, the increasing availability of multidimensional networks from World Wide Web architectures (Leetaru and Schrodt, 2013), telecommunication infrastructures (Blondel et al., 2015) and human sensing devices (Cattuto et al., 2010), have motivated more flexible statistical models. This endeavor is particularly relevant in the dynamic multilayer network field, providing information on the time-varying connectivity patterns among a set of actors in different contexts. Notable examples include dynamic relationships between individuals according to multiple forms of social interactions (Snijders et al., 2013), and time-varying connectivity structures among countries based on different types of international relations (Hoff, 2015).

In modeling these highly complex networks, it is of paramount interest to learn the wiring processes underlying the observed data and to infer differences in network structures across layers and times. Improved estimation of the data's generating mechanism can refine the understanding of social processes and enhance the quality in prediction of future networks. In order to successfully accomplish these goals, it is important to define statistical models which can incorporate the different sources of information in the observed data, without affecting flexibility. However, current literature lacks similar methods, to our knowledge.

Motivated by this gap, we develop a Bayesian nonparametric model for dynamic multilayer networks which efficiently incorporates dependence within each network, across time, and between the different layers, while preserving flexibility. Our formulation borrows network information by defining the edge probabilities as a function of pairwise similarities between actors in a latent space. To share information among layers without affecting flexibility in characterizing layer-specific structures, we force a subset of the latent coordinates of each actor to be common across layers, and let the remaining coordinates to vary between layers. Finally, we accommodate network dynamics by allowing these actors coordinates to change in time, and incorporate time information by modeling the dynamic actors coordinates via Gaussian processes (e.g., Rasmussen and Williams, 2006). Our model is tractable and has a theoretical justification, while providing simple procedures for inference and prediction. In addition, we find that our procedures outperform current methods in inference and out-of-sample prediction on both simulated and real data.

The paper proceeds as follows. Section 1.1 reviews recent contributions in the literature, and Section 1.2 provides a motivational example of dynamic multilayer networks. Section 2 describes our Bayesian latent space model, while Section 3 discusses its properties. Methods for posterior computation and prediction are provided in Section 4. Section 5 highlights the performance gains of our model compared to other possible competitors in a simulation study, whereas Section 6 illustrates on infection studies, monitoring face-to-face contacts, our success in inference and prediction, compared to possible competitors. Future directions of research and additional fields of application are discussed in Section 7.

1.1 Related Work

Current literature for multidimensional network data considers settings in which the multiple networks are either dynamic or multilayer. Statistical modeling of dynamic networks has focused on stochastic processes which are designed to borrow information between edges and across time (e.g., Hanneke et al., 2010; Snijders et al., 2010; Xing et al., 2010; Yang et al., 2011; Durante and Dunson, 2014; Crane, 2016), whereas inference for multilayer net-

works has motivated formulations which can suitably induce dependence between edges and across the different types of relationships—characterizing the multiple layers (e.g., Gollini and Murphy, 2016; Han et al., 2015; Heaney, 2014). These contributions have generalized exponential random graph models (Holland and Leinhardt, 1981; Frank and Strauss, 1986) and latent variables models (Nowicki and Snijders, 2001; Hoff et al., 2002; Airoldi et al., 2008) for a single network to allow inference in multidimensional frameworks, when the multiple networks arise either from dynamic or multilayer studies. These methods are valuable building blocks for more flexible models, but fall far short of the goal of providing efficient procedures in more complex settings when the networks are both dynamic and multilayer.

The routine collection of dynamic multilayer networks is a recent development and statistical modeling of such data is still in its infancy. For example, Lee and Monge (2011) considered a generalization of exponential random graph models for multilayer networks, but performed a separate analysis for each time point. Oselio et al. (2014) focused instead on a dynamic stochastic block model which borrows information across time and within each network, but forces the underlying block structures to be shared between layers. Dynamic multilayer networks are complex objects combining homogenous structures shared between actors, layers, and smoothly evolving across time, with layer-specific patterns and across-actor heterogeneity. Due to this, any procedure that fails to incorporate the different sources of information in the observed data (e.g., Lee and Monge, 2011) is expected to lose efficiency, whereas models focusing on shared patterns (e.g., Oselio et al., 2014) may lack flexibility.

More general formulations are the multilayer stochastic actor-oriented model (Snijders et al., 2013) and the multilinear tensor regression (Hoff, 2015). Snijders et al. (2013) allowed for dynamic inference on network properties within and between layers, but failed to incorporate across-actor heterogeneity. This may lead to a lack of flexibility in prediction. Hoff (2015) considered autoregressive models with the vector of parameters having a tensor factorization representation. This formulation allows for across-actor heterogeneity, but forces the model parameters to be constant across time. In addition, the parameterization of the interdependence between layers relies on homogeneity assumptions. Consistent with these methods, our representation incorporates the different types of dependencies in the observed data, but crucially preserves flexibility to avoid restrictive homogeneity assumptions.

1.2 Motivating Application

Our motivation is drawn from epidemiologic studies monitoring hourly face-to-face contacts among individuals in a rural area of Kenya, during three consecutive days. Data are available from the human sensing platform SocioPatterns (`http://www.sociopatterns.org`) and have been collected using wearable devices that exchange low-power radio packets when two individuals are located within a sufficiently close distance to generate a potential occasion of contagion. Leveraging this technology it is possible to measure for each hour in the three consecutive days which pairs of actors had a face-to-face proximity contact. These information are fundamental to monitor the spread of diseases and learn future patterns. Refer also to Cattuto et al. (2010) for a detailed description of the data collection technology, and Kiti et al. (2016) for a more in-depth discussion of our motivating application.

As illustrated in Figure 1, the observed data can be structured as a dynamic multilayer network characterized by a sequence of $V \times V$ symmetric adjacency matrices $Y_{t_i}^{(k)}$ collected

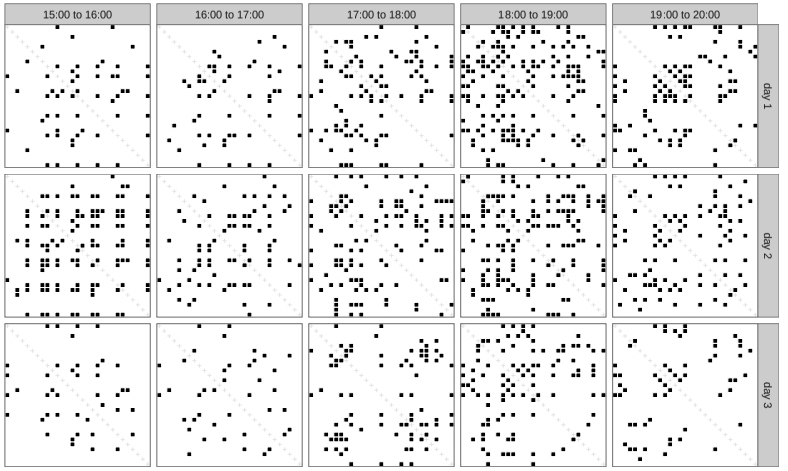


Figure 1: For five consecutive hours in the three days, adjacency matrices $Y_{t_i}^{(k)}$ representing the observed contact networks. Black refers to an edge, and white to a non-edge.

on a common time grid $t_1, \ldots, t_n$ in different days $k = 1, \ldots, K$. The matrices have entries $Y_{t_i[vu]}^{(k)} = Y_{t_i[uv]}^{(k)} = 1$ if actors v and u interact at time t_i in day k, and $Y_{t_i[vu]}^{(k)} = Y_{t_i[uv]}^{(k)} = 0$ otherwise, for every $v = 2, \ldots, V$ and $u = 1, \ldots, v-1$. Dynamic multilayer proximity data provide key information on infections spread (Cauchemez et al., 2011), facilitating the design of outbreak prevention policies (Vanhems et al., 2013). The importance of this endeavor has motivated a wide variety of descriptive studies summarizing the information provided by face-to-face proximity data in several environments, including hospitals (Vanhems et al., 2013), schools (Stehlé et al., 2011) and households (Kiti et al., 2016).

Although the above analyses provide key insights on the connectivity processes underlying the observed proximity data, explicitly accounting for variability in network structures via carefully tailored statistical models can lead to improved learning of the connectivity patterns and properties, while providing methodologies for formal inference in the network framework, including estimation techniques, uncertainty quantification and prediction. In Section 6, we apply our methodology to this motivating application, showing how a careful statistical model for dynamic multilayer network data can provide substantial performance gains in learning underlying social processes and predicting future contact patterns.

2. Dynamic Multilayer Latent Space Model

Let $Y_{t_i}^{(k)}$ denote the $V \times V$ adjacency matrix characterizing the undirected network observed at time $t_i \in \Re^+$ in layer k, for any $t_i = t_1, \ldots, t_n$ and $k = 1, \ldots, K$. Each matrix has binary elements $Y_{t_i[vu]}^{(k)} = Y_{t_i[uv]}^{(k)} \in \{0, 1\}$ measuring the absence or presence of an edge among actors

Type of information	**Illustrative example**
Network information	If individual v had a face-to-face contact with both actors u and w at time t_i in day k, this information may be relevant to learn the face-to-face contact behavior between u and w at time t_i in day k.
Layer information	If individuals v and u had a face-to-face contact at time t_i in day k, this information may be relevant to learn the contact behavior between v and u at the same time t_i in other days.
Time information	If individuals v and u had a face-to-face contact at time t_i in day k, this information may be relevant to learn the contact behavior between v and u at the next time t_{i+1} in the same day.

Table 1: Relevant sources of information characterizing a dynamic multilayer network.

v and u at time t_i in layer k, for every $v = 2, \ldots, V$ and $u = 1, \ldots, v-1$. When modeling the data $Y = \{Y_{t_i}^{(k)} : t_i = t_1, \ldots, t_n,\ k = 1, \ldots, K\}$, we assume that Y is a realization, on a finite time grid, of the stochastic process $\mathcal{Y} = \{\mathcal{Y}^{(k)}(t) : t \in \mathbb{T} \subset \Re^+,\ k = 1, \ldots, K\}$, and seek a provably flexible representation for the probabilistic generative mechanism associated with the stochastic process $\mathcal{Y}$. Note that, as each random adjacency matrix $\mathcal{Y}^{(k)}(t)$ is symmetric and the diagonal elements—denoting self-relations—are not of interest, it is sufficient to model its lower triangular elements $\{\mathcal{Y}_{vu}^{(k)}(t) : v = 2, \ldots, V,\ u = 1, \ldots, v-1\}$.

One major modeling objective is to carefully incorporate dependence among edges, between layers and across time, without affecting flexibility. Recalling the motivating application in Section 1.2 and Figure 1, it is reasonable to expect three main sources of information in the dynamic multilayer face-to-face contact data, summarized in Table 1. Incorporating such information can substantially improve the quality of inference and prediction, while facilitating dimensionality reduction. However, in reducing dimensionality and borrowing of information, it is important to avoid restrictive formulations that lead to inadequate characterizations of dynamic patterns, layer-specific structures, and across-actor heterogeneity.

We accomplish the aforementioned goals via a flexible dynamic latent bilinear model, combining shared and layer-specific actors coordinates which are allowed to change in time via Gaussian process priors. Section 2.1 describes our model formulation with a focus on sharing network and layer information, whereas Section 2.2 clarifies how time information is effectively incorporated via Gaussian processes.

2.1 Dynamic Bilinear Model with Shared and Layer-Specific Coordinates

In order to incorporate network and layer information, while preserving flexibility, we adapt latent bilinear models for a single network (Hoff, 2005) to our dynamic multilayer framework. In particular, we characterize the edges as conditionally independent Bernoulli random variables given their corresponding edge probabilities, and borrow information within each network by defining these probabilities as a function of the actors coordinates in a latent space. In order to incorporate dependence among the different layers, we force a subset of these coordinates to be common across layers, and allow the remaining coordinates to change

between layers. This last choice facilitates a flexible characterization of more irregular events and incorporates differences between layers in specific contact patterns.

Consistent with the aforementioned statistical model, and letting $\pi_{vu}^{(k)}(t) \in (0,1)$ denote the probability of an edge between actors v and u at time $t \in \mathbb{T}$ in layer k, we define

$$\mathcal{Y}_{vu}^{(k)}(t) \mid \pi_{vu}^{(k)}(t) \sim \text{Bern}\{\pi_{vu}^{(k)}(t)\}, \tag{1}$$

independently for every time $t \in \mathbb{T}$, layer $k = 1, \ldots, K$ and pair of actors $[vu]$, $v = 2, \ldots, V$, $u = 1, \ldots, v-1$. To incorporate our multilayer bilinear representation having shared and layer-specific latent coordinates, we express the log-odds of each edge probability as

$$\text{logit}\{\pi_{vu}^{(k)}(t)\} = \mu(t) + \sum_{r=1}^{R} \bar{x}_{vr}(t)\bar{x}_{ur}(t) + \sum_{h=1}^{H} x_{vh}^{(k)}(t)x_{uh}^{(k)}(t), \tag{2}$$

where $\bar{x}_{vr}(t) \in \Re$ is the rth coordinate of actor v at time t shared across the different layers, whereas $x_{vh}^{(k)}(t) \in \Re$ denotes the hth coordinate of actor v at time t specific to layer k, for every $t \in \mathbb{T}$, $k = 1, \ldots, K$, $v = 1, \ldots, V$, $r = 1, \ldots, R$ and $h = 1, \ldots, H$. Finally, $\mu(t) \in \Re$ represents a time-varying baseline parameter centering the log-odds processes to improve computational and mixing performance.

In equation (2) the probability of an edge between actors v and u at time $t \in \mathbb{T}$ in layer k, increases with $\sum_{r=1}^{R} \bar{x}_{vr}(t)\bar{x}_{ur}(t)$ and $\sum_{h=1}^{H} x_{vh}^{(k)}(t)x_{uh}^{(k)}(t)$. Note that $\sum_{r=1}^{R} \bar{x}_{vr}(t)\bar{x}_{ur}(t) \in \Re$ characterizes a similarity measure between actors v and u at time t common to all the layers, whereas $\sum_{h=1}^{H} x_{vh}^{(k)}(t)x_{uh}^{(k)}(t) \in \Re$ defines a layer-specific deviation from this shared similarity, which enhances flexibility in modeling network structures specific to layer k. These similarity measures are defined as the dot product of shared and layer-specific actors coordinates in a latent space, allowing actors with coordinates in the same direction to have an higher chance of contact than actors with coordinates in opposite directions.

The dot product characterization in equation (2) is in the same spirit as the factorization in Hoff (2005) for a single network, and allows dimensionality reduction from $n \times K \times V \times (V-1)/2$ time-varying log-odds to $n \times \{1 + V \times (R + H \times K)\}$ dynamic latent coordinates, where typically $R \ll V$ and $H \ll V$. Although it is possible to consider other concepts of similarity when relating the latent coordinates to the log-odds (e.g., Hoff et al., 2002), the dot product representation facilitates the definition of simple algorithms for posterior inference and has been shown to efficiently borrow information within the network, while accommodating topological structures of interest such as homophily, transitivity, and others.

2.1.1 Interpretation and Identifiability

Recall the motivating application described in Section 1.2, our representation has an intuitive interpretation. In fact, the latent coordinates of each actor may represent his propensity towards $R + H$ latent interests or tasks. According to the factorization in equation (2), actors having propensities for the different interests in the same directions are expected to be more similar than actors with interests in opposite directions. Therefore, these actors are more likely to interact according to equation (1). Part of these actors interests or tasks may be associated with a common routine or constrained by daytime schedules. Therefore, the actors propensities towards these interests are expected to remain constant across days

providing a subset of R shared coordinates. Finally, these propensities are allowed to change across time inducing dynamic variations in the contacts within each day.

Although inference for the latent coordinates is potentially of interest, the factorization in equation (2) is not unique. In fact, it is easy to show that there exist many different latent coordinates values leading to the same collection of edge probabilities under the factorization in equation (2). However, such an overcomplete representation does not lead to identifiability issues if inference focuses on identified functionals of the latent coordinates, and has beneficial effects in terms of posterior computation and borrowing of information (Bhattacharya and Dunson, 2011; Ghosh and Dunson, 2009). Our focus is not on estimating the latent coordinates, but on leveraging the factorization in equation (2) to enhance efficiency in learning of identified functionals of the edge probabilities and improve prediction of future networks. Therefore, we avoid identifiability constraints on the latent coordinates in equation (2) as they are not required to ensure identifiability of the edge probabilities.

2.2 Gaussian Process Priors for the Time-Varying Latent Coordinates

Equation (2) facilitates borrowing of information among edges and between layers. This is obtained by leveraging the shared dependence on a common set of latent coordinates. We additionally incorporate across-layer heterogeneity via a set of layer-specific latent positions. Moreover, these coordinates are allowed to vary across time in order to accommodate flexible dynamic variations in the multilayer network structure. While it is possible to estimate the edge probabilities separately for every time point, this approach may be suboptimal in ruling out relevant temporal dependence. As seen in Figure 1, it is reasonable to expect a degree of dependence between the contact networks observed at close times, with this dependence decreasing with the time lag. Therefore, a formulation sharing information across time may provide substantial benefits in terms of efficiency, uncertainty propagation, and prediction.

Motivated by dynamic modeling of a single network, Durante and Dunson (2014) addressed a related goal by considering Gaussian process priors for the actors latent coordinates, with an additional shrinkage effect to facilitate automatic adaptation of the latent space dimensions and avoid overfitting. In the spirit of their proposed methods, we define

$$\bar{x}_{vr}(\cdot) \sim \mathrm{GP}(0, \tau_r^{-1} c_{\bar{x}}), \qquad \text{with} \;\; c_{\bar{x}}(t_i, t_j) = \exp\{-\kappa_{\bar{x}}(t_i - t_j)^2\}, \kappa_{\bar{x}} > 0, \tag{3}$$

independently for $v = 1, \ldots, V$, $r = 1, \ldots, R$ and

$$x_{vh}^{(k)}(\cdot) \sim \mathrm{GP}(0, \tau_h^{(k)-1} c_x), \quad \text{with} \;\; c_x(t_i, t_j) = \exp\{-\kappa_x(t_i - t_j)^2\}, \kappa_x > 0, \tag{4}$$

independently for $v = 1, \ldots, V$, $h = 1, \ldots, H$ and $k = 1, \ldots, K$. In the prior specification given in equations (3)–(4), the quantities $c_{\bar{x}}(t_i, t_j)$ and $c_x(t_i, t_j)$ denote the squared exponential correlation functions of the Gaussian processes for the shared and layer-specific latent coordinates, respectively. The quantities $\tau_1^{-1}, \ldots, \tau_R^{-1}$ and $\tau_1^{(k)-1}, \ldots, \tau_H^{(k)-1}$, for each $k = 1, \ldots, K$, are instead positive shrinkage parameters controlling the concentration of the latent coordinates around the zero constant mean function. Focusing on the prior for the trajectories of the rth shared coordinates $\bar{x}_{vr}(\cdot)$, $v = 1, \ldots, V$, a value of τ_r^{-1} close to zero forces these trajectories to be concentrated around the zero constant mean function, reducing the effect of the rth shared dimension in defining the edge probabilities in equation (2).

In order to penalize highly parameterized representations and facilitate adaptive deletion of unnecessary dimensions, the above shrinkage effects are designed to be increasingly strong as the latent coordinates indices $r = 1, \ldots, R$ and $h = 1, \ldots, H$ increase. This goal is accomplished by considering the following multiplicative inverse gamma priors (Bhattacharya and Dunson, 2011) for the shrinkage parameters:

$$\frac{1}{\tau_r} = \prod_{m=1}^{r} \frac{1}{\delta_m}, \;\; r = 1, \ldots, R, \;\text{with}\; \delta_1 \sim \mathrm{Ga}(a_1, 1), \;\; \delta_{m \geq 2} \sim \mathrm{Ga}(a_2, 1), \tag{5}$$

$$\frac{1}{\tau_h^{(k)}} = \prod_{l=1}^{h} \frac{1}{\delta_l^{(k)}}, \;\; h = 1, \ldots, H, \;\; k = 1, \ldots, K, \;\text{with}\; \delta_1^{(k)} \sim \mathrm{Ga}(a_1, 1), \;\; \delta_{l \geq 2}^{(k)} \sim \mathrm{Ga}(a_2, 1). \tag{6}$$

According to Bhattacharya and Dunson (2011), the multiplicative inverse gamma prior in equations (5)–(6) induces prior distributions for the shrinkage parameters with a cumulative shrinkage effect. In particular, these priors are increasingly concentrated close to zero as the indices $r = 1, \ldots, R$ and $h = 1, \ldots, H$ increase, for appropriate $a_1 \in \Re^+$ and $a_2 \in \Re^+$, facilitating adaptive dimensionality reduction and reducing overfitting issues. Refer also to Durante (2017) for an additional discussion on the multiplicative inverse gamma prior and its shrinkage properties. To conclude the prior specification, we let $\mu(\cdot) \sim \mathrm{GP}(0, c_\mu)$, with $c_\mu(t_i, t_j) = \exp\{-\kappa_\mu(t_i - t_j)^2\}$, $\kappa_\mu > 0$.

The Gaussian process prior provides an accurate choice in our setting which incorporates time dependence, allowing the amount of information shared between networks to increase as the time lag decreases. Beside improving efficiency, the Gaussian process prior can deal with multilayer networks observed at potentially unequally spaced time grids, and is closely related to the multivariate Gaussian distribution, providing substantial benefits in terms of computational tractability and interpretability. In fact, following Rasmussen and Williams (2006), equations (3)–(4) imply the following prior for the shared and layer-specific latent coordinates at the finite time grid $t_1, \ldots, t_n$ on which data are observed:

$$\{\bar{x}_{vr}(t_1), \ldots, \bar{x}_{vr}(t_n)\}^\intercal \sim \mathrm{N}_n(0, \tau_r^{-1} \Sigma_{\bar{x}}), \tag{7}$$

independently for $v = 1, \ldots, V$, $r = 1, \ldots, R$ and

$$\{x_{vh}^{(k)}(t_1), \ldots, x_{vh}^{(k)}(t_n)\}^\intercal \sim \mathrm{N}_n(0, \tau_h^{(k)-1} \Sigma_x), \tag{8}$$

independently for $v = 1, \ldots, V$, $h = 1, \ldots, H$, and $k = 1, \ldots, K$. In equations (7)–(8), the $n \times n$ variance and covariance matrices $\Sigma_{\bar{x}}$ and Σ_x have elements $\Sigma_{\bar{x}[ij]} = \exp\{-\kappa_{\bar{x}}(t_i - t_j)^2\}$ and $\Sigma_{x[ij]} = \exp\{-\kappa_x(t_i - t_j)^2\}$, respectively. The same holds for the baseline process obtaining $\{\mu(t_1), \ldots, \mu(t_n)\}^\intercal \sim \mathrm{N}_n(0, \Sigma_\mu)$.

3. Model Properties

In order to ensure accurate learning and prediction, it is important to guarantee that our factorization in equation (2) along with the Gaussian process priors for its components are sufficiently flexible to approximate a broad variety of dynamic multilayer edge probability processes. These properties are stated in Propositions 1 and 2. See Appendix A for detailed proofs. In particular, Proposition 1 guarantees that equation (2) is sufficiently flexible to

characterize any collection $\pi(t) = \{\pi_{vu}^{(k)}(t) \in (0,1) : k = 1,\ldots,K,\ v = 2,\ldots,V,\ u = 1,\ldots,v-1\}$ for every time $t \in \mathbb{T}$.

Proposition 1 *Let $\pi(t) = \{\pi_{vu}^{(k)}(t) \in (0,1) : k = 1,\ldots,K,\ v = 2,\ldots,V,\ u = 1,\ldots,v-1\}$ denote the collection of edge probabilities between the actors in each layer at time $t \in \mathbb{T}$. Then every collection $\pi(t)$ can be represented as in equation (2) for some R and H.*

Although Proposition 1 guarantees that factorization in equation (2) is sufficiently general, it is important to ensure that the same flexibility is maintained when defining priors on the parameters in equation (2). Proposition 2 guarantees that our model and prior choices induce a prior on the dynamic multilayer edge probability process with full support.

Proposition 2 *If $\mathbb{T}$ is compact, then for every $\pi^0 = \{\pi_{vu}^{0(k)}(t) \in (0,1) : t \in \mathbb{T},\ k = 1,\ldots,K,\ v = 2,\ldots,V,\ u = 1,\ldots,v-1\}$ and $\epsilon > 0$,*

$$\text{pr}\left(\sup_{t\in\mathbb{T}}\left[\sum_{k=1}^{K}\sqrt{\sum_{v=2}^{V}\sum_{u=1}^{v-1}\{\pi_{vu}^{(k)}(t) - \pi_{vu}^{0(k)}(t)\}^2}\right] < \epsilon\right) > 0,$$

with $\pi_{vu}^{(k)}(t)$ factorized as in equation (2) with Gaussian process priors on the latent coordinates.

Full prior support is a key property in ensuring that our Bayesian formulation assigns a positive probability to a neighborhood of every possible true dynamic multilayer edge probability process, avoiding flexibility issues that may arise if zero mass is assigned to a subset of the parameters space where the truth may be.

In order to highlight the dependence structures induced by our model and priors, we additionally study the prior variances and covariances associated with the log-odds processes $z = \{z_{vu}^{(k)}(t) = \text{logit}\{\pi_{vu}^{(k)}(t)\} \in \Re : t \in \mathbb{T},\ k = 1,\ldots,K,\ v = 2,\ldots,V,\ u = 1,\ldots,v-1\}$. By conditioning on the shrinkage parameters, and leveraging equations (7) and (8), we obtain

$$\text{var}\{z_{vu}^{(k)}(t) \mid \tau, \tau^{(k)}\} = 1 + \sum_{r=1}^{R}\tau_r^{-2} + \sum_{h=1}^{H}\tau_h^{(k)-2}, \quad \text{cov}\{z_{vu}^{(k)}(t), z_{pq}^{(k)}(t) \mid \tau, \tau^{(k)}\} = 1,$$

for every layer $k = 1,\ldots,K$, time $t \in \mathbb{T}$ and pairs of actors $[vu]$, $v > u$ and $[pq]$, $p > q$ with $p \neq v$ or $q \neq u$. The covariance between different layers, at the same time $t \in \mathbb{T}$ is

$$\text{cov}\{z_{vu}^{(k)}(t), z_{vu}^{(g)}(t) \mid \tau, \tau^{(k)}, \tau^{(g)}\} = 1 + \sum_{r=1}^{R}\tau_r^{-2}, \quad \text{cov}\{z_{vu}^{(k)}(t), z_{pq}^{(g)}(t) \mid \tau, \tau^{(k)}, \tau^{(g)}\} = 1,$$

for every pair of layers k and g with $k \neq g$, time $t \in \mathbb{T}$ and pairs of actors $[vu]$, $v > u$ and $[pq]$, $p > q$ with $p \neq v$ or $q \neq u$. Finally, the covariances at different times are

$$\text{cov}\{z_{vu}^{(k)}(t_i), z_{vu}^{(k)}(t_j) \mid \tau, \tau^{(k)}\} = e^{-\kappa_\mu(t_i-t_j)^2} + \sum_{r=1}^{R}\tau_r^{-2}e^{-2\kappa_{\bar{x}}(t_i-t_j)^2} + \sum_{h=1}^{H}\tau_h^{(k)-2}e^{-2\kappa_x(t_i-t_j)^2},$$

$$\text{cov}\{z_{vu}^{(k)}(t_i), z_{vu}^{(g)}(t_j) \mid \tau, \tau^{(k)}, \tau^{(g)}\} = e^{-\kappa_\mu(t_i-t_j)^2} + \sum_{r=1}^{R}\tau_r^{-2}e^{-2\kappa_{\bar{x}}(t_i-t_j)^2},$$

$$\text{cov}\{z_{vu}^{(k)}(t_i), z_{pq}^{(k)}(t_j) \mid \tau, \tau^{(k)}\} = \text{cov}\{z_{vu}^{(k)}(t_i), z_{pq}^{(g)}(t_j) \mid \tau, \tau^{(k)}, \tau^{(g)}\} = e^{-\kappa_\mu(t_i-t_j)^2},$$

for every pair of layers k and g with $k \neq g$, times $t_i \in \mathbb{T}$, $t_j \in \mathbb{T}$ and pairs of actors $[vu]$, $v > u$ and $[pq]$, $p > q$ with $p \neq v$ or $q \neq u$.

It is easy to show that a priori the log-odds of the edge probabilities have mean zero, whereas, based on the above results, their variances increase with the sum of the shared and layer-specific shrinkage parameters. Allowing a subset of the coordinates to be common to all layers, has the effect of introducing dependence between the log-odds of edge probabilities in different layers. The Gaussian process priors provide an efficient choice to incorporate dependence between edge probabilities in different times. The strength of this dependence is regulated by the time lag and the positive smoothing parameters κ_μ, $\kappa_{\bar{x}}$ and κ_x. The lower these quantities, the stronger the dependence is between coordinates in different times.

4. Posterior Computation and Prediction

Equations (1)–(2) along with the Gaussian process priors for the latent coordinates can be seen as a nonlinear Bayesian logistic regression with the parameters entering in a bilinear form. Although performing posterior computation in this setting is apparently a cumbersome task, leveraging the Pólya-gamma data augmentation for Bayesian logistic regression (Polson et al., 2013) and adapting derivations in Durante and Dunson (2014), it is possible to obtain a simple and tractable Gibbs sampler having conjugate full conditionals. In fact, the Pólya-gamma data augmentation allows recasting the problem from a logistic regression to a multiple linear regression having transformed Gaussian response data, while derivations in Durante and Dunson (2014) provide results to linearize the factorization in equation (2).

Joining the above procedures and exploiting equations (7)–(8), the updating of the latent coordinates processes at every step simply relies on standard Bayesian linear regression. Algorithm 1 in Appendix B provides derivations and guidelines for step-by-step implementation of our Gibbs sampler. We summarize below the main steps of the MCMC routine.

Step [1]: For each time $t_i = t_1,\ldots,t_n$, layer $k = 1,\ldots,K$ and pair of actors $[vu]$, $v = 2,\ldots,V$, $u = 1,\ldots,v-1$, sample the corresponding Pólya-gamma augmented data.

Step [2]: Update the baseline process $\mu = \{\mu(t_1),\ldots,\mu(t_n)\}^\intercal$ from its full conditional multivariate Gaussian distribution. This is obtained by recasting the logistic regression for μ in terms of a multiple linear regression having transformed Gaussian response.

Step [3]: Update the shared coordinates. In performing this step we block-sample in turn the coordinates trajectories $\bar{x}_{(v)} = \{\bar{x}_{v1}(t_1),\ldots,\bar{x}_{v1}(t_n),\ldots,\bar{x}_{vR}(t_1),\ldots,\bar{x}_{vR}(t_n)\}^\intercal$ of each actor $v = 1,\ldots,V$ conditionally on the others $\{\bar{x}_{(u)} : u \neq v\}$. This choice allows us to linearize equation (2), with $\bar{x}_{(v)}$ acting as a coefficients vector and $\{\bar{x}_{(u)} : u \neq v\}$ representing appropriately selected regressors. Leveraging the Pólya-gamma data augmentation also this step relies on a multiple linear regression with transformed Gaussian response data, providing Gaussian full conditionals for each $\bar{x}_{(v)}$, $v = 1,\ldots,V$.

Step [4]: Update the layer-specific coordinates. For each layer $k = 1,\ldots,K$, this step relies on the same strategy considered for the shared coordinates, providing again Gaussian full conditionals for each $x_{(v)}^{(k)} = \{x_{v1}^{(k)}(t_1),\ldots,x_{v1}^{(k)}(t_n),\ldots,x_{vH}^{(k)}(t_1),\ldots,x_{vH}^{(k)}(t_n)\}^\intercal$, $v = 1,\ldots,V$ and $k = 1,\ldots,K$. Moreover the updating can be performed separately for each layer, allowing this step to be efficiently implemented in parallel.

Step [5] and [6]: The updating of the gamma parameters characterizing the priors in equations (5)–(6) follows conjugate analysis, proving gamma full conditionals.

Step [7]: Update the dynamic multilayer edge probabilities simply by applying equation (2) to the samples of the baseline process, the shared and the layer-specific coordinates.

In performing posterior computation we set R and H at conservative upper bounds, allowing the multiplicative inverse gamma priors for the shrinkage parameters to delete redundant latent space dimensions not required to characterize the data. Hence, posterior inference is not substantially affected by the choice of R and H, unless these bounds are fixed at excessively low values compared to the complexity of the data analyzed. We additionally assess the goodness of these bounds via in-sample and out-of-sample predictive performance.

4.1 Edge Prediction

Edge prediction is an important topic in dynamic modeling of multilayer networks. For example some networks may have unobserved edges due to inability to monitor certain types of relationships or actors at specific times. Likewise, some layers may be available in time earlier than others, facilitating prediction of those yet missing. The availability of efficient procedures that are able to reconstruct partially unobserved connectivity structures or forecast future networks can have important consequences in many applications, such as destabilization of terrorists networks or epidemic prevention (e.g., Tan et al., 2016).

Our statistical model for dynamic multilayer networks in equations (1)–(2) facilitates the definition of simple procedures for formal edge prediction relying on the expectation $\mathrm{E}\{\mathcal{Y}_{vu}^{(k)}(t_i) \mid Y_{\text{obs}}\}$ of the posterior predictive distribution for $\mathcal{Y}_{vu}^{(k)}(t_i)$, with Y_{obs} denoting the observed data. In fact, under equations (1)–(2), this functional is easily available as

$$\mathrm{E}\{\mathcal{Y}_{vu}^{(k)}(t_i) \mid Y_{\text{obs}}\} = \mathrm{E}_{\pi_{vu}^{(k)}(t_i)}[\mathrm{E}_{\mathcal{Y}_{vu}^{(k)}(t_i)}\{\mathcal{Y}_{vu}^{(k)}(t_i) \mid \pi_{vu}^{(k)}(t_i)\} \mid Y_{\text{obs}}] = \mathrm{E}\{\pi_{vu}^{(k)}(t_i) \mid Y_{\text{obs}}\}, \quad (9)$$

for every time $t_i \in \mathbb{T}$, layer $k = 1, \ldots, K$ and actors $v = 2, \ldots, V$, $u = 1, \ldots, v-1$, where $\mathrm{E}\{\pi_{vu}^{(k)}(t_i) \mid Y_{\text{obs}}\}$ simply coincides with the posterior mean of $\pi_{vu}^{(k)}(t_i)$. Hence prediction requires the posterior distribution of the edge probabilities. These quantities are available also for unobserved edges by adding the following data augmentation step in Algorithm 1:

Step [8]: Impute the missing edges given the current state of $\pi_{vu}^{(k)}(t_i)$ from a $\text{Bern}\{\pi_{vu}^{(k)}(t_i)\}$, for all the combinations of times $t_i \in \mathbb{T}$, layers $k = 1, \ldots, K$ and actors $v = 2, \ldots, V$, $u = 1, \ldots, v-1$ corresponding to unobserved edges.

5. Simulation Study

We consider a simulation study to evaluate the performance of our methodology in a simple scenario partially mimicking the size and structure of the face-to-face dynamic multilayer networks. Consistent with our aim we assess performance in the tasks outlined in Table 2.

To highlight the benefits of our methods (Joint Analysis), we compare the performance with two competitors. The first (Collapsed Analysis) assumes the layers arise from a common latent space model, forcing all the actors coordinates to be shared among layers. This formulation is a special case of our model holding out in (2) the layer-specific similarities.

Quantity	Focus of the performance assessment
Edge probabilities	Assess accuracy in learning the true edge probability process underlying the simulated network data. As already discussed, accurate modeling of the edge probabilities $\pi_{vu}^{(k)}(t_i)$ is fundamental for inference and prediction.
Unobserved edges	Assess out-of-sample predictive performance in forecasting future unobserved networks leveraging the expectation $\mathrm{E}\{\mathcal{Y}_{vu}^{(k)}(t_i) \mid Y_{\text{obs}}\}$ of the posterior predictive distribution for the unobserved edges.
Network density	Accuracy in modeling the expected overall chance of connectivity $\theta^{(k)}(t_i) = \mathrm{E}[\sum_{v=2}^{V}\sum_{u=1}^{v-1}\mathcal{Y}_{vu}^{(k)}(t_i)/\{V(V-1)/2\}] = \sum_{v=2}^{V}\sum_{u=1}^{v-1}\pi_{vu}^{(k)}(t_i)/\{V(V-1)/2\}$, for every time $t_i \in \mathbb{T}$ and layer $k = 1, \ldots, K$.
Actors degrees	Accuracy in learning the total number of different individuals $\mathrm{d}_v^{(k)}(t_i) = \mathrm{E}\{\sum_{u \neq v}\mathcal{Y}_{vu}^{(k)}(t_i)\} = \sum_{u \neq v}\pi_{vu}^{(k)}(t_i)$, each actor v is expected to interact with, for every $v = 1, \ldots, V$, time $t_i \in \mathbb{T}$ and layer $k = 1, \ldots, K$.

Table 2: Quantities of interest, which are considered in performance assessments. Although it is possible to provide inference on several properties, from an epidemiological perspective, the network density and actors degrees are of interest in characterizing the overall risk of contagion and the infectivity of each actor, respectively.

Hence, posterior analysis requires minor modifications of Algorithm 1. The second competitor (Separate Analysis) estimates the dynamic latent space model of Durante and Dunson (2014) separately for each layer. Although both methods are reasonable choices, modeling only shared structures lacks flexibility in capturing layer-specific patterns, whereas separate analyses lose efficiency in inference and prediction when the layers share common patterns. We describe the simulation setting in Section 5.1 and discuss results in Section 5.2.

5.1 Outline of the Simulation Settings

We consider $V = 30$ actors whose connections are monitored for $K = 2$ days on the same time grid $t_1, \ldots, t_{17}$, with $t_i \in \mathbb{T} = \{1, \ldots, 17\}$. In simulating the networks, we sample the edges from conditionally independent Bernoulli variables given their corresponding edge probabilities as in equation (1), with these probabilities defined to mimic possible scenarios in our application. To accomplish this goal, we consider, for each day, five representative edge probability matrices $\pi^{0(k)}(t_i)$ at times $t_i \in \{1, 5, 9, 13, 17\}$, displayed in Figure 2, and define the matrices at the remaining times as a convex combination of these representative matrices. Focusing, for example, on the times t_{14}, t_{15} and t_{16} between t_{13} and t_{17}, we let

$$\begin{aligned}
\pi^{0(k)}(t_{14}) &= 0.75\pi^{0(k)}(t_{13}) + 0.25\pi^{0(k)}(t_{17}),\\
\pi^{0(k)}(t_{15}) &= 0.50\pi^{0(k)}(t_{13}) + 0.50\pi^{0(k)}(t_{17}),\\
\pi^{0(k)}(t_{16}) &= 0.25\pi^{0(k)}(t_{13}) + 0.75\pi^{0(k)}(t_{17}).
\end{aligned}$$

The same construction holds for all the other time windows, except for times t_4 and t_6, where we induce a more rapid variation by letting $\pi^{0(k)}(t_4) = \pi^{0(k)}(t_6) = \pi^{0(k)}(t_5)$, instead

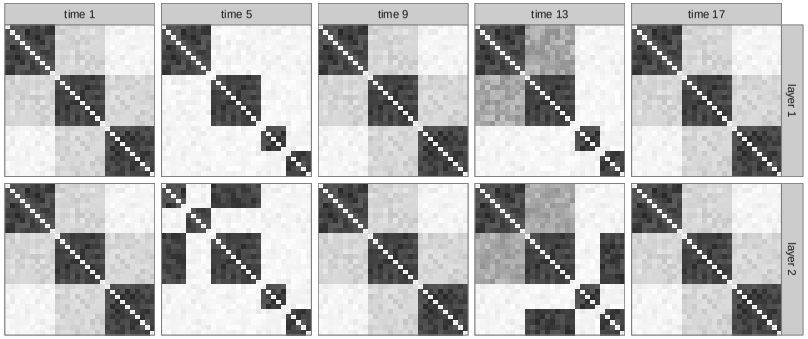


Figure 2: True representative edge probability matrices $\pi^{0(k)}(t_i)$. The color in each cell goes from white to black as the corresponding edge probability goes from 0 to 1.

of $\pi^{0(k)}(t_4) = 0.25\pi^{0(k)}(t_1) + 0.75\pi^{0(k)}(t_5)$, and $\pi^{0(k)}(t_6) = 0.75\pi^{0(k)}(t_5) + 0.25\pi^{0(k)}(t_9)$. We incorporate this setting to study the effects of the Gaussian process' constant smoothness assumption in scenarios having time-varying smoothness. In order to empirically evaluate Proposition 1, the edge probability matrices in Figure 2 are not generated from equation (2), but are instead constructed to characterize possible face-to-face contact scenarios.

Focusing on the first day, which corresponds to layer $k = 1$, the edge probability matrices at times t_1, t_9 and t_{17} characterize contacts during breakfast, lunch and dinner, respectively. These favor block-structures due to homophily by age and gender. In fact, it is reasonable to expect that young individuals, adult women and adult men, corresponding to actors in the first, second, and third block, may have a higher chance of contact with individuals in the same socio-demographic group. We additionally allow adult women to have a moderate chance of contact with young individuals and adult men during breakfast, lunch, and dinner times. The edge probability matrix at time t_5 characterizes instead block-structures due environmental restrictions during the morning, with young individuals attending school, adult women sharing the house, and two sub-groups of adult men working in two different places. Finally, the edge probability matrix at time t_{13} is similar to the one at time t_5 with exception of an increased chance of contact between young individuals and adult women who are expected to share the same environment in the afternoon after school.

The dynamic contact networks in the different days share similar patterns, as shown in Figure 1. We maintain this property by considering the same representative edge probability matrices in the second day, which corresponds to layer $k = 2$, with exception of t_5 and t_{13}. In the morning of the second day, we assume five young individuals contract a disease and remain at home. This increases their chance of contact with the adult women taking care of them, and reduces the probability of a contact with the other young individuals. The edge probability matrix in the afternoon of the second day characterizes a similar scenario, but focuses on five adult men. Including different edge probability matrices at times t_5 and t_{13}, allows us to assess performance in learning layer-to-layer differences in contact patterns.

	Squared Posterior Bias			Posterior Variance			Posterior Concentration		
		$n\downarrow$	$V\downarrow$		$n\downarrow$	$V\downarrow$		$n\downarrow$	$V\downarrow$
Joint	**0.009**	0.014	0.012	**0.011**	0.018	0.016	**0.020**	0.032	0.028
Collapsed	**0.016**	0.019	0.019	**0.007**	0.011	0.010	**0.023**	0.030	0.029
Separate	**0.016**	0.025	0.019	**0.018**	0.027	0.023	**0.034**	0.052	0.042

Table 3: For our methodology and the two competitors, measures of posterior concentration around the true dynamic multilayer edge probability process, at varying n and V. Bold numbers are the measures associated with our initial simulation scenario.

In order to assess predictive performance, we perform posterior analysis under our model and the competing methods, holding out from the observed data the networks from time t_{13} to t_{17} in the second day. This choice provides a scenario of interest in our application. For example a subset of actors may contract a disease at time t_{12} in the second day, motivating the design of prevention policies relying on the forecasted contact patterns at future times.

5.2 Results in the Simulation

We perform posterior computation under our model with $\kappa_\mu = \kappa_{\bar{x}} = \kappa_x = 0.05$ to favor smooth trajectories a priori and $a_1 = 2$, $a_2 = 2.5$ to facilitate adaptation of the latent spaces dimensions. Although our model can be easily modified to learn these hyperparameters from the data as in Murray and Adams (2010) and Bhattacharya and Dunson (2011), respectively, borrowing of information within the dynamic multilayer networks has the effect of reducing sensitivity to the choice of the hyperparameters. In fact, we found results robust to moderate changes in these quantities, and therefore prefer to elicit them to improve convergence and mixing. We consider 5000 Gibbs iterations with $R = H = 5$, and set a burn-in of 1000. Traceplots for the time-varying edge probabilities in the two layers show no evidence against convergence, and mixing is good in our experience with most of the effective sample sizes for the quantities of interest being around 1500 out of 4000 samples. Posterior computation for our competitors is performed with the same smoothing parameters and $a_1 = 2$, $a_2 = 2.5$, using $R + H = 10$ dimensional latent spaces, to improve comparison with our results.

5.2.1 Performance in Learning the True Edge Probability Process

Flexible modeling of $\pi^{(k)}(t)$ is fundamental for learning and prediction. Table 3 summarizes the concentration of the posterior distributions for the dynamic multilayer edge probabilities using the posterior mean of the squared difference between $\pi^{(k)}_{vu}(t_i) \mid Y_{\text{obs}}$ and $\pi^{0(k)}_{vu}(t_i)$. To provide empirical insights on posterior consistency we study this property also at varying n and V. In the first case we focus on the dynamic multilayer network studied at odd times $t_i \in \{1, 3, 5, \ldots, 17\}$, whereas in the second case we consider the reduced networks where four actors within each block are not analyzed. To highlight the contribution of the squared posterior bias and the posterior variance, we factorize the concentration measure as

$$\text{E}([\{\pi^{(k)}_{vu}(t_i) \mid Y_{\text{obs}}\} - \pi^{0(k)}_{vu}(t_i)]^2) = [\text{E}\{\pi^{(k)}_{vu}(t_i) \mid Y_{\text{obs}}\} - \pi^{0(k)}_{vu}(t_i)]^2 + \text{var}\{\pi^{(k)}_{vu}(t_i) \mid Y_{\text{obs}}\}.$$

To assess the overall performance, these measures are averaged across the edge probabilities

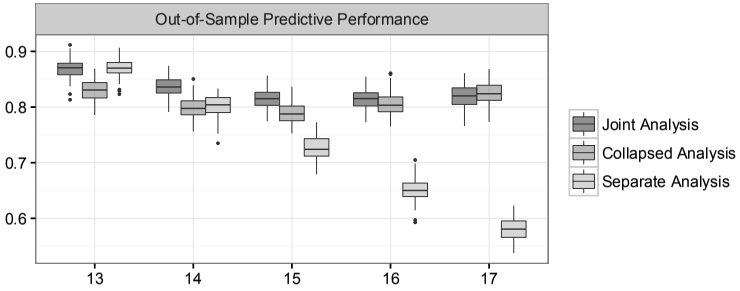


Figure 3: For times from t_{13} to t_{17} —held out in the second day from posterior computation —boxplots of the areas under the ROC curves (AUC) when predicting the unobserved edges with our method and the two competitors, from 100 simulations.

characterizing the dynamic multilayer network process.

According to Table 3, borrowing of information across layers has the effect of reducing both bias and posterior variance compared to the separate analyses, improving accuracy and efficiency in inference and prediction. Enhancing flexibility via layer-specific latent coordinates additionally facilitates bias reduction compared to collapsed analyses, but provides posterior distributions with slightly higher variance. This result is not surprising provided that the collapsed analyses assume the edge probability process to be common across layers, reducing the number of parameters. Moreover, although low variance is a desired feature when bias is small, this property may lead to poor inference and prediction when the posterior distribution is centered far from the truth. In our simulation—having few differences between the true edge probability processes in the two layers—the bias is low also for the collapsed analyses, but this bias is expected to grow when the layers have more differences.

The above results are maintained at varying n and V, while showing improved concentration for higher n or V. Although it would be interesting to prove posterior consistency, state-of-the-art literature provide results for the single binary outcome case (e.g., Ghosal and Roy, 2006), and it is not clear how to adapt this theory to our much complex network–valued data. However, results in Table 3 provide positive empirical support for this property.

5.2.2 Out-of-sample Predictive Performance

As discussed in Sections 2 and 4, developing models characterized by lower bias is important to improve inference and prediction. In fact, our method has an improved ability to predict unobserved edges compared to the competing models, as shown in Figure 3.[1] In predicting held-out edges we leverage the procedures discussed in Section 4.1. Since equation (1) is also valid for our competitors, results in Section 4.1 can be used for the collapsed and separate

1. In assessing predictive performance in Figure 3, we generate for each time from t_{13} to t_{17}—held out in the second day from posterior computation—100 networks from (1) with edge probabilities $\pi_{vu}^{0(2)}(t_i)$, and compute for each network the area under the ROC curve when predicting its edges with the expectation of the posterior predictive distribution associated with each method—according to Section 4.1.

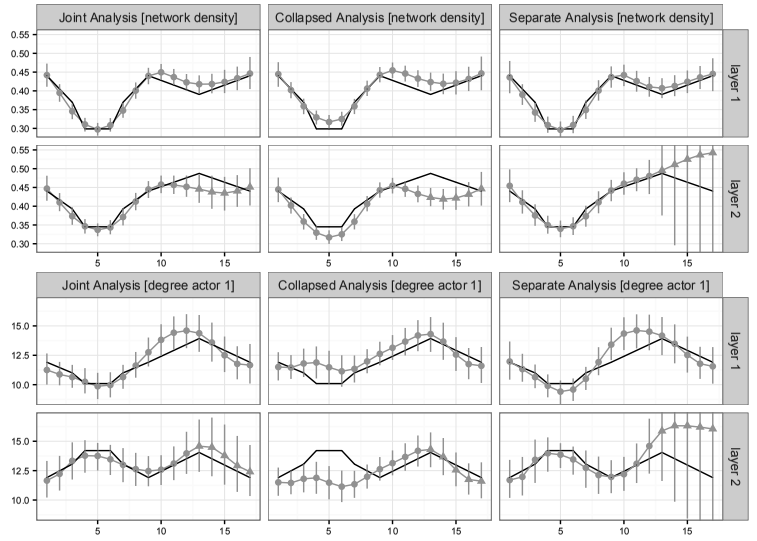


Figure 4: For the three methods, posterior mean (grey lines) and pointwise 0.95 credibility intervals (grey segments) for the dynamic expected network density (upper panels) and the dynamic expected degree of the first actor (lower panels) in the two layers. Black lines are the true trajectories, and triangles the held-out networks.

analyses after replacing $\mathrm{E}\{\pi_{vu}^{(k)}(t_i) \mid Y_{\mathrm{obs}}\}$ with $\mathrm{E}\{\pi_{vu}(t_i) \mid Y_{\mathrm{obs}}\}$ and $\mathrm{E}\{\pi_{vu}^{(k)}(t_i) \mid Y_{\mathrm{obs}}^{(k)}\}$, respectively.

According to Figure 3 the separate analysis obtains good predictive performance for the times subsequent to the last observed network, but this accuracy decays with increasing time lag. As the underlying edge probability process is smooth, it is reasonable to expect good predictive performance at time t_{13} exploiting information at time t_{12}, whereas the reduced accuracy for later times is likely due to inefficient exploitation of the information from the first layer. As expected, the collapsed analysis provides accurate predictions when there is a high degree of similarity between the edge probabilities in the two layers, such as at time t_{17}, but the accuracy decreases in correspondence of more evident differences. Differently from the above procedures, our model incorporates dependence between layers without assuming a single edge probability process common to all of them, providing predictions which are less sensitive to the underlying data generating mechanism. In fact, as shown in Figure 3, we obtain an improved prediction of unobserved edges compared to the competing methods.

5.2.3 Performance in Learning Network Properties of Interest

Accurate modeling of the dynamic multilayer edge probabilities has key benefits in improving inference on time-varying network structures in the different layers. Figure 4 summarizes

the posterior distribution of the dynamic expected network density, and the time-varying expected degree of a selected actor in the two layers.

As shown in Figure 4, our method outperforms the competing approaches also in providing inference for the selected network properties. The collapsed analysis forces the trajectories of these networks summary measures to be equal across layers, causing poor performance in capturing differences among layers in specific times. Modeling the dynamic edge probability processes underlying each layer separately improves flexibility, but the inefficient exploitation of the information shared between layers leads to poor performance in modeling network properties associated with unobserved edges. Our formulation in equations (1)–(2) incorporates dependence between layers and preserves flexibility via a set of layer-specific latent coordinates, which reduces in-sample and out-of-sample bias. Although our model has accurate performance, the constant smoothness assumption characterizing the Gaussian process priors leads to a slight over-smoothing around the more rapid variation in times t_4 and t_6. One possibility to avoid this behavior is to replace the Gaussian process priors with nested Gaussian processes (Zhu and Dunson, 2013) having time-varying smoothness.

6. Application to Dynamic Multilayer Face-to-Face Contact Data

We now turn to applying the methods proposed in Sections 2 and 4 to the face-to-face contact networks described in Section 1.2. Raw contact data are available for 75 individuals belonging to five different households in rural Kenya. Each household has different families living in the same compound and reporting to one head. Face-to-face contacts for the individuals in each household are collected hourly from 06:00 to 20:00 during three consecutive days, with these three days windows differing for most of the households. Hence, contact data are unavailable for almost all the pairs corresponding to actors in different households, making it impossible for any methodology to infer across-households connectivity structures. In order to avoid complications due to non-overlapping days, we focus on the face-to-face contact networks for the individuals in the most populated household comprising $V = 29$ actors. This choice is further supported by the fact that within households contacts play a fundamental role in the speed of contagion, motivating primary interests in household-based analyses; see e.g., House and Keeling (2009) and the references cited therein.

6.1 Results in the Application

The face-to-face proximity data can be structured as a dynamic multilayer network recording for each day $k \in \{1, 2, 3\}$ the time-varying face-to-face contacts among the $V = 29$ actors on the same hourly basis $t_1, \ldots, t_{14}$, where $t_i \in \{1, \ldots, 14\}$ is simply a discrete time index denoting the current hour. We apply our statistical model and the competing methods to the aforementioned data, where we hold out the contact network at the last time in the third day $Y^{(3)}_{t_{14}}$ to assess out-of-sample predictive performance.[2]

2. Posterior computation under the three models uses the same default hyperparameters as in the simulation study. Similarly to the simulation study, we consider 5000 Gibbs iterations and obtain no evidence against convergence after a burn-in of 1000. The effective sample sizes for the quantities of interest for inference are around 1200 out of 4000 samples. Although this provides a slightly reduced result compared to the simulation studies, the mixing is still good given the complexity of our dynamic multilayer network data.

		Area under the curve			Accuracy with cutoff 0.5		
Number of latent coordinates		8	10	12	8	10	12
Joint	In–sample	0.97	0.97	0.97	95%	95%	94%
	Out–of–sample	0.95	0.95	0.95	94%	94%	95%
Collapsed	In–sample	0.94	0.94	0.95	92%	92%	92%
	Out–of–sample	0.95	0.95	0.95	94%	94%	94%
Separate	In–sample	0.97	0.97	0.97	94%	94%	94%
	Out–of–sample	0.82	0.83	0.82	89%	89%	89%

Table 4: For our method and the two competitors, in-sample and out-of-sample predictive performance measured via the AUC, and the percentage of correctly predicted edges—using a cutoff probability of 0.5—at varying $R + H$, with $R = H$.

6.1.1 In-sample and Out-of-sample Predictive Performance

As one assessment of our proposed methodology, Table 4 compares our performance for in-sample and out-of-sample edge prediction with those provided by the competing procedures. Consistent with the strategy outlined in equation (9), the ROC curves for in-sample edge prediction are constructed using the observed data Y_{obs} and the posterior mean of their corresponding edge probabilities, estimated under the three different methods. The ROC curves for assessing out-of-sample predictive performance are defined in a similar way, but focusing on the held-out network at time t_{14} in day 3. To evaluate sensitivity to the choice of the upper bounds for the latent space dimensions, we study predictive performance also under other settings of R and H, including $R = H = 4$ and $R = H = 6$.

Edge prediction is clearly more accurate in-sample than out-of-sample. This is because Y_{obs} is part of the posterior computation, whereas the network at time t_{14} in day 3 is held out. According to Table 4, the separate analysis achieves comparable results to our model for in-sample edge prediction in accommodating layer-specific patterns, but provides poor predictions of future edges due to inability to incorporate information from the previous days. The collapsed analysis allows instead borrowing of information between the three days via shared coordinates, but does not incorporate differences in contact patters across these days. As a result, the collapsed analysis provides comparable performance to our model for out-of-sample edge prediction, but has a reduced in-sample predictive ability. Our dynamic multilayer representation efficiently incorporates both shared structures and layer-specific differences, facilitating good predictive performance both in-sample and out-of-sample. As discussed in Section 4, these results are not substantially affected by moderate variations in R and H, and therefore we perform posterior inference for $R = H = 5$, as in the simulation.

6.1.2 Learning Network Properties of Interest

The above results motivate additional posterior analyses of the dynamic multilayer stochastic process estimated under our model. Figure 5 summarizes the posterior distribution for the trajectory of the expected network density in the three days. The expected frequency

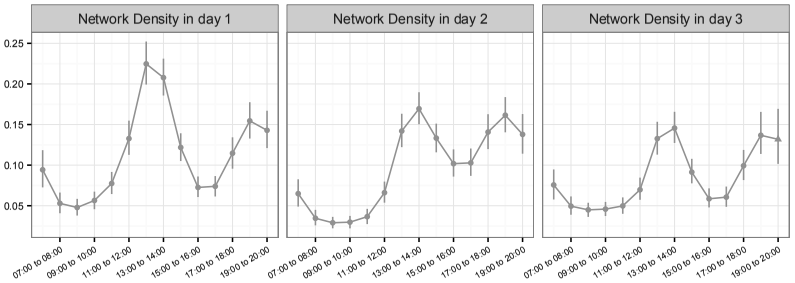


Figure 5: Posterior mean (grey lines) and pointwise 0.95 credibility intervals (grey segments) for the time-varying expected network density in the three days. The triangle for the last time in the third day denotes the held-out contact network.

of proximity contacts evolves on similar patterns during the three days, while showing some day-specific variations at certain hours. This result further confirms the importance of analyzing such data with statistical models that are able to borrow information across days, while maintaining flexibility in capturing day-specific patterns. Consistent with Kiti et al. (2016), the expected network density remains in general on low levels during the morning and afternoon when the actors are in different environments such as school and workplace, and peaks in correspondence of lunch and dinner times when the households members congregate. We learn a similar pattern when considering the dynamic expected frequencies of contact within and between groups of actors having similar traits, such as gender and age.[3] These trajectories provide key additional information compared to the expected network density, highlighting how infectious diseases can spread within and between groups of actors. The posterior mean of these trajectories is visualized in Figure 6.

According to Figure 6, all the trajectories inherit the pattern discussed for the expected network density, with clear peaks during lunch and dinner times. Although there are not evident differences in the dynamic expected frequencies of face-to-face contact within and between gender groups, we notice how contacts among males are typically less likely than contacts between females, whereas face-to-face interactions among actors of different genders become notable during lunch. Therefore, these hours might be at risk of spreading diseases from a gender group to the other. Consistent with the static analyses considered in Kiti et al. (2016), we observe more differences in the trajectories by age groups, with a substantially high chance of contact between young actors, whereas the dynamic expected frequency of interaction among adults remains on low levels. In fact, young individuals potentially have less restrictions from environment or work schedule than adults, and therefore have more chances of interaction, especially with actors of their same age. Hence, diseases may spread more easily among young individuals and reach adults during lunch times when the chance of contact between the two age groups is higher.

3. We can easily derive the posterior mean of these trajectories by averaging, for each time t_i and day k, all the estimated edge probabilities corresponding to pairs of actors having the same combination of traits, e.g., both males, both females, one male and one female.

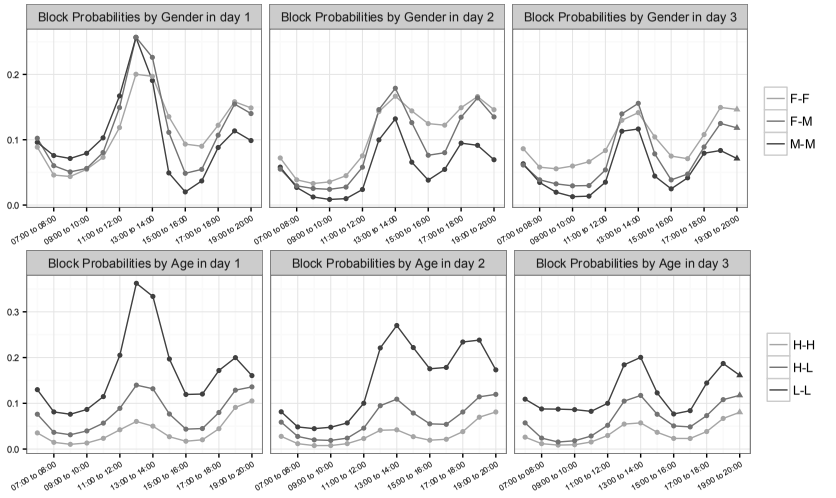


Figure 6: Upper panels: posterior mean of the dynamic expected frequencies of contact for actors having same gender—both males or both females—and different gender, in each day. Lower panels: same quantities with respect to high (≥ 15) and low (<15) age groups.

Figures 5 and 6 provide global measures for the dynamic risk of contagion in the network and for groups of actors. In order to develop more refined prevention policies it is of key interest to monitor the infectivity of each actor, and how this measure evolves with time and days. Consistent with this goal, Figure 7 provides a graphical analysis of selected contact networks, with the actors positions depending on the estimated edge probabilities averaged, for each day, over the two time windows having the more dense networks which may lead to a faster disease spread. As shown in Figure 7, the contact patterns in the same time window share a similar configuration across the three days, while displaying some day-to-day difference in the connectivity behavior of subsets of actors. For example, the community structure formed by actors 6, 17 and 28 during lunch times is less separated from the rest of the network in the first day, compared to the second and third day. This result provides additional support for our methodology which can simultaneously incorporate shared patterns and possible day-specific deviations from these common structures. Consistent with Figures 5 and 6, young individuals—represented by circles—are typically characterized by more frequent contacts, whereas the adult actors—corresponding to squares—have a more peripheral position in the network. As a result the degree of the young individuals is typically higher, making these actors particularly relevant in the economy of disease contagion and transmission. In this respect, actors 9 and 15 may represent dangerous hubs.

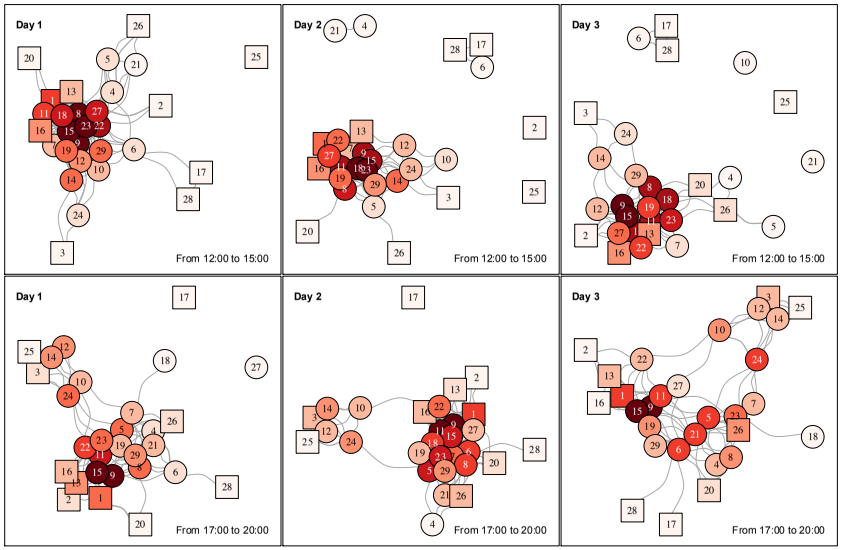


Figure 7: Weighted network visualization with weights obtained by averaging the posterior mean of the edge probabilities in the two time windows having the more dense networks, within each of the three days. Squares and circles represent the actors having high (≥ 15) and low (< 15) age, respectively. The color of each node goes from white to dark red as the estimated expected degree of the corresponding actor increases, relative to the others. Actors positions are obtained by applying the Fruchterman and Reingold (1991) force–directed placement algorithm, whereas the width of each edge is proportional to its averaged edge probability.

We can additionally notice some community structure within the network, particularly during dinner times. For example, actors 3, 10, 12, 14, 24 and 25 are tightly connected, but separated from the others, with exception of the young subjects 10 and 24 which may play a key role in the transmission of a disease from their community to the rest of the network. Other two communities can be more clearly observed during dinner times in day three. It is interesting to notice how these groups of actors comprise both young and adult individuals, and therefore may represent different families within the household under analysis.

7. Discussion

The increasing availability of multidimensional, complex, and dynamic information on social interaction processes motivates a growing demand for novel statistical models. In order to successfully enhance quality in inference and prediction, these models need to efficiently incorporate the complex set of dependencies in the observed data, without affecting flexibility. Motivated by this consideration and by epidemiological studies monitoring disease transmission via dynamic face-to-face interactions, we have developed a Bayesian nonparametric model for dynamic multilayer networks leveraging latent space representations. In order to preserve flexibility and borrow information across layers and time, we modeled the edge probabilities as a function of shared and layer-specific latent coordinates which evolve in time via Gaussian process priors. We provided theoretical support for our model, and developed simple procedures for posterior computation and formal prediction. Finally, we illustrated on both simulated data and on infection studies monitoring face-to-face contacts that our methods perform better than competitors in terms of inference and prediction.

Although we focus on face-to-face interaction networks collected at multiple times and days, our methods have a broad range of applications. Notable examples include dynamic cooperations among countries with respect to different types of international relations, time-varying interactions between researchers according to multiple forms of academic collaborations, and dynamic contacts between terrorists in relation to different types of dark interactions. In all these relevant applications, our flexible methodology can provide an appealing direction in accurately learning and predicting hidden wiring mechanisms and their implication in several environments and phenomena.

In addition, our methods motivate further directions of research. An important one is to facilitate scaling to larger dynamic multilayer network data. Currently, the computational complexity of our Gibbs sampler—as a function of the dimensions of the input data—is of order $O(VKn^3)$, and corresponds to the most intensive step updating the layer-specific latent coordinates trajectories for the V actors. Although the latent space representation reduces computational complexity from quadratic in the number of actors V to linear, the cubic complexity in the number of time points n associated with the Gaussian process prior, may still represent a computational barrier when data are monitored for wide time windows. A possible strategy to successfully address this issue is to consider more scalable processes such as the low-rank approximations to the Gaussian process (Banerjee et al., 2013) or state-space models. Another important generalization is accommodating the more informative contact counts instead of just a binary variable indicating presence or absence of face-to-face interactions. In accomplishing this goal one possibility is to adapt the methodology proposed in Canale and Dunson (2011) to our framework and assume the weighted edges are realizations from a rounded Gaussian whose mean is factorized as in equation (2). Finally, additional studies on the theoretical properties of these stochastic processes, such as those recently considered in Crane (2017), are of relevant interest.

Acknowledgments

We are grateful to the Action Editor Amos Storkey and the referee for the useful comments on a first version of the manuscript. Daniele Durante would like to acknowledge support for this project from the University of Padova, Italy (grant CPDA154381/15). Rebecca Steorts was supported in whole or in part with funding from the Laboratory for Analytic Sciences (LAS). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the LAS and/or any agency or entity of the United States Government.

Appendix A. Proofs of Propositions 1 and 2

Proof of Proposition 1. As the logistic mapping is one-to-one continuous, Proposition 1 is valid if and only if—for each time $t \in \mathbb{T}$—every possible collection of log-odds $z(t) = \{z_{vu}^{(k)}(t) \in \Re : k = 1, \ldots, K,\ v = 2, \ldots, V,\ u = 1, \ldots, v-1\}$ can be factorized as

$$z_{vu}^{(k)}(t) = \mu(t) + \bar{s}_{vu}(t) + s_{vu}^{(k)}(t) = \mu(t) + \bar{x}_v(t)^\intercal \bar{x}_u(t) + x_v^{(k)}(t)^\intercal x_u^{(k)}(t),$$

for all $k = 1, \ldots, K,\ v = 2, \ldots, V,\ u = 1, \ldots, v-1$, where $x_v^{(k)}(t) = \{x_{v1}^{(k)}(t), \ldots, x_{vH}^{(k)}(t)\}^\intercal$ and $\bar{x}_v(t) = \{\bar{x}_{v1}(t), \ldots, \bar{x}_{vR}(t)\}^\intercal$, for every $v = 1, \ldots, V$.

Assume without loss of generality $\mu(t) = 0$ for every $t \in \mathbb{T}$ and let $\bar{X}(t)$ and $X^{(k)}(t)$ for $k = 1, \ldots, K$, denote the $V \times R$ and $V \times H$ matrices containing the shared and layer-specific latent coordinates, respectively, at time $t \in \mathbb{T}$. Since we are not interested in the diagonal elements of the $V \times V$ edge probabilities and log-odds matrices, it is always possible to write

$$Z^{(k)}(t) = \bar{S}(t) + S^{(k)}(t), \quad k = 1, \ldots, K,$$

where $Z^{(k)}(t)$ is the $V \times V$ symmetric matrix having the log-odds of the edges at time t in layer k as off-diagonal elements, whereas $\bar{S}(t)$ and $S^{(k)}(t)$ are $V \times V$ positive semidefinite symmetric matrices having quantities $\bar{s}_{vu}(t),\ v = 2, \ldots, V,\ u = 1, \ldots, v-1$ and $s_{vu}^{(k)}(t)$, $v = 2, \ldots, V,\ u = 1, \ldots, v-1$ as off-diagonal elements. Since we are not interested in self-relations, there is no loss of generality in assuming $\bar{S}(t)$ and $S^{(k)}(t)$ are positive semidefinite, since for any configuration of shared and layer-specific similarities there exist infinitely many positive semidefinite matrices having these quantities as off-diagonal elements.

Since $\bar{S}(t)$ and $S^{(k)}(t)$ are positive semidefinite, they admit the eigendecompositions $\bar{S}(t) = \bar{U}(t)\bar{\Lambda}(t)\bar{U}(t)^\intercal$ and $S^{(k)}(t) = U^{(k)}(t)\Lambda^{(k)}(t)U^{(k)}(t)^\intercal$, where $\bar{U}(t)$ and $U^{(k)}(t)$ denote the $V \times R_t$ and $V \times H_t^{(k)}$ matrices of eigenvectors, whereas $\bar{\Lambda}(t)$ and $\Lambda^{(k)}(t)$ are the corresponding diagonal matrices with the positive eigenvalues. Therefore, letting $R \geq R_t$ for every $t \in \mathbb{T}$ and $H \geq H_t^{(k)}$, for every $k = 1, \ldots, K$ and $t \in \mathbb{T}$, Proposition 1 follows after defining $\bar{X}(t)$ and $X^{(k)}(t)$ as the block matrices $\bar{X}(t) = \{\bar{U}(t)\bar{\Lambda}(t)^{1/2}, 0_{V \times (R - R_t)}\}$ and $X^{(k)}(t) = \{U^{(k)}(t)\Lambda^{(k)}(t)^{1/2}, 0_{V \times (H - H_t^{(k)})}\}$ for every $k = 1, \ldots, K$. ■

Proof of Proposition 2. Leveraging proof of Corollary 2 in Durante and Dunson (2014), to prove Proposition 2 it suffices to show that

$$\text{pr}\left(\sup_{t \in \mathbb{T}} \left[\sum_{k=1}^{K} \sqrt{\sum_{v=2}^{V}\sum_{u=1}^{v-1} \{z_{vu}^{(k)}(t) - z_{vu}^{0(k)}(t)\}^2}\right] < \epsilon\right) > 0,$$

where $z_{vu}^{(k)}(t)$ and $z_{vu}^{0(k)}(t)$ are the log-odds of $\pi_{vu}^{(k)}(t)$ and $\pi_{vu}^{0(k)}(t)$, respectively. Recalling the proof of Proposition 1, the above probability can be factorized as

$$\text{pr}\left(\sup_{t \in \mathbb{T}} \left[\sum_{k=1}^{K} \sqrt{\sum_{v=2}^{V}\sum_{u=1}^{v-1} \{\mu(t) + \bar{s}_{vu}(t) + s_{vu}^{(k)}(t) - \mu^0(t) - \bar{s}_{vu}^0(t) - s_{vu}^{0(k)}(t)\}^2}\right] < \epsilon\right),$$

with $\bar{s}_{vu}(t) = \bar{x}_v(t)^\intercal \bar{x}_u(t)$, $s_{vu}^{(k)}(t) = x_v^{(k)}(t)^\intercal x_u^{(k)}(t)$, $\bar{s}_{vu}^0(t) = \bar{x}_v^0(t)^\intercal \bar{x}_u^0(t)$ and $s_{vu}^{0(k)}(t) = x_v^{0(k)}(t)^\intercal x_u^{0(k)}(t)$. Exploiting the triangle inequality and the independence of the Gaussian process priors for the different trajectories, a lower bound for the above probability is

$$\text{pr}\left(\sup_{t \in \mathbb{T}} \left[\sqrt{\sum_{v=2}^{V}\sum_{u=1}^{v-1} \{\mu(t) - \mu^0(t) + \bar{s}_{vu}(t) - \bar{s}_{vu}^0(t)\}^2}\right] < \frac{\epsilon}{2K}\right) \times$$
$$\times \prod_{k=1}^{K} \text{pr}\left(\sup_{t \in \mathbb{T}} \left[\sqrt{\sum_{v=2}^{V}\sum_{u=1}^{v-1} \{s_{vu}^{(k)}(t) - s_{vu}^{0(k)}(t)\}^2}\right] < \frac{\epsilon}{2K}\right).$$

Applying Theorem 2 in Durante and Dunson (2014) to each term in this factorization, it is easy to show that all the above probabilities are strictly positive, proving Proposition 2. ■

Appendix B. Pseudocode for Posterior Computation

Algorithm 1 provides guidelines for step-by-step implementation of our Gibbs sampler.

Algorithm 1 Gibbs sampler for the dynamic multilayer latent space model

[1] Generate the Pólya-gamma augmented data
for each $t_i = t_1, \ldots, t_n$, $k = 1, \ldots K$, $v = 2, \ldots, V$ and $u = 1, \ldots, v-1$ **do**
Sample the augmented data $\omega_{vu}^{(k)}(t_i)$ from the full conditional Pólya-gamma

$$\omega_{vu}^{(k)}(t_i) \mid - \sim \text{PG}\left\{1, \mu(t_i) + \bar{x}_v(t_i)^\intercal \bar{x}_u(t_i) + x_v^{(k)}(t_i)^\intercal x_u^{(k)}(t_i)\right\},$$

where $\text{PG}(a, b)$ is the Pólya-gamma random variable with parameters $a > 0$ and $b \in \Re$.
end for

---

[2] Update the baseline trajectory $\mu = \{\mu(t_1), \ldots, \mu(t_n)\}^\intercal$ from $\mu \mid - \sim \text{N}_n(\mu_\mu, \Psi_\mu)$ where $\Psi_\mu = \left[\text{diag}\left\{\sum_{k=1}^{K}\sum_{v=2}^{V}\sum_{u=1}^{v-1} \omega_{vu}^{(k)}(t_1), \ldots, \sum_{k=1}^{K}\sum_{v=2}^{V}\sum_{u=1}^{v-1} \omega_{vu}^{(k)}(t_n)\right\} + \Sigma_\mu^{-1}\right]^{-1}$ and $\mu_\mu = \Psi_\mu \eta_\mu$ with

$$\eta_\mu = \begin{bmatrix} \sum_{k=1}^{K}\sum_{v=2}^{V}\sum_{u=1}^{v-1} \{Y_{t_1[vu]}^{(k)} - 1/2 - \omega_{vu}^{(k)}(t_1)[\bar{x}_v(t_1)^\intercal \bar{x}_u(t_1) + x_v^{(k)}(t_1)^\intercal x_u^{(k)}(t_1)]\} \\ \vdots \\ \sum_{k=1}^{K}\sum_{v=2}^{V}\sum_{u=1}^{v-1} \{Y_{t_n[vu]}^{(k)} - 1/2 - \omega_{vu}^{(k)}(t_n)[\bar{x}_v(t_n)^\intercal \bar{x}_u(t_n) + x_v^{(k)}(t_n)^\intercal x_u^{(k)}(t_n)]\} \end{bmatrix}.$$

---

[3] Sample the vectors of shared coordinates $\bar{x}_v(t_1), \ldots, \bar{x}_v(t_n)$ **for** $v = 1, \ldots, V$ **for** each actor $v = 1, \ldots, V$ **do**
Block-sample $\{\bar{x}_v(t_1), \ldots, \bar{x}_v(t_n)\}$ given the others $\{\bar{x}_u(t_i) : u \neq v, t_i = t_1, \ldots t_n\}$.

[a] Let $\bar{x}_{(v)} = \{\bar{x}_{v1}(t_1), \ldots, \bar{x}_{v1}(t_n), \ldots, \bar{x}_{vR}(t_1), \ldots, \bar{x}_{vR}(t_n)\}^\intercal$

[b] Define a Bayesian logistic regression with $\bar{x}_{(v)}$ acting as coefficient vector and having prior, according to equation (7), $\bar{x}_{(v)} \sim \text{N}_{n \times R}\left\{0, \text{diag}(\tau_1^{-1}, \ldots, \tau_R^{-1}) \otimes \Sigma_{\bar{x}}\right\}$

[c] For every $k = 1, \ldots, K$, the Bayesian logistic regression for the updating of $\bar{x}_{(v)}$ is

$$Y^{(k)}_{(v)} \sim \text{Bern}(\pi^{(k)}_{(v)}), \quad \text{logit}(\pi^{(k)}_{(v)}) = 1_{V-1} \otimes \mu + \bar{X}_{(-v)}\bar{x}_{(v)} + X^{(k)}_{(-v)}x^{(k)}_{(v)},$$

with

- $Y^{(k)}_{(v)}$ obtained by stacking vectors $\{Y^{(k)}_{t_1[vu]}, \ldots, Y^{(k)}_{t_n[vu]}\}^\intercal$ for all pairs having v as a one of the two actors
- $\pi^{(k)}_{(v)}$ the corresponding vector of edge probabilities
- $x^{(k)}_{(v)} = \{x^{(k)}_{v1}(t_1), \ldots, x^{(k)}_{v1}(t_n), \ldots, x^{(k)}_{vH}(t_1), \ldots, x^{(k)}_{vH}(t_n)\}^\intercal$
- $\bar{X}_{(-v)}$ and $X^{(k)}_{(-v)}$the matrices of regressors whose entries are suitably chosen from $\{\bar{x}_u(t_i) : u \neq v, t_i = t_1, \ldots t_n\}$ and $\{x^{(k)}_u(t_i) : u \neq v, t_i = t_1, \ldots t_n\}$

According to the above specification, and letting $\Omega^{(k)}_{(v)}$ denote the diagonal matrix with the corresponding Pólya-gamma augmented data, we obtain

$$\bar{x}_{(v)} \mid - \sim \text{N}_{n\times R}\left(\mu_{\bar{x}_{(v)}}, \Psi_{\bar{x}_{(v)}}\right),$$

with

$$\Psi_{\bar{x}_{(v)}} = \{\bar{X}^\intercal_{(-v)}(\Omega^{(1)}_{(v)} + \ldots + \Omega^{(K)}_{(v)})\bar{X}_{(-v)} + \text{diag}(\tau_1, \ldots, \tau_R) \otimes \Sigma^{-1}_{\bar{x}}\}^{-1},$$
$$\mu_{\bar{x}_{(v)}} = \Psi_{\bar{x}_{(v)}}(\bar{X}^\intercal_{(-v)}[\textstyle\sum_{k=1}^K\{Y^{(k)}_{(v)} - 1_{V-1} \otimes 1_n 0.5 - \Omega^{(k)}_{(v)}(1_{V-1} \otimes \mu + X^{(k)}_{(-v)}x^{(k)}_{(v)})\}]).$$

end for

---

[4] Sample the layer-specific coordinates $x^{(k)}_v(t_1), \ldots, x^{(k)}_v(t_n)$ for $v = 1, \ldots, V$ and $k = 1, \ldots, K$

for each layer $k = 1, \ldots, K$ **do**
  for each actor $v = 1, \ldots, V$ **do**
    Block-sample $\{x^{(k)}_v(t_1), \ldots, x^{(k)}_v(t_n)\}$ given $\{x^{(k)}_u(t_i) : u \neq v, t_i = t_1, \ldots t_n\}$. In particular, letting $x^{(k)}_{(v)} = \{x^{(k)}_{v1}(t_1), \ldots, x^{(k)}_{v1}(t_n), \ldots, x^{(k)}_{vH}(t_1), \ldots, x^{(k)}_{vH}(t_n)\}^\intercal$ and adapting derivations in step [3] to the sampling of the layer-specific coordinates, we obtain

$$x^{(k)}_{(v)} \mid - \sim \text{N}_{n\times H}\left(\mu_{x^{(k)}_{(v)}}, \Psi_{x^{(k)}_{(v)}}\right),$$

with

$$\Psi_{x^{(k)}_{(v)}} = \{X^{(k)\intercal}_{(-v)}\Omega^{(k)}_{(v)}X^{(k)}_{(-v)} + \text{diag}(\tau^{(k)}_1, \ldots, \tau^{(k)}_H) \otimes \Sigma^{-1}_x\}^{-1},$$
$$\mu_{x^{(k)}_{(v)}} = \Psi_{x^{(k)}_{(v)}}[X^{(k)\intercal}_{(-v)}\{Y^{(k)}_{(v)} - 1_{V-1} \otimes 1_n 0.5 - \Omega^{(k)}_{(v)}(1_{V-1} \otimes \mu + \bar{X}_{(-v)}\bar{x}_{(v)})\}].$$

  end for
end for

---

[5] Update the gamma quantities defining the shared shrinkage parameters $\tau_1^{-1}, \ldots, \tau_R^{-1}$

$$\delta_1 \mid - \sim \text{Ga}\left\{a_1 + \frac{V \times n \times R}{2}, 1 + \frac{1}{2}\sum_{m=1}^R \theta^{(-1)}_m \sum_{v=1}^V \bar{x}^\intercal_{vm}\Sigma^{-1}_{\bar{x}}\bar{x}_{vm}\right\},$$
$$\delta_{r\geq 2} \mid - \sim \text{Ga}\left\{a_2 + \frac{V \times n \times (R - r + 1)}{2}, 1 + \frac{1}{2}\sum_{m=r}^R \theta^{(-r)}_m \sum_{v=1}^V \bar{x}^\intercal_{vm}\Sigma^{-1}_{\bar{x}}\bar{x}_{vm}\right\},$$

where $\theta^{(-r)}_m = \prod_{f=1, f\neq r}^m \delta_f$ for $r = 1, \ldots, R$ and $\bar{x}_{vm} = \{\bar{x}_{vm}(t_1), \ldots, \bar{x}_{vm}(t_n)\}^\intercal$.

---

[6] Sample the variables defining the layer-specific parameters $\tau_1^{(k)-1}, \ldots, \tau_H^{(k)-1}$, $k = 1, \ldots, K$

for each layer $k = 1, \ldots, K$ **do**

$$\delta^{(k)}_1 \mid - \sim \text{Ga}\left\{a_1 + \frac{V \times n \times H}{2}, 1 + \frac{1}{2}\sum_{l=1}^H \theta^{(-1)}_l \sum_{v=1}^V x^{(k)\intercal}_{vl}\Sigma^{-1}_x x^{(k)}_{vl}\right\},$$
$$\delta^{(k)}_{h\geq 2} \mid - \sim \text{Ga}\left\{a_2 + \frac{V \times n \times (H - h + 1)}{2}, 1 + \frac{1}{2}\sum_{l=h}^H \theta^{(-h)}_l \sum_{v=1}^V x^{(k)\intercal}_{vl}\Sigma^{-1}_x x^{(k)}_{vl}\right\},$$

where $\theta^{(-h)}_l = \prod_{f=1, f\neq h}^l \delta^{(k)}_f$ for $h = 1, \ldots, H$ and $x^{(k)}_{vl} = \{x^{(k)}_{vl}(t_1), \ldots, x^{(k)}_{vl}(t_n)\}^\intercal$.
end for

---

[7] Obtain the posterior samples for the edge probabilities $\pi^{(k)}_{vu}(t_i)$ via

$$\pi^{(k)}_{vu}(t_i) = [1 + \exp\{-\mu(t_i) - \bar{x}_v(t_i)^\intercal\bar{x}_u(t_i) - x^{(k)}_v(t_i)^\intercal x^{(k)}_u(t_i)\}]^{-1},$$

for every $t_i = t_1, \ldots, t_n$, layer $k = 1, \ldots, K$ and actors $v = 2, \ldots, V$, $u = 1, \ldots, v - 1$.

---

[8] Impute the missing edges given the current state of $\pi^{(k)}_{vu}(t_i)$ from a Bern$\{\pi^{(k)}_{vu}(t_i)\}$, for all the combinations of times $t_i \in \mathbb{T}$, layers $k = 1, \ldots, K$ and actors $v = 2, \ldots, V$, $u = 1, \ldots, v - 1$ corresponding to unobserved edges.

References

E.M. Airoldi, D.M. Blei, S.E. Fienberg, and E.P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9:1981–2014, 2008.

A. Banerjee, D.B. Dunson, and S.T. Tokdar. Efficient Gaussian process regression for large datasets. *Biometrika*, 100(1):75–89, 2013.

A. Bhattacharya and D.B. Dunson. Sparse Bayesian infinite factor models. *Biometrika*, 98 (2):291–306, 2011.

V.D. Blondel, A. Decuyper, and G. Krings. A survey of results on mobile phone datasets analysis. *EPJ Data Science*, 4(1):1, 2015.

A. Canale and D.B. Dunson. Bayesian kernel mixtures for counts. *Journal of the American Statistical Association*, 106(496):1528–1539, 2011.

C. Cattuto, W. Van den Broeck, A. Barrat, V. Colizza, J.F. Pinton, and A. Vespignani. Dynamics of person-to-person interactions from distributed RFID sensor networks. *PloS one*, 5(7):e11596, 2010.

S. Cauchemez, A. Bhattarai, T.L. Marchbanks, R.P. Fagan, S. Ostroff, N.M. Ferguson, D. Swerdlow, and the Pennsylvania H1N1 working group. Role of social networks in shaping disease transmission during a community outbreak of 2009 h1n1 pandemic influenza. *Proceedings of the National Academy of Sciences*, 108(7):2825–2830, 2011.

H. Crane. Combinatorial Levy processes. *arXiv preprint*, arXiv:1612.05746, 2017.

H. Crane. Dynamic random networks and their graph limits. *The Annals of Applied Probability*, 26(2):691–721, 2016.

D. Durante. A note on the multiplicative gamma process. *Statistics & Probability Letters*, 122:198–204, 2017.

D. Durante and D.B. Dunson. Nonparametric Bayes dynamic modelling of relational data. *Biometrika*, 101(4):883–898, 2014.

O. Frank and D. Strauss. Markov graphs. *Journal of the American Statistical Association*, 81(395):832–842, 1986.

T.M. Fruchterman and E.M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164, 1991.

S. Ghosal and A. Roy. Posterior consistency of Gaussian process prior for nonparametric binary regression. *The Annals of Statistics*, 34(5):2413–2429, 2006.

J. Ghosh and D.B. Dunson. Default prior distributions and efficient posterior computation in Bayesian factor analysis. *Journal of Computational and Graphical Statistics*, 18(2): 306–320, 2009.

I. Gollini and T.B. Murphy. Joint modeling of multiple network views. *Journal of Computational and Graphical Statistics*, 25(1):246–265, 2016.

F. Guo, C. Blundell, H. Wallach, and K. Heller. The Bayesian echo chamber: Modeling social influence via linguistic accommodation. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, pages 315–323. Journal of Machine Learning Research, Workshops & Proceedings, 2015.

Q. Han, K. Xu, and E.M. Airoldi. Consistent estimation of dynamic and multi-layer block models. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1511–1520. Journal of Machine Learning Research, Workshops & Proceedings, 2015.

S. Hanneke, W. Fu, and E.P. Xing. Discrete temporal models of social networks. *Electronic Journal of Statistics*, 4:585–605, 2010.

M.T. Heaney. Multiplex networks and interest group influence reputation: An exponential random graph model. *Social Networks*, 36(1):66–81, 2014.

P.D. Hoff. Bilinear mixed-effects models for dyadic data. *Journal of the American Statistical Association*, 100(469):286–295, 2005.

P.D. Hoff. Multilinear tensor regression for longitudinal relational data. *The Annals of Applied Statistics*, 9(3):1169–1193, 2015.

P.D. Hoff, A.E. Raftery, and M.S. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97(460):1090–1098, 2002.

P.W. Holland and S. Leinhardt. An exponential family of probability distributions for directed graphs. *Journal of the American Statistical Association*, 76(373):33–50, 1981.

T. House and M.J. Keeling. Household structure and infectious disease transmission. *Epidemiology and Infection*, 137(5):654–661, 2009.

M.C. Kiti, M. Tizzoni, T.M. Kinyanjui, D.C. Koech, P.K. Munywoki, M. Meriac, L. Cappa, A. Panisson, A. Barrat, C. Cattuto, and D.J. Nokes. Quantifying social contacts in a household setting of rural Kenya using wearable proximity sensors. *EPJ Data Science*, 5 (1):1, 2016.

S. Lee and P. Monge. The coevolution of multiplex communication networks in organizational communities. *Journal of Communication*, 61(4):758–779, 2011.

K. Leetaru and P.A. Schrodt. GDELT: Global data on events, location, and tone, 1979–2012. In *International Studies Association Annual Convention*, 2013.

M. Marathe and N. Ramakrishnan. Recent advances in computational epidemiology. *IEEE Intelligent Systems*, 28(4):96–101, 2013.

I. Murray and R.P. Adams. Slice sampling covariance hyperparameters of latent Gaussian models. In *Advances in Neural Information Processing Systems*, pages 1732–1740, 2010.

K. Nowicki and T.A.B. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001.

B. Oselio, A. Kulesza, and A.O. Hero. Multi-layer graph analysis for dynamic social networks. *IEEE Journal of Selected Topics in Signal Processing*, 8(4):514–523, 2014.

N.G. Polson, J.G. Scott, and J. Windle. Bayesian inference for logistic models using Pólya–Gamma latent variables. *Journal of the American Statistical Association*, 108(504):1339–1349, 2013.

C.E. Rasmussen and C.K. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

A. Schein, M. Zhou, D. Blei, and H. Wallach. Bayesian Poisson Tucker decomposition for learning the structure of international relations. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 2810–2819. Journal of Machine Learning Research, Workshops & Proceedings, 2016.

T.A.B. Snijders, G.G. van de Bunt, and C.E.G. Steglich. Introduction to stochastic actor-based models for network dynamics. *Social Networks*, 32(1):44–60, 2010.

T.A.B. Snijders, A. Lomi, and V.J. Torló. A model for the multiplex dynamics of two-mode and one-mode networks, with an application to employment preference, friendship, and advice. *Social Networks*, 35(2):265–276, 2013.

J. Stehlé, N. Voirin, A. Barrat, C. Cattuto, L. Isella, J.F. Pinton, M. Quaggiotto, W. van den Broeck, C. Régis, B. Lina, and P. Vanhems. High-resolution measurements of face-to-face contact patterns in a primary school. *PloS one*, 6(8):e23176, 2011.

S.Y. Tan, J. Wu, L. Lü, M.J. Li, and X. Lu. Efficient network disintegration under incomplete information: The comic effect of link prediction. *Scientific Reports*, 6:1–9, 2016.

P. Vanhems, A. Barrat, C. Cattuto, J.F. Pinton, N. Khanafer, C. Régis, B. Kim, B. Comte, and N. Voirin. Estimating potential infection transmission routes in hospital wards using wearable proximity sensors. *PloS one*, 8(9):e73970, 2013.

E.P. Xing, W. Fu, and L. Song. A state-space mixed membership blockmodel for dynamic network tomography. *The Annal of Applied Statistics*, 4(2):535–566, 2010.

T. Yang, Y. Chi, S. Zhu, Y. Gong, and R. Jin. Detecting communities and their evolutions in dynamic social networks—a Bayesian approach. *Machine Learning*, 82(2):157–189, 2011.

B. Zhu and D.B. Dunson. Locally adaptive Bayes nonparametric regression via nested Gaussian processes. *Journal of the American Statistical Association*, 108(504):1445–1456, 2013.

Journal of Machine Learning Research 18 (2017) 1-29 Submitted 10/16; Revised 3/17; Published 4/17

Time-Accuracy Tradeoffs in Kernel Prediction: Controlling Prediction Quality

Samory Kpotufe samory@princeton.edu
ORFE, Princeton University

Nakul Verma verman@janelia.hhmi.org
Janelia Research Campus, HHMI

Editor: Michael Mahoney

Abstract

Kernel regression or classification (also referred to as *weighted* ϵ-NN methods in Machine Learning) are appealing for their simplicity and therefore ubiquitous in data analysis. However, practical implementations of kernel regression or classification consist of quantizing or sub-sampling data for improving time efficiency, often at the cost of prediction quality. While such tradeoffs are necessary in practice, their statistical implications are generally not well understood, hence practical implementations come with few performance guarantees. In particular, it is unclear whether it is possible to maintain the statistical accuracy of kernel prediction—crucial in some applications—while improving prediction time.

The present work provides guiding principles for combining kernel prediction with data-quantization so as to guarantee good tradeoffs between prediction time and accuracy, and in particular so as to approximately maintain the good accuracy of vanilla kernel prediction.

Furthermore, our tradeoff guarantees are worked out explicitly in terms of a tuning parameter which acts as a *knob* that favors either time or accuracy depending on practical needs. On one end of the knob, prediction time is of the same order as that of *single*-nearest-neighbor prediction (which is statistically inconsistent) while maintaining consistency; on the other end of the knob, the prediction risk is nearly minimax-optimal (in terms of the original data size) while still reducing time complexity. The analysis thus reveals the interaction between the data-quantization approach and the kernel prediction method, and most importantly gives explicit control of the tradeoff to the practitioner rather than fixing the tradeoff in advance or leaving it opaque.

The theoretical results are validated on data from a range of real-world application domains; in particular we demonstrate that the theoretical *knob* performs as expected.

1. Introduction

Kernel regression or classification approaches—which predict at a point x by weighting the contributions of nearby sample points using a *kernel* function—are some of the most studied approaches in nonparametric prediction and enjoy optimality guarantees in general theoretical settings. In Machine Learning, these methods often appear as *weighted* ϵ-Nearest-Neighbor prediction, and are ubiquitous due to their simplicity.

However, vanilla kernel prediction (regression or classification) methods are rarely implemented in practice since they can be expensive at prediction time: given a large training data $\{X_i, Y_i\}$ of size n, predicting Y at a new query x can take time $\Theta(n)$ since much of the data $\{X_i\}$ has to be visited to compute kernel weights.

Practical implementations therefore generally combine fast-similarity search procedures with some form of data-quantization or sub-sampling (where the original large training data $\{X_i, Y_i\}$ is replaced with a smaller quantization set $\{q, Y_q\}$[1], see Figure 1) for faster processing (see e.g. Lee and Gray, 2008; Atkeson et al., 1997). However, the effect of these approximations on prediction accuracy is not well understood theoretically, hence good tradeoffs rely mostly on proper engineering and domain experience. A main motivation of this work is to elucidate which features of practical approaches to time-accuracy tradeoffs should be most emphasized in practice, especially in applications where it is crucial to approximately maintain the accuracy of the original predictor (e.g. medical analytics, structural-health monitoring, robotic control, where inaccurate prediction is relatively costly, yet fast prediction is desired).

The present work provides simple guiding principles that guarantee good tradeoffs between prediction time and accuracy under benign theoretical conditions that are easily met in real-world applications.

We first remark that fast-similarity search procedures alone can only guarantee sub-optimal time-improvement, replacing the linear order $O(n)$ with a root of n; this is easy to show (see Proposition 1). Therefore, proper data-quantization or sub-sampling is crucial to achieving good tradeoffs between prediction time and accuracy. In particular we first show that it is possible to guarantee prediction time of order $O(\log n)$ for structured data (data on a manifold or sparse data) while maintaining a near minimax prediction accuracy *as a function of the original data size* n (rather than the suboptimal quantization size).

Interestingly our sufficient conditions on quantization for such tradeoffs are not far from what is already intuitive in practice: quantization centers (essentially a compression of the data) are usually picked so as to be *close to the data*, yet at the same time *far apart* so as to succinctly capture the data (illustrated in Figure 1). Formalizing this intuition, we consider quantization centers $Q = \{q\}$ that form (a) an r-cover of the data $\{X_i\}$ (i.e. r-close to data), and (b) an r-packing, i.e. are r far-apart. We show that, with proper choice of r, and after *variance correction*, the first property (a) maintains minimax prediction accuracy, while the second property (b) is crucial to fast prediction.

As alluded to earlier, the achievable tradeoffs are worked out in terms of the unknown *intrinsic* structure of the data, as captured by the now common notion of *doubling dimension*, known to be small for structured data. The tradeoffs improve with the doubling dimension of the data, independent of the *ambient* dimension D of Euclidean data in $\mathbb{R}^D$. In fact our analysis is over a generic metric space and its intrinsic structure and thus allow general representations of the data.

Finally, the most practical aspect of our tradeoff guarantees is that they explicitly capture the interaction between the quantization parameter r and the kernel bandwidth h. Note that, for a *fixed* h, it is clear that increasing r (more quantization) can only improve time while potentially decreasing accuracy; however the choice of h is not fixed in practice, and paired with the choice of r, the direction of the tradeoffs become unclear. Our analysis simplifies these choices, as we show that the ratio $\alpha = r/h$ acts as a practical *knob* that

1. Y_q is some label assigned to quantization point q, depending on the labels of those data points X_i's represented by q.

©2017 Samory Kpotufe and Nakul Verma.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-538.html.

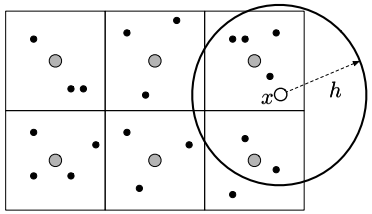


Figure 1: Quantization in kernel regression: given a bandwidth $h > 0$, the estimate $f_Q(x)$ is a weighted average of the Y contributions of quantization-centers $q \in Q$ (gray points) which fall in the ball $B(x, h)$. Each $q \in Q$ contributes the average Y value of the sample points close to it.

can be tuned ($\alpha \to 1$ or $\alpha \to 0$) to favor either prediction time or accuracy according to application needs. In other words, rather than fixing a tradeoff, our analysis shows how to *control* the tradeoff through the interaction of the parameters r and h; such simplified control is valuable to practitioners since appropriate tradeoffs differ across applications.

The resulting approach to kernel prediction, which we term *Kernel-Netting*, or *Netting* for short, can be instantiated with existing fast-search procedures and quantization methods properly adjusted to satisfy the two conditions (a) and (b). In particular, the two conditions are maintained by a simple *farthest-first-traversal* of the data, and are shown experimentally to yield the tradeoffs predicted by the analysis. In particular the prediction accuracy of the vanilla kernel predictor is nearly maintained as shown on datasets from real-world applications. Furthermore, actual tradeoffs are shown empirically to be easily controlled through the knob parameter α.

The rest of the paper is organized as follows. In Section 2 we discuss related work, followed by a discussion of our theoretical results in Section 3. A detailed instantiation of *Netting* is presented in Section 4. This is followed by experimental evaluations in Section 5. All the supporting proofs of our theoretical results are deferred to the appendix.

2. Related Work

We discuss various types of tradeoffs studied in the literature.

2.1 Tradeoffs in General

While practical tradeoffs have always been of interest in Machine Learning, they have been gaining much recent attention as the field matures into new application areas with large data

sizes and dimension. There are many important directions with respect to time-accuracy tradeoffs. We next overview a few of the recent results and directions.

One line of research is concerned with deriving faster implementations for popular procedures such as Support Vector Machines (e.g. Bordes et al., 2005; Rahimi and Recht, 2007; Le et al., 2013; Dai et al., 2014; Alaoui and Mahoney, 2015), parametric regression via sketching (e.g. Clarkson and Woodruff, 2013; Pilanci and Wainwright, 2014; Clarkson et al., 2013; Shender and Lafferty, 2013; Raskutti and Mahoney, 2014).

Another line of research is concerned with understanding the difficulty of learning in constrained settings, including under time constraints, for instance in a minimax sense or in terms of computation-theoretic hardness (see e.g. Agarwal et al., 2011; Cai et al., 2015; Chandrasekaran and Jordan, 2013; Berthet and Rigollet, 2013; Zhu and Lafferty, 2014).

Our focus in this paper is on kernel prediction approaches, and most closely related works are discussed next.

2.2 Tradeoffs in Kernel Prediction

Given a bandwidth parameter $h > 0$, a kernel estimate $f_n(x)$ is obtained as a weighted average of the Y values of (predominantly) those data points that lie in a ball $B(x, h)$. Note that data points outside of $B(x, h)$ might also contribute to the estimate but they are typically given negligible weights. These weights are computed using *kernel* functions that give larger weights to data points closest to x. Kernel weights might be used directly to average Y values (kernel regression, see Gyorfi et al., 2002), or they might be combined in more sophisticated ways to approximate f by a polynomial in the vicinity of x (local polynomial regression, see Gyorfi et al., 2002).

In the naive implementation, evaluation takes time $\Omega(n)$ since weights have to be computed for all points. However many useful approaches in the applied literature help speedup evaluation by combining fast proximity search procedures with methods for approximating kernel weights (see e.g. Carrier et al., 1988; Lee and Gray, 2008; Atkeson et al., 1997; Morariu et al., 2009). More specifically, let $X_{1:n} = \{X_i\}_1^n$, such faster approaches quickly identify the samples in $B(x, h) \cap X_{1:n}$, and quickly compute the kernel weights of these samples. The resulting speedup can be substantial, but *guarantees* on speedups are hard to obtain as this depends on the unknown size of $B(x, h) \cap X_{1:n}$ for future queries x. In particular, it is easy to show as in Proposition 1 below, that $|B(x, h) \cap X_{1:n}|$ is typically at least a root of n for settings of h optimizing statistical accuracy (e.g. ℓ_2 excess risk).

Proposition 1 *There exists a distribution P supported on a subset on $\mathbb{R}^D$ such that the following holds. Let $X_{1:n} \sim P^n$ denote a randomly drawn train set of size n, and $h = O\left(n^{-1/(2+D)}\right)$ (this is the order of a statistically optimal bandwidth in kernel regression where $\mathbb{E}[Y|x]$ is Lipschitz, see Gyorfi et al., 2002). Fix any x in the interior of the support of P. For n sufficiently large, we have with probability at least $1/2$ on $X_{1:n}$ that*

$$|B(x, h) \cap X_{1:n}| = \Omega\left(n^{2/(2+D)}\right).$$

The distribution P in the above proposition corresponds to actually reasonable distribution such as ones with (upper and lower) bounded density on a full-dimensional subset

of $\mathbb{R}^D$. The proposition (whose proof is given in the appendix) is easily extended to low-dimensional subsets $\mathcal{X}$ of $\mathbb{R}^D$, resulting in larger lower-bounds of the form $\Omega\left(n^{2/(2+d)}\right)$, $d = \dim(\mathcal{X})$ (lower-dimensional balls tend to have larger mass).

Approaches that further quantize the data $X_{1:n}$ (or sub-sample from it), e.g. Carrier et al. (1988) can attain better time complexity, however likely at the cost of decreased accuracy in estimates. The resulting tradeoffs on time and accuracy are generally not well understood, and largely depend on how the data is compressed. The present work derive general insights on how to quantize to guarantee good tradeoffs.

The present work extends on insights from an earlier conference work (Kpotufe, 2009) which provides initial results in the case of regression. However, that earlier work only manages a fixed (theoretical) tradeoff between time complexity and accuracy. Instead, as discussed above, the present work provides a *controllable tradeoff* and works out the continuum of achievable rates and time complexity as the knob α is tuned up or down. This requires a refined analysis of all supporting results. Furthermore, the main theoretical results are now shown to carry over to the classification regime as a corollary to the risk bounds for regression; the time complexity bounds are the same for regression and classification since kernel classification involves no extra-work over kernel regression. Finally, the refined analysis of the present work is further supported by extensive experimental evaluation.

3. Overview of Theoretical Results

We start this section with our main theoretical assumptions and useful definitions towards the discussion of results.

3.1 Preliminaries

We are interested in classification or regression where the learner has access to training data $(X_{1:n}, Y_{1:n}) = \{X_i, Y_i\}_1^n \sim P_{X,Y}^n$ where $P_{X,Y}$ is unknown. The goal is to return a regression estimate or classifier that maps future inputs X to outputs Y while achieving good tradeoffs on accuracy and estimation time for $Y = Y(X)$.

The input X belongs to a metric space $(\mathcal{X}, \rho)$. The learner has access to ρ while $\mathcal{X}$ is only observed through the data $X_{1:n}$. The output Y belongs to a space $\mathcal{Y}$, where $\mathcal{Y} = \{0, 1\}$ in the case of classification, or $\mathcal{Y} = \mathbb{R}^{d_Y}$ in the case of regression (we allow multivariate regression outputs).

Next we overview our working assumptions on the input and output spaces. Our main assumptions are rather mild and are specified under Assumptions 1, 2. Our theoretical results generally have no other assumption unless explicitly specified.

3.1.1 Input Space $\mathcal{X}$

We need to capture the complexity of the input data X through notions of data space complexity such as *metric dimension* of the underlying space $(\mathcal{X}, \rho)$.

For intuition, consider a natural two-dimensional object such as a square. A square can be covered by $4 = 2^2$ squares of half its side length. Similarly, a cube can be covered by $8 = 2^3$ cubes of half its side length. Notice that the exponents correspond to the natural

dimensions of the objects. This sort of intuition can be formalized in a number of ways involving how the space $\mathcal{X}$ might be covered by subsets such as *balls* under the metric ρ.

Definition 2 *Let $x \in \mathcal{X}$ and $r > 0$. The* ball $B(x, r)$ *is defined as* $\{x' \in \mathcal{X} : \rho(x', x) \leq r\}$.

Next we introduce proper definitions of what we mean by a *cover* of a space, along with related notions that appear throughout the discussion.

Definition 3 (Covers, packings, and nets) *Consider a metric space $(\mathcal{X}, \rho)$ and $r > 0$.*

- *$Q \subset \mathcal{X}$ is an r-cover of $\mathcal{X}$ if for all $x \in \mathcal{X}$, there exists $q \in Q$, s.t. $\rho(x, q) \leq r$. In other words, the balls $B(q, r)$* cover *$\mathcal{X}$, i.e. $\mathcal{X} \subset \cup_{q \in Q} B(q, r)$.*
- *$Q \subset \mathcal{X}$ is an r-packing if for all $q, q' \in Q$, $\rho(q, q') > r$.*
- *$Q \subset \mathcal{X}$ is an r-net of $X_{1:n}$ if it is an r-cover and an r-packing.*

Note that, on the algorithmic side, we will be interested in packings and covers of the input data $X_{1:n}$ (rather than of $\mathcal{X}$) as a means to attain good time-accuracy tradeoffs.

We can now introduce the following (common) notion of metric dimension.

Definition 4 *The* doubling dimension *of $(\mathcal{X}, \rho)$ is the smallest d such that any ball $B(x, r)$ can be covered by 2^d balls of radius $r/2$.*

The *doubling dimension* captures the inherent complexity of various settings of contemporary interest including sparse datasets and low-dimensional manifolds (Dasgupta and Freund, 2008) and is common in the Machine Learning literature (see for instance Krauthgamer and Lee, 2004; Clarkson, 2005; Beygelzimer et al., 2006; Dasgupta and Freund, 2008; Gottlieb et al., 2013; Reddi and Poczos, 2014).

We have the following assumptions on X:

Assumption 1 *The metric space $(\mathcal{X}, \rho)$ has bounded diameter $\max_{x,x' \in \mathcal{X}} \rho(x, x') = \Delta_{\mathcal{X}}$, and doubling dimension d.*

3.1.2 Output Space $\mathcal{Y}$

Both classification and regression can be treated under the same assumptions on conditional distributions $P_{Y|x}$. In particular we are interested in the behavior of the unknown *regression function* f:

Definition 5 *The* regression function *is defined for all $x \in \mathcal{X}$ as $f(x) = \mathbb{E}[Y|x]$. For classification, its range is $[0, 1]$ while for regression, its range is $\mathbb{R}^{d_Y}$. In a slight abuse of notation, we will let the norm $\|f(x) - f(x')\|$ denote the absolute value on $[0, 1]$ (classification) or the Euclidean norm on $\mathbb{R}^{d_Y}$ (regression).*

We have the following assumptions:

Assumption 2 *The regression function f is λ-Lipschitz for some unknown $\lambda > 0$:*

$$\forall x, x' \in \mathcal{X}, \quad \left\|f(x) - f(x')\right\| \leq \lambda \rho\left(x, x'\right).$$

Let $Y(x) \sim P_{Y|x}$, its variance, namely $\mathbb{E}_{Y|X=x} \|Y - f(x)\|^2$, is uniformly bounded by σ_Y^2 over $x \in \mathcal{X}$. Furthermore, f is bounded over $\mathcal{X}$, i.e. $\sup_{x,x' \in \mathcal{X}} \|f(x) - f(x')\| \leq \Delta_f$.

The Lipschitz condition is common and known to be mild: it captures the idea that whenever x and x' are close, so are $f(x)$ and $f(x')$, uniformly.

Note that the boundedness conditions hold automatically in the case of classification since then Y is itself bounded. In the case of regression, Y itself is allowed to be unbounded; the boundedness assumption on f is immediate from the continuity of f (implied by the Lipschitz condition) whenever $(\mathcal{X}, \rho)$ has bounded diameter (see Assumption 1).

3.1.3 Risk and Excess Risk of a Regression Estimate

Suppose $g : \mathcal{X} \to \mathcal{Y}$ is some estimate of f. We define its ℓ_2 *pointwise risk at* x to be $R(g,x) \doteq \mathbb{E}_{Y|X=x} \|Y - g(x)\|^2$ and its *integrated risk* to be $R(g) \doteq \mathbb{E}_X R(g,X)$. Standard manipulations show that

$$\begin{aligned} R(g,x) &= R(f,x) + \|f(x) - g(x)\|^2, \text{ and therefore} \\ R(g) &= R(f) + \mathbb{E}_X \|g(X) - f(X)\|^2. \end{aligned}$$

In this paper we are interested in the *integrated excess risk*

$$\|g - f\|^2 \doteq R(g) - R(f) = \mathbb{E}_X \|g(X) - f(X)\|^2. \tag{1}$$

Many factors contribute to the complexity of nonparametric regression, more precisely, to the excess risk $\|f_n - f\|^2$ of the regression estimate f_n. Important quantities identified in the literature are the *smoothness* of the regression function $f(x)$, the *Y-variance* $\mathbb{E}\left[\|Y - f(x)\|^2 | X = x\right] \leq \sigma_Y^2$, and the *dimension* of the input space $\mathcal{X}$.

Curse of dimension. Suppose $\mathcal{X} \subset \mathbb{R}^D$. Then, for *any* regression estimate f_n, there exists $P_{X,Y}$ where f is λ-Lipschitz, such that the error $\|f_n - f\|^2$ is of the order $\left(\lambda^D \sigma_Y^2 / n\right)^{2/(2+D)}$ (Stone, 1982). Clearly, the dimension D has a radical effect on the quality of estimates. With kernel regression estimates however, D in the above rate can be replaced by $d \ll D$, where d is the doubling dimension as we will soon see.

Such adaptivity in kernel regression was first shown in Bickel and Li (2006) for data on a manifold,[2] and more recently in Kpotufe and Garg (2013) for a different measure of intrinsic dimension related to *doubling measures* (Clarkson, 2005) and doubling dimension.

We will show how to maintain adaptive rates in terms of the unknown doubling dimension while reducing the data to speedup prediction. This will also be true in the case of classification as explained below.

3.1.4 Risk and Excess risk of a Classifier

The pointwise risk of a classifier $l : \mathcal{X} \mapsto \{0,1\}$ is defined as $R_{0,1}(l,x) = \mathbb{E}_{Y|X=x} \mathbb{1}[l(x) \neq Y]$, and the integrated risk, or *classification error*, is given as $R_{0,1}(l) = \mathbb{E}_X R(l,X)$.

A kernel classifier is a so-called *plug-in* classifier. These are classifiers of the form $l_g(x) = \mathbb{1}[g(x) \geq 1/2]$ where $g \in [0,1]$ is an estimate of the regression function $f(x) = \mathbb{E}[Y|x]$. It is well known that the *Bayes* classifier l_f attains the smallest pointwise risk at every $x \in \mathcal{X}$.

The excess risk of a plug-in classifier is directly upper-bounded by regression excess risk (Devroye et al., 1996):

$$R_{0,1}(l_g) - R_{0,1}(l_f) \leq 2\mathbb{E}\,|g(X) - f(X)| \leq 2\,\|f - g\|. \tag{2}$$

2. The paper concerns local polynomial regression but the results are easily extended to kernel regression.

We can therefore proceed with a single analysis for both classification and regression based solely on the assumptions on the regression function f. We note that the above discussion on *curse of dimension* also apply to classification, unless more regularity is assumed on f (see e.g. a discussion of such additional *noise* assumptions in Audibert and Tsybakov, 2007).

3.2 Results and Key Insights on Tradeoffs

We are now ready to discuss the two main goals of Netting, namely improved estimates and good evaluation time. Many instantiations of the approach are possible, combining a choice of kernel function, and choice of quantization Q with fast-proximity search. The results in this section remain generic, and only assume that Q forms an r-net of the data for some r properly tied to the kernel bandwidth. In Section 4 we discuss how to exactly obtain such quantizations, and discuss details of the prediction procedures.

In contrast with usual analyses of kernel regression, the added technicality in establishing these results is in dealing with non-i.i.d. data: the quantization points $q \in Q$ on which the estimates are defined, and their assigned Y_q values, are *interdependent*, and also depend on the data in nontrivial manners (since we only assume a generic Q that forms an r-net). These interdependencies are handled by properly decomposing the risk, and conditioning on the right sequence of events (see conditional variance and bias bounds of Appendix A).

3.2.1 Main Results

Our main results on tradeoffs are given in the next theorem. The theorem relies on the benign Assumptions 1 and 2 on $\mathcal{X}$ and $\mathcal{Y}$, and considers Netting with a generic quantization Q of the data. It is shown that good tradeoffs can be guaranteed whenever Q is an r-net of the data (see Definition 3). The tradeoffs are given in terms of a *knob* $\alpha = r/h$ which can be dialed up or down to favor either accuracy or time. The regression estimate, under a generic quantization Q is given as follows:

Definition 6 (Regression estimate) *Given a bandwidth $h > 0$, and $0 < \alpha < 3/4$, consider an αh-net Q of the sample $X_{1:n}$, where Q is independent of $Y_{1:n}$. Assign every point $X_i \in X_{1:n}$ to a single $q \in Q$ such that $\rho(X_i, q) \leq \alpha h$. For $q \in Q$, let n_q denote the number of points from $X_{1:n}$ assigned to it, and let $\bar{Y}_q$ denote the average Y value of points assigned to q.*

Let the kernel $K(u)$ be a non increasing function of $u \in [0,\infty)$; K is positive on $u \in [0,1)$, maximal at $u = 0$, and is 0 for $u \geq 1$. Define the regression estimate as

$$\begin{aligned} f_Q(x) &= \sum_{q \in Q} w_q(x) \bar{Y}_q, \text{ where} \\ w_q &= \frac{n_q (K(\rho(x,q)/h) + \epsilon)}{\sum_{q' \in Q} n_{q'} (K(\rho(x,q')/h) + \epsilon)}, \end{aligned} \tag{3}$$

for a positive correction $0 < \epsilon \leq K(3/4)/n^2$.

Most common kernels satisfy the above conditions, e.g., triangle kernel $K(u) = |1-u|_+$, box kernel $K(u) = \mathbb{1}[u \leq 1]$, Epanechnikov $K(u) = (1-u^2)_+$, etc.

The Kernel correction ϵ simply insures that we do not divide by 0; the (technical) upper-bound on ϵ ensures that the error term induced by ϵ is of smaller order than the desired rate. Our results hold for any ϵ as described above, and in our implementation we just set it very small to $1/n^2$.

Theorem 7 *Let f_Q be defined as in (5). The following holds under Assumptions 1 and 2.*

1. Regression: *for any $h > 0$, we have*

$$\mathop{\mathbb{E}}_{(X_{1:n},Y_{1:n})} \|f_Q - f\|^2 \leq \frac{C\left(\sigma_Y^2 + \Delta_f^2\right)}{n \cdot ((3/4-\alpha)h/(2\Delta_{\mathcal{X}}))^d} + (1+\alpha)^2\lambda^2 h^2 + \frac{\Delta_f^2}{n}$$

   where d is the doubling dimension of $\mathcal{X}$, $\Delta_{\mathcal{X}}$ is its diameter, and the constant C depends only on $K(\cdot)$.

2. Classification: *it follows by (2) that the excess risk of the corresponding Netting classifier $l_{f_Q} = \mathbb{1}[f_Q \geq 1/2]$ over the Bayes classifier $l_f = \mathbb{1}[f \geq 1/2]$ satisfies*

$$\mathbb{E}_{X_{1:n},Y_{1:n}} R_{0,1}(l_{f_Q}) - R_{0,1}(l_f) \leq 2\left(\frac{C\left(\sigma_Y^2 + \Delta_f^2\right)}{n \cdot ((3/4-\alpha)h/(2\Delta_{\mathcal{X}}))^d} + (1+\alpha)^2\lambda^2 h^2 + \frac{\Delta_f^2}{n}\right)^{1/2}.$$

3. Time complexity: *the estimate $f_Q(x)$, used in either regression or classification, can be obtained in worst-case time $C'\left(\log(\Delta_{\mathcal{X}}/\alpha h) + \alpha^{-d}\right)$, for some C' that depends on d, but independent of the choice of $x \in \mathcal{X}$, h and α.*

The choice of $0 < \alpha < 3/4$ in the above theorem can be relaxed to any $0 < \alpha < \alpha_0 < 1$, simply replacing $3/4$ everywhere it appears with α_0. The need to for an upper-bound less than 1 on α stems from the need for considering only those centers q that fall in the interior of $B(x,h)$ and hence contribute non-negligible weight to the estimate $f_Q(x)$.

The accuracy results (items 1 and 2) of the above theorem holds simultaneously for any h and is decomposed into variance and bias terms. Thus the bound can be optimized over choices of h as is done later in Corollary 8 (which further establishes that a good choice of h can be obtained through cross-validation).

The risk bounds mainly depend on α through the variance terms of the form $C/n \cdot ((3/4-\alpha)h)^d$ while the bias term is effectively $O(\lambda h)^2$. This bound is best as $\alpha \to 0$ and recovers known variance bounds (see e.g. Gyorfi et al., 2002; Tsybakov and Zaiats, 2009) for kernel regression and classification for $\alpha = 0$ (in which case there is no quantization, i.e. $Q = X_{1:n}$). We emphasize that the bound here is adaptive to the unknown intrinsic dimension d of the data space $\mathcal{X}$ (for $\mathcal{X} \subset \mathbb{R}^D$), which is now known to be a feature of kernel regression and other distance-based procedures (see e.g. Bickel and Li, 2006; Scott and Nowak, 2006; Lafferty and Wasserman, 2007; Kpotufe, 2011; Kpotufe and Dasgupta, 2012; Kpotufe and Garg, 2013; Gottlieb et al., 2013). In other words, the achieved tradeoffs are best for structured data as captured by its intrinsic dimension d.

The time complexity of Theorem 7 is expressed as worst-case over any choice of x and h. In particular, it depends exponentially on the unknown doubling dimension d of $\mathcal{X}$ (through

α and possibly through C'), which as previously discussed, can be much smaller than the ambient dimension D for structured high-dimensional data (e.g. $\mathcal{X}$ is $O(d)$-sparse, or is an $O(d)$-dimensional submanifold of $\mathbb{R}^D$). This sort of worst-case time dependence on intrinsic dimension cannot be avoided as one can construct doubling-metrics $\mathcal{X}$ where fast-similarity search requires time $\Omega\left(2^{O(d)}\log n\right)$ (see e.g. Krauthgamer and Lee, 2004), but fortunately does not seem typical of practical data (see experiments).

The time complexity of the theorem is best as $\alpha \to 3/4$ and depends on the number of points in the range $B(x,h) \cap Q$ (which can be shown to be at most $O(\alpha^{-d})$, independent of n) but does not depend on the number of points in $B(x,h) \cap X_{1:n}$ (the effective range for vanilla kernel-prediction which can be a root of n as shown in Proposition 1). For fixed $\alpha \approx 3/4$, the time complexity is of order $O(\log(\Delta_{\mathcal{X}}/h))$; thus if h is of the form $n^{-1/O(d)}$ (the theoretical order for risk-optimal h) the time complexity is $O(\log n)$.

We emphasize that a time complexity of $O(\log n)$ is the best attainable for fast-range search (see e.g. Krauthgamer and Lee, 2004; Beygelzimer et al., 2006) when the range $B(x,h)$ contains a *constant* number of points. In particular $O(\log n)$ is the time complexity for *single*-nearest-neighbor search (a range of 1 point). This is interesting in that the procedure remains statistically consistent (unlike 1-NN or any constant-NN approach), i.e., $f_Q \to_P f$ for fixed α provided $h \to 0$ and $nh^d \to \infty$.

As is often the case when providing guarantees which hold in the worst-case, the bounds of the above theorem are quite conservative in practice, but accurately identify the dependence of the Netting approach on the parameter α. This is further shown in the experimental evaluations of Section 5; in particular the dependence on the unknown d is often much milder than the worst-case time complexity above, as the achievable time-savings are significant for relatively small values of α ensuring good accuracy.

We discuss the main technical insights in our analysis in the next two subsections on accuracy and time complexity.

3.2.2 Maintaining Good Accuracy

While Q in Theorem 7 is an αh-net, the accuracy statements (items 1 and 2) in fact only require Q to be an αh-cover of $X_{1:n}$. As stated, the classification result (item 2) is a direct consequence of the regression result (item 1). We therefore only need to discuss the technical insights behind the regression accuracy bounds. Item 1 of Theorem 7 is obtained as Lemma 14 proved in the appendix. The main insights are as follows.

As is well known, the excess ℓ_2 risk of a regression estimate f_Q is a sum of its variance and its square bias. The bias of a kernel regression estimate $\hat{f}(x)$ at a point x is a function how far the data points contributing to the estimate are from x, and is order $O(h)$. So, suppose Q is an αh-cover of the data; every center q being close to the data $X_{1:n}$, the data points contributing to the estimate $f_Q(x)$ (the points assigned to the centers q contributing to the estimate) are $O(h)$ close to x, so the bias remains of order $O(h)$ (Lemma 13).

The variance of a quantization-based estimator such as f_Q is more problematic and requires **variance correction**, consisting here of weighting every center q by the number n_q of data points assigned to it. To see this, remember that the variance of a kernel estimate $\hat{f}(x)$ (in fact of distance-based estimates in general) depend on the number of points say n_x contributing to the estimate, and is of the form $1/n_x$. Without the variance correction (i.e.

if every q is weighted only by the kernel K) the information in n_x is lost; hence effectively only the net centers in $B(x,h) \cap Q$ contribute to the estimate; suppose there are $n_{Q,x} < n_x$ such centers, we will then get a worse variance bound of the form $1/n_{Q,x}$. By adjusting the kernel weights of every q with n_q, we properly account for all data contributing their estimates, and can show that the variance of f_Q is then of similar order as that of a vanilla kernel estimate (Lemma 12).

Analysis outline. The bias analysis is broken into the bias due to net centers q close to the query x, and the bias due to centers q far from x; for q's close to x the analysis is fairly standard by following the above insights; the new technicality is in handling the bias due to q's far from x, which is done by considering both situations where x is appropriately close to the data and far from the data.

For the variance analysis, notice that our basic variance correction can be viewed as replacing the original kernel K with one that accounts for both distances $\rho(x,q)$ and density at q. This makes the variance analysis relatively non-standard, along with the fact that we have little control over the data-dependent choice of Q. The variance analysis is divided over queries x that are *particularly* close to centers in Q (and therefore result in high weights w_q and low variance) and those queries x that are far from Q (and therefore result in unstable estimates). Integrating properly over the data space $\mathcal{X}$ and the choice of data $X_{1:n}, Y_{1:n}$ helps circumvent the dependence of Q on $X_{1:n}$, and also reveals that the estimator f_Q remains adaptive to the unknown intrinsic dimension d of $\mathcal{X}$ (by further breaking the integration over an appropriately refined $O(h)$-cover of $\mathcal{X}$ depending on α).

Risk dependence on α. The risk bound of Theorem 7 depends on α through the variance term of the form $1/n \cdot ((3/4-\alpha)h)^d$, assuming $\Delta_{\mathcal{X}} = 1$. Noting that the typical volume of a ball of radius r in a d-dimensional space is of the form r^d, if P_X is nearly uniform in a neighborhood of a query x (or has upper and lower bounded density) then the term $n \cdot ((3/4-\alpha)h)^d$ corresponds to the expected number of points in a ball $B(x,(3/4-\alpha)h)$. In contrast, the variance of a kernel estimate would depend on the number of points in the ball $B(x,h)$. The difference is easily explained: the estimate $f_Q(x)$ is based on the data through assigned centers $q \in Q$ falling in $B(x,h)$; however, the data assigned to these q's—this is the actual data driving the estimate and its variance—are only guaranteed to be in $B(x,(1-\alpha)h)$. Rather than using 1, i.e., considering q's as far as h from x, we consider only q's at most $3h/4$ from x to ensure they contribute non-negligible weights w_q to $f_Q(x)$ (in fact, we can use any $0 < \alpha_0 < 1$ rather than $3/4$), which results in a variance bound depending on the mass of $B(x,(3/4-\alpha)h)$. Thus the risk bound captures the worst-case dependence on α and cannot be further tightened without additional assumptions of the interaction between Q and P_X.

Choice of h. It follows from Theorem 7 above that the bandwidth h can be chosen by cross-validation to obtain a nearly minimax-optimal rate in terms of the unknown distributional parameters d and λ, where α is viewed as a constant. This is stated in the corollary below where $\hat{h}$ is assumed to be picked out of a range $H \doteq \{\Delta_{\mathcal{X}} \cdot 2^{-i}/(3/4-\alpha)\}_0^{\lceil \log n \rceil}$, using an independent validation sample of size n.

Corollary 8 *Assume the conditions of Theorem 7. Assume further that $\mathcal{Y}$ has bounded diameter, and w.l.o.g. contains the 0 vector. Let $\Delta_{\mathcal{Y}} = \max\left\{diam(\mathcal{Y}), \sigma_Y^2 + \Delta_f^2\right\}$. Suppose $n \geq \left(\frac{\lambda \Delta_{\mathcal{X}}}{\Delta_{\mathcal{Y}}(3/4-\alpha)}\right)^{1/(2+d)}$. There exist C depending on $\mathcal{X}$ and the kernel K such that the following holds. Define*

$$\Phi(n) = C\left(\frac{\lambda \Delta_{\mathcal{X}}}{3/4-\alpha}\right)^{2d/(2+d)}\left(\frac{\Delta_{\mathcal{Y}}^2}{n}\right)^{2/(2+d)}, \text{ and } C_1 = \frac{K(3/4)+\epsilon}{K(0)+\epsilon}, C_2 = 1/C_1.$$

A bandwidth $\hat{h}$ can be selected by cross-validation over $O(\log n)$ choices to guarantee

$$\mathbb{E}\left\|f_{Q_{\alpha\hat{h}}} - f\right\|^2 \leq \Phi(n) + 4\Delta_{\mathcal{Y}}^2\sqrt{\frac{\ln n}{n}} + 2(C_2-C_1)^2\Delta_{\mathcal{Y}}^2 \cdot \mathbb{1}\left[n < \left(\frac{\Delta_{\mathcal{Y}}(3/4-\alpha)}{\lambda\Delta_{\mathcal{X}}}\right)^2\right]. \quad (4)$$

In the case of classification, by equation (2), the excess risk for $l_{f_{Q_{\alpha\hat{h}}}}$ is bounded by twice the square-root of the r.h.s. above.

Notice that, for n sufficiently large, the third term in (4) is 0, so the rate is given by $\Phi(n)$. This rate is minimax-optimal for α fixed, with respect to the intrinsic dimension d and Lipschitz parameter λ (see matching lower-bound in Kpotufe, 2011).

3.2.3 Maintaining a Fast Evaluation Time

The time complexity of Theorem 7 (item 3) only requires that the quantization Q be an αh-packing of the data $X_{1:n}$. Together with the requirement of Q being an αh-cover for good accuracy, we get the theorem's requirement of Q being an αh-net.

The time complexity relies on noticing that the time to estimate $f_Q(x)$ depends on the number of centers q in the range $B(x,h) \cap Q$ and how fast these points can be identified by a given fast-range search procedure. The weights n_q can be precomputed during preprocessing and therefore do not contribute to estimation time.

Intuitively, the farther points in $B(x,h) \cap Q$ are, the fewer such points there should be. In particular, if they are over h far apart, there could only be one point in $B(x,h) \cap Q$. This intuition carries over to αh-nets where all points in $B(x,h) \cap Q$ are over αh apart. This is stated in the following lemma which relies on simple arguments that are standard in analyses of fast proximity search procedures. Its proof is given in the appendix.

Lemma 9 *Suppose $\mathcal{X}$ has doubling dimension d. Let Q be a set of centers forming an αh-packing, $0 < \alpha < 1$. Then for any fixed $c \geq 1$, $\max_{x \in \mathcal{X}} |Q \cap B(x, c \cdot h)| \leq C\alpha^{-d}$, where C depends on d and c.*

Again, the above bound cannot be tightened, since following directly from the definition of doubling-dimension, it is easy to construct $\mathcal{X}$ such that there is a ball of radius 1 with largest α-packing of size roughly α^{-d} (e.g. $\mathcal{X}$ is a d-dimensional hypercube with l_∞ metric).

Now it is left to ensure that $Q \cap B(x,h)$ can be identified quickly. As previously discussed, we can employ a generic fast-range search procedure. We will expect the time complexity to be of the form $\tau(Q) + \alpha^{-d}$, where $\tau(Q)$ is the time to identify a near-neighbor q of a query x in Q (range-search typically then consists of traversing a precomputed neighborhood of q). In particular, the statement of Theorem 7 relies on results from Krauthgamer and Lee (2004) and is proven in Lemma 16 in the appendix.

4. Detailed Instantiation of Netting

We now discuss instantiations of the insights of Theorem 7. In Section 4.1 below, we discuss a few ways to exactly obtain an r-net of the data. This is followed by the regression and classification procedures in Section 4.2.

4.1 Obtaining an r-net

Algorithm 1 below details a standard way to building an r-net offline using a *farthest-first-traversal*. This is an $O(n^2)$ procedure, but is easily implemented to handle all $r > 0$ simultaneously in $O(n)$ space (notice that $Q_r \subset Q_{r'}$ in Algorithm 1 whenever $r > r'$).

Algorithm 1: r-net with farthest-first traversal

Input: points $\{x_i\}$ of size n, $r > 0$.
Initialize $Q_r \leftarrow \{x_1\}$.
Define $\rho(x_i, Q_r) \doteq \min_{x_j \in Q_r} \rho(x_i, x_j)$.
while $\max_{\{x_i\}} \rho(x_i, Q_r) > r$ **do**
  Add $x \doteq \text{argmax}_{\{x_i\}} \rho(x_i, Q_r)$ to Q_r.
end while
Return r-net Q_r.

For fixed r, an r-net can be obtained *online* in time $\sum_{i=1}^{n} t(i)$ where $t(\cdot)$ is the time-complexity of an online black-box nearest neighbor search procedure (e.g. Krauthgamer and Lee, 2004; Beygelzimer et al., 2006). Thus, if interested in a fixed r, the r-net could be obtained in time $O(n \times$ time for a nearest-neighbor search) which, as observed empirically for fast-search procedures, could behave as low as $O(n \log n)$ depending on the fast-search procedure and the data (see e.g. the nice discussions in Krauthgamer and Lee, 2004; Beygelzimer et al., 2006). This online approach is given in Algorithm 2.

Algorithm 2: r-net online

Input: points $\{x_i\}$ of size $n > 1$, $r > 0$.
Initialize $Q_r \leftarrow \{x_1\}$.
Define $\rho(x_i, Q_r) \doteq \min_{x_j \in Q_r} \rho(x_i, x_j)$.
for $i = 2$ to n **do**
  Add x_i to Q_r if $\rho(x_i, Q_r) > r$.
end for
Return r-net Q_r.

Proposition 10 (Correctness of Algorithms 1 and 2) *Q_r is an r-net over $\{x_i\}$.*

Proof By construction Q_r is an r-packing in both procedures. It is also an r-cover since the only points not added to (the current) Q_r are within distance r of Q_r. ■

4.2 Prediction Procedure

Algorithm 3 describes the prediction procedure f_Q in the case of regression. Instantiations for classification follow as direct plug-ins $l_{f_Q} \doteq \mathbb{1}[f_Q \geq 1/2]$ of the corresponding regression estimate f_Q as discussed in Section 3.1.4.

Algorithm 3: Netting: r-nets, $f_Q(x)$

Input: $h > 0$, query point x, αh-net Q of the data.
Assign every $x \in X_{1:n}$ to closest center $q \in Q$ (break ties arbitrarily).
Precomputed: $\bar{Y} \leftarrow$ average Y value over training data.
  $n_q \leftarrow$ number of training points assigned to $q \in Q$.
  $\bar{Y}_q \leftarrow$ Average Y value of training points assigned to $q \in Q$.
  Pick any $0 < \epsilon \leq K(3/4)/n^2$.

$$\textbf{Return} \quad f_Q(x) \leftarrow \frac{\sum_{q \in Q} n_q K(\rho(x,q)/h)\bar{Y}_q + \epsilon n \bar{Y}}{\sum_{q' \in Q} n_{q'} K(\rho(x,q')/h) + \epsilon n}.$$

The estimation of $f_{Q_r}(x)$ is already given in Theorem 7, and is given in an equivalent form (closer to implementation) in Algorithm 3. The form given in Algorithm 3 pulls out the kernel correction ϵ and makes it clear that we only need to identify those q's with nonnegative kernel values (using any black-box range-search procedure). In light of Theorem 7 we can expect this version of Netting to be competitive with kernel regression in terms of accuracy but considerably better in terms of time complexity. We will see that this holds empirically.

5. Experiments

In this section we discuss the practical benefits of Netting, and in particular verify experimentally that the knob α can be adjusted to favor estimation time or prediction accuracy on benchmark datasets from a diverse set of application domains.

5.1 Tradeoff Between Estimation Time and Prediction Accuracy

Datasets. We selected a number of prediction benchmarks from different application domains, including some large-scale prediction tasks. Datasets are summarized in Table 1.

The first prediction task is taken from robotic controls. The goal here is to learn the inverse dynamics for the movement of an anthropomorphic robotic arm. That is, we want to predict the torque required to move the robotic arm to a given state, where each state is a 21-dimensional vector of position, velocity and acceleration (Vijayakumar and Schaal, 2000).[3] Different torques are applied at seven joint positions, we predict the first torque value in our experiments as a regression task. This dataset contains 44,484 training examples.

The second prediction task comes from the medical image analysis domain. The dataset consists of 384 features extracted from 53,500 Computed Tomography (CT) scan images

3. The dataset was taken from Rasmussen and Williams (2006).

Dataset Name	Application Domain	n	D	Task
SARCOS[3]	Robotic Control	44k	21	regression
Location of CT slices (axial)[4]	Medical Image Analysis	54k	384	regression
MiniBooNE[4]	Particle Physics	130k	50	classification

Table 1: Summary of the datasets used for time-accuracy tradeoff experiments.

Datasets	SARCOS (42k)	CT Slices (51k)	MiniBooNE (128k)
$\alpha = 1/6$	0.99 - 2.03	0.93 - 1.29	0.99 - 1.17
$\alpha = 2/6$	**0.99 - 4.10**	0.92 - 2.04	0.99 - 1.65
$\alpha = 3/6$	**0.98 - 6.31**	**0.91 - 3.17**	**0.99 - 4.05**
$\alpha = 4/6$	**0.96 - 7.70**	**0.91 - 5.40**	**0.98 - 6.42**
$\alpha = 5/6$	0.89 - 9.26	0.85 - 11.94	**0.94 - 8.83**
$\alpha = 6/6$	0.77 - 10.14	0.43 - 15.33	0.88 - 10.22

Table 2: The two numbers in each cell are *error ratio*, **vs.** *time ratio* (w.r.t. base kernel method), for different datasets. The error ratio (loss in accuracy) is the error of the base kernel method over the error of the method (for a given α value). The time ratio (time improvement) is the estimation time of the base kernel method over the estimation time of the method (for a given α value). Shown are averages over multiple runs (as described in the experimental section). For the MiniBooNE dataset, the error is the 0-1 classification error, while for the other datasets we use RMSE. Some of the best tradeoffs are highlighted, appearing at different α values for different datasets. As predicted by the analysis, in many such cases, little accuracy is lost when following our conditions on quantization. Furthermore, as predicted by the analysis, the *knob* α can be seen to control the tradeoffs, i.e. relative accuracy goes down as $\alpha \to 1$, while time gain goes up.

from 74 distinct patients (Graf et al., 2011).[4] The goal is to predict the relative location of the CT slice on the axial axis of the human body. This is a regression task.

The third prediction task is from Particle Physics, where the goal is to classify elementary particles. Particles are either electron neutrinos (signal) or muon neutrinos (background) collected from the MiniBooNE (Booster Neutrino) experiment (Roe et al., 2005).[4] There are 50 features computed for each event (observation), with a total of 130,065 events.

Experimental setup. We consider increasing settings of the trade-off knob α from $\frac{1}{6}, \frac{2}{6}, \ldots, \frac{6}{6}$, while we monitor the prediction accuracy and evaluation time for different sample sizes of each dataset. The results (accuracy and prediction time w.r.t. the vanilla kernel method) for the largest sample sizes are reported in Table 2, while in Figure 2 we report results for increasing sample sizes.

A basic experiment is as follows. We select 2,000 random samples from each dataset for testing, and use (part of) the rest for training. Training sizes are logarithmically space

4. The dataset was taken from UCI Machine Learning repository (Lichman, 2013).

from 100 samples to the maximum training dataset size. For each training size, results are averaged over 5 draws of training and testing samples.

Implementation details. Both r-nets and kernel regression (baseline) can significantly benefit from using a fast nearest neighbor search mechanism. We use the default fast rangesearch functionality available in Matlab for nearest neighbor search in all our experiments. The choice of weighting kernel can also affect the prediction quality. We use the triangular kernel ($K(u) = (1 - |u|)_+$) for all our experiments; we get similar trends (with slightly worse accuracies) if we use the box kernel instead.

The bandwidth parameter for each procedure (corresponding to each choice of α, where $\alpha = 0$ is the vanilla kernel regression estimate) were selected using a 5-fold cross validation (over the training sample).

Bandwidth range. For a good range of bandwidth (for any procedure) we use a two step approach: we first approximate a good bandwidth choice h_1 by iterating over 10 sizes ranging from minimum training-data diameter to maximum training-data diameter (equally spaced). We then use h_1 to get a refined range to search, namely $[h_1/2, 2h_1]$, over which we do a full sweep of 100 equally spaced bandwidth values.

Demo code. A Matlab implementation of the r-nets algorithm along with a test demo is available at `http://www.cse.ucsd.edu/~naverma/code/rnets_prediction.zip`.

5.2 Discussion of Results

We report the root mean squared error (RMSE) for regression tasks, and the 0-1 loss for classification.

In Figure 2, the errors and estimation (wall-clock) times are reported for each dataset for increasing values of the sample size n. For baseline comparison we also include the performance of a simple "average" predictor, i.e. one that returns the global average Y value of the training samples.

In Table 2, the results are on the same experiments as in Figure 2, but we now focus on the largest sample size for each dataset. While Figure 2 shows the general trend of tradeoffs as n varies, Table 2 emphasizes the exact accuracy and time ratios relative to the vanilla kernel procedure as a function of the knob α.

The empirical results match the theoretical analysis. The knob α is seen to control the achievable tradeoffs, i.e., an α value close to 0 results in an accuracy on par with that of kernel regression, while as $\alpha \to 1$, the accuracy decreases while time improves, sometimes dramatically for some of the datasets. Some of the best tradeoffs are highlighted in bold in Table 2. Some of the best tradeoffs are obtained for SARCOS (regression) and the MiniBoone (classification) tasks, where we observe negligible decrease in accuracy for nearly 8 to 9 times speedups in prediction time. The largest decrease in accuracy is observed with the CT Slices dataset, although the achieved tradeoffs are still of significant practical value.

The general trend is maintained as the sample size n grows (see Figure 2), with better accuracy for low-values of α (as the statistical aspect gets easier with sample size), and more speedup (as also suggested by the time bounds in the earlier analysis).

Finally, we contrast these results with those obtained using a more common quantization approach such as a *kd-tree*. A kd-tree (see appendix) is a hierarchical partitioning of the

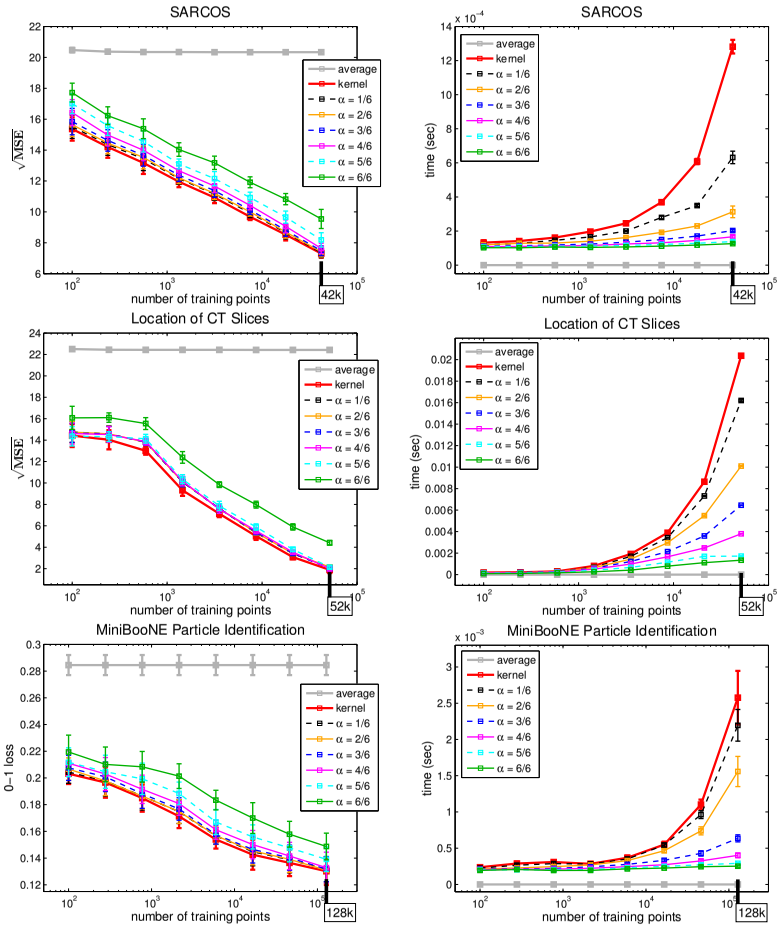


Figure 2: Average prediction error (left column) and the corresponding evaluation time (right column) of r-nets (Algorithm 3).

space that aims to quickly decrease the diameter of the data in each partition-cell. Each level of a kd-tree (from 0 to about $\log n$, with the last level being all of the data) yields a quantization Q of the data by simply assigning datapoints in each cell to their mean. For comparison, we pick a bandwidth h by cross-validation (as described above for the other methods), and estimate f by the quantized predictor f_Q of (5). In other words we simply replace our earlier quantization with r-nets with those defined by levels of the common kd-tree. Again we use the default Matlab fast-similarity search over Q.

The resulting error and time ratios (w.r.t. the base kernel method) are given in Table 3 below, where we only show the last few levels which the best prediction errors (note that the last level, not shown, is exactly the vanilla kernel method since there is no quantization).

Datasets	SARCOS (42k)	CT Slices (51k)	MiniBooNE (128k)
level 14	0.80 - 1.58	0.34 - 4.22	0.90 - 4.95
level 15	0.93 - 1.42	0.41 - 2.89	0.92 - 3.49
level 16	-	-	0.95 - 2.24

Table 3: kd-tree quantization results

We see that the best error-ratios (error of kernel over error of f_Q) attained are worse than those using the r-nets as prescribed by our analysis, and furthermore the time-speedups (time of kernel over time of f_Q) are also significantly worse. This is most apparent for the CT Slices dataset, which seems the hardest for maintaining the accuracy of the original kernel predictor (0.41-2.89 is the best tradeoff for kd-tree, vs, 0.91-5.40 for the r-net approach). We note however that, if time is favored, then a kd-tree can be preferable to the r-net approach since much better time can be attained at lower tree levels (further quantization) but with significant decrease in prediction accuracy (see Appendix C).

6. Final Remarks and Potential Future Directions

Our analysis reveals simple sufficient conditions on data-quantization to approximately maintain prediction accuracy while significantly improving prediction time. Furthermore, we identify a practical knob tying together quantization and prediction parameters, so as to simplify practical control of achievable tradeoffs. These new insights are validated experimentally on real-world datasets.

One possible future direction is to extend the insights herein to situations where the data is a mixture of structured subspaces of varying complexity so as to better capture the richness of real-world data. It could be that different quantization schemes should be employed for each such subspace once identified. Another potential direction is in characterizing what happens with a random subsampling scheme (as a way to quantize data) rather than with data-quantization; in particular it will be interesting to characterize situations where a random subsample (approximately) achieves the sufficient quantization conditions derived in the present work.

Acknowledgements

We would like to thank Kevin Liou for assisting in running the experiments.

Appendix A. Analysis for r-nets

We first analyze the risk for a fixed choice of h, then we give guarantees for a simple cross-validation procedure for choosing a good h.

Throughout this section we assume that Q is independent of $Y_{1:n}$, and is an αh-cover of $X_{1:n}$. Here we consider the Kernel Netting estimate f_Q defined as follows.

Let the kernel $K(u)$ be a non increasing function of $u \in [0,\infty)$; K is positive on $u \in [0,1)$, maximal at $u = 0$, and is 0 for $u \geq 1$. For $q \in Q$, let n_q denote the number of points from $X_{1:n}$ assigned to it, and let $\bar{Y}_q$ denote the average Y value of points assigned to q.

Define the regression estimate as

$$\begin{aligned} f_Q(x) &= \sum_{q\in Q} w_q(x)\bar{Y}_q, \text{ where} \\ w_q &= \frac{n_q(K(\rho(x,q)/h)+\epsilon)}{\sum_{q'\in Q} n_{q'}(K(\rho(x,q')/h)+\epsilon)}, \end{aligned}$$

where the positive constant $\epsilon \leq K(3/4)/n^2$ ensures the ratio is well defined.

All the results below are given under Assumptions 1 and 2.

A.1 Risk Bound for Fixed h

Throughout this section we assume $0 < h < \Delta_{\mathcal{X}}$, $0 < \alpha < 3/4$ and we let $Q = Q_{\alpha h}$. We'll bound the risk for f_Q for any fixed choice of h. The results in this section only require the fact that Q is a cover of the data and thus preserves local information.

For any estimator $f_n(x) \doteq f_n(x; X_{1:n}, Y_{1:n})$, define $\widetilde{f}_n(x) \doteq \mathbb{E}_{Y_{1:n}|X_{1:n}} f_n(x)$, i.e., the conditional expectation of the estimate for $X_{1:n}$ fixed. In particular we will be interested in $\widetilde{f}_Q(x) \doteq \mathbb{E}_{Y_{1:n}|X_{1:n}} f_Q(x)$. We have the following standard decomposition of the (conditional) excess risk into variance and bias terms:

$$\forall x \in \mathcal{X},\ \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \|f_n(x) - f(x)\|^2 = \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \left\|f_n(x) - \widetilde{f}_n(x)\right\|^2 + \left\|\widetilde{f}_n(x) - f(x)\right\|^2. \quad (5)$$

For linear estimates (as those discussed in this work), the above decomposition yields the following simple proposition providing a rough bound on the risk for all $x \in \mathcal{X}$.

Proposition 11 *Fix $x \in \mathcal{X}$ and consider a linear estimate $f_n(x) = \sum_i w_i(x) Y_i$, where $\sum_i w_i(x) = 1$. We then have*

$$\forall x \in \mathcal{X},\ \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \|f_n(x) - f(x)\|^2 \leq \sigma_Y^2 + \Delta_f^2, \quad (6)$$

where σ_Y^2 and Δ_f^2 are respectively shown to bound the variance and bias terms in (5).

Proof It is easily verified that, for independent random vectors v_i with expectation $\mathbf{0}$, $\mathbb{E}\left\|\sum_i v_i\right\|^2 = \sum_i \mathbb{E}\|v_i\|^2$. We therefore have for the variance,

$$\mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \left\|f_n(x) - \widetilde{f}_n(x)\right\|^2 \leq \sum_i w_i^2(x) \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \left\|Y_i - \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} Y_i\right\|^2 \leq \sum_i w_i^2(x)\cdot\sigma_Y^2 \leq \sigma_Y^2.$$

For the bias term we have by a simple triangle inequality,

$$\left\|\widetilde{f}_n(x) - f(x)\right\|^2 = \left\|\sum_i w_i(x)\left(f(x_i) - f(x)\right)\right\|^2 \leq \left(\sum_i w_i(x)\Delta_f\right)^2 \leq \Delta_f^2.$$

■

The rough bound of Proposition 11 is used for particular $x \in \mathcal{X}$ that end up in potentially bad estimates. The following lemmas establish refined variance and bias bounds for those queries $x \in \mathcal{X}$ close to the data $X_{1:n}$.

We'll proceed by bounding each term of (5) separately in the following two lemmas, and then combining these bounds in Lemma 14. We let μ denote the marginal measure over $\mathcal{X}$ and μ_n denote the corresponding empirical measure.

Lemma 12 (Variance at x) *Let $h > 0$, and $0 < \alpha < 3/4$. Fix $X_{1:n}$, and let Q be an αh-cover of $X_{1:n}$. Let $x \in \mathcal{X}$ and let $\mathcal{E}_\alpha(x)$ denote the event that $X_{1:n} \cap (B(x,(3/4-\alpha)\cdot h)) \neq \emptyset$. We have*

$$\mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \left\|f_Q(x) - \widetilde{f}_Q(x)\right\|^2 \leq \frac{2K(0)\cdot\sigma_Y^2}{K(3/4)\cdot n\cdot\mu_n\left(B(x,(3/4-\alpha)\cdot h)\right)}\cdot\mathbb{1}[\mathcal{E}_\alpha(x)] + \sigma_Y^2\cdot\mathbb{1}\left[\bar{\mathcal{E}}_\alpha(x)\right].$$

Proof Conditioned on $X_{1:n}$ and $Q \subset X_{1:n}$, the Y_i values are mutually independent and so are the $\bar{Y}_q$ values. Therefore, in what follows, we repeatedly use the fact that $\|\sum_i v_i\|^2 = \sum_i \mathbb{E}\|v_i\|^2$ for independent $\mathbf{0}$-mean vectors v_i. We have

$$\begin{aligned} \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \left\|f_Q(x) - \widetilde{f}_Q(x)\right\|^2 &= \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \left\|\sum_{q\in Q} w_q(x)\left(\bar{Y}_q - \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \bar{Y}_q\right)\right\|^2 \\ &= \sum_{q\in Q} w_q^2(x) \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \left\|\bar{Y}_q - \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \bar{Y}_q\right\|^2 \\ &= \sum_{q\in Q} w_q^2(x) \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \left\|\sum_{i:X_i\in X_{1:n}(q)} \frac{1}{n_q}\left(Y_i - \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} Y_i\right)\right\|^2 \\ &= \sum_{q\in Q} w_q^2(x) \sum_{i:X_i\in X_{1:n}(q)} \frac{1}{n_q^2} \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} \left\|Y_i - \mathop{\mathbb{E}}_{Y_{1:n}|X_{1:n}} Y_i\right\|^2 = \sum_{q\in Q} w_q^2(x)\frac{\sigma_Y^2}{n_q} \\ &\leq \left(\max_{q\in Q}\left\{w_q(x)\frac{\sigma_Y^2}{n_q}\right\}\right)\sum_{q\in Q} w_q = \max_{q\in Q}\left\{w_q(x)\frac{\sigma_Y^2}{n_q}\right\} \\ &= \max_{q\in Q} \frac{(K(x,q,h)+\epsilon)\sigma_Y^2}{\sum_{q'\in Q} n_{q'}(K(x,q',h)+\epsilon)} \\ &\leq \min\left\{\sigma_Y^2, \frac{2K(0)\sigma_Y^2}{\sum_{q\in Q} n_q K(x,q,h)}\right\}. \end{aligned} \quad (7)$$

Assume the event $\mathcal{E}_\alpha(x)$. To bound the fraction in (7), we lower-bound the denominator as:

$$\sum_{q\in Q} n_q K(x,q,h) \geq \sum_{q:\rho(x,q)\leq 3h/4} n_q K(3/4) \geq K\left(3/4\right)\cdot n\cdot \mu_n(B(x,(3/4-\alpha)\cdot h)).$$

The last inequality follows by remarking that, since Q is an αh-cover of $X_{1:n}$, the ball $B(x,(3/4-\alpha)\cdot h)$ can only contain points from $\cup_{q:\rho(x,q)\leq 3h/4} X_{1:n}(q)$. Plug this last inequality into (7) and conclude. ■

Lemma 13 (Bias at x) *Fix $X_{1:n}$, let $0<\alpha<3/4$, and $h>0$. Suppose Q is an αh-cover of $X_{1:n}$. Let $x\in\mathcal{X}$, and define $\mathcal{E}_\alpha(x)$ as the event that $X_{1:n}\cap(B(x,(3/4-\alpha)\cdot h))\neq\emptyset$. We have*

$$\left\|\widetilde{f}_Q(x)-f(x)\right\|^2 \leq \left((1+\alpha)^2\lambda^2h^2+\frac{\Delta_f^2}{n}\right)\cdot \mathbb{1}[\mathcal{E}_\alpha(x)]+\Delta_f^2\cdot\mathbb{1}\left[\bar{\mathcal{E}}_\alpha(x)\right].$$

Proof The bias term $\left\|\widetilde{f}_Q(x)-f(x)\right\|^2$ equals

$$\left\|\sum_{q\in Q}\frac{w_q(x)}{n_q}\sum_{i:X_i\in X_{1:n}(q)}(f(X_i)-f(x))\right\|^2 \leq \sum_{q\in Q}\frac{w_q(x)}{n_q}\sum_{i:X_i\in X_{1:n}(q)}\|f(X_i)-f(x)\|^2 \quad (8)$$

$$\leq \sum_{q\in Q}\frac{w_q(x)}{n_q}\sum_{i:X_i\in X_{1:n}(q)}\lambda^2\rho\left(X_i,x\right)^2 \quad (9)$$

where in (8) we applied Jensen's inequality on the norm square. It immediately follows from (8) that $\left\|\widetilde{f}_Q(x)-f(x)\right\|^2$ is at most Δ_f^2, for any $x\in\mathcal{X}$.

Now assume the event $\mathcal{E}_\alpha(x)$. We bound the r.h.s of (8) by breaking the summation over the following two subsets of Q: those centers q close to x and those far from x. Starting with close centers, and using (9) we have

$$\sum_{q:\rho(x,q)<h}\frac{w_q(x)}{n_q}\sum_{i:X_i\in X_{1:n}(q)}\lambda^2\rho\left(X_i,x\right)^2 \leq \sum_{q:\rho(x,q)<h}\frac{w_q(x)}{n_q}\sum_{i:X_i\in X_{1:n}(q)}\lambda^2\left(\rho\left(x,q\right)+\rho\left(q,X_i\right)\right)^2$$

$$\leq \sum_{q:\rho(x,q)<h}\frac{w_q(x)}{n_q}\sum_{i:X_i\in X_{1:n}(q)}(1+\alpha)^2\lambda^2h^2.$$

Next, we have

$$\sum_{q:\rho(x,q)\geq h}\frac{w_q(x)}{n_q}\sum_{X_i\in X_{1:n}(q)}\|f(X_i)-f(x)\|^2 \leq \sum_{q:\rho(x,q)\geq h} w_q(x)\Delta_f^2$$

$$= \frac{\Delta_f^2\sum_{q:\rho(x,q)\geq h} n_q\epsilon}{\sum_{q:\rho(x,q)\geq h} n_q\epsilon+\sum_{q:\rho(x,q)<h} n_q\left(K(x,q,h)+\epsilon\right)}$$

$$= \Delta_f^2\left(1+\frac{\sum_{q:\rho(x,q)<h} n_q\left(K(x,q,h)+\epsilon\right)}{\sum_{q:\rho(x,q)\geq h} n_q\epsilon}\right)^{-1}$$

$$\leq \Delta_f^2\left(1+\frac{K(3/4)}{\sum_{q:\rho(x,q)\geq h} n_q\epsilon}\right)^{-1} \leq \Delta_f^2\left(1+\frac{K(3/4)}{n\epsilon}\right)^{-1} \leq \frac{\Delta_f^2}{1+n},$$

where the first inequality is due to the fact that, since $\mu_n(B(x,(3/4-\alpha)\cdot h))>0$, the set $B(x,3h/4)\cap Q$ cannot be empty (remember that Q is an αh-cover of $X_{1:n}$). This concludes the argument. ■

Lemma 14 (Integrated excess risk) *Given a bandwidth $h>0$, and $0<\alpha<3/4$, consider an αh-cover Q of the sample $X_{1:n}$. Assign every point $X_i\in X_{1:n}$ to a single $q\in Q$ such that $\rho\left(X_i,q\right)\leq\alpha h$. We have:*

$$\mathop{\mathbb{E}}_{(X_{1:n},Y_{1:n})}\|f_Q-f\|^2 \leq \frac{C\left(\sigma_Y^2+\Delta_f^2\right)}{n\cdot((3/4-\alpha)h/(2\Delta_{\mathcal{X}}))^d}+(1+\alpha)^2\lambda^2h^2+\frac{\Delta_f^2}{n}$$

where the constant C depends only on $K(\cdot)$.

Proof To ease notation, let $\tau\doteq(3/4-\alpha)$. Applying Fubini's theorem, the expected excess risk, $\mathbb{E}_{(X_{1:n},Y_{1:n})}\|f_Q-f\|^2$, can be written as

$$\mathop{\mathbb{E}}_{X}\mathop{\mathbb{E}}_{(X_{1:n},Y_{1:n})}\|f_Q(X)-f(X)\|^2\cdot\left(\mathbb{1}[\mu_n(B(X,\tau h))>0]+\mathbb{1}[\mu_n(B(X,\tau h))=0]\right).$$

By lemmas 12 and 13 and equation (5), for $X=x$ fixed, $\mathbb{E}_{(X_{1:n},Y_{1:n})}\|f_Q(x)-f(x)\|^2\cdot\mathbb{1}[\mu_n(B(x,\tau h))>0]$ is upper bounded by

$$C_1\mathop{\mathbb{E}}_{X_{1:n}}\left[\frac{\sigma_Y^2\mathbb{1}[\mu_n(B(x,\tau h))>0]}{n\mu_n(B(x,\tau h))}\right]+(1+\alpha)^2\lambda^2h^2+\frac{\Delta_f^2}{n}$$

$$\leq C_1\left(\frac{2\sigma_Y^2}{n\mu(B(x,\tau h))}\right)+(1+\alpha)^2\lambda^2h^2+\frac{\Delta_f^2}{n}, \quad (10)$$

where for the last inequality we used the fact that for a binomial $b(n,p)$, $\mathbb{E}\left[\frac{\mathbb{1}[b(n,p)>0]}{b(n,p)}\right]\leq\frac{2}{np}$ (see lemma 4.1 of Gyorfi et al. (2002)).

On the other hand, $\mathbb{E}_{(X_{1:n},Y_{1:n})} \|f_Q(x) - f(x)\|^2 \cdot \mathbb{1}[\mu_n(B(x,\tau h)) = 0]$ is upper bounded by

$$\left(\sigma_Y^2 + \Delta_f^2\right) \mathop{\mathbb{E}}_{X_{1:n}} \mathbb{1}[\mu_n(B(x,\tau h)) = 0] = \left(\sigma_Y^2 + \Delta_f^2\right)(1 - \mu(B(x,\tau h)))^n$$
$$\leq \left(\sigma_Y^2 + \Delta_f^2\right) e^{-n\mu(B(x,\tau h))} \leq \frac{\left(\sigma_Y^2 + \Delta_f^2\right)}{n\mu(B(x,\tau h))}. \quad (11)$$

Combining (10) and (11) we can then bound the expected excess risk as

$$\mathop{\mathbb{E}}_{(X_{1:n},Y_{1:n})} \|f_Q - f\|^2 \leq \frac{C_2\left(\sigma_Y^2 + \Delta_f^2\right)}{n} \mathop{\mathbb{E}}_{X}\left[\frac{1}{\mu(B(X,\tau h))}\right] + (1+\alpha)^2\lambda^2 h^2 + \frac{\Delta_f^2}{n}. \quad (12)$$

The expectation on the r.h.s. can now be bounded using a standard covering argument (see e.g. Gyorfi et al. (2002)). Let $\{z_i\}_1^N$ be a $\tau h/2$-cover of $\mathcal{X}$. Notice that for any z_i, $x \in B(z_i, \tau h/2)$ implies $B(x,\tau h) \supset B(z_i, \tau h/2)$. We therefore have

$$\begin{aligned}\mathop{\mathbb{E}}_{X}\left[\frac{1}{\mu(B(X,\tau h))}\right] &\leq \sum_{i=1}^N \mathop{\mathbb{E}}_{X}\left[\frac{\mathbb{1}[X \in B(z_i,\tau h/2)]}{\mu(B(X,\tau h))}\right] \leq \sum_{i=1}^N \mathop{\mathbb{E}}_{X}\left[\frac{\mathbb{1}[X \in B(z_i,\tau h/2)]}{\mu(B(z_i,\tau h/2))}\right] \\ &= N \leq C_3\left(\frac{2\Delta_{\mathcal{X}}}{\tau h}\right)^d, \text{ where } C_3 \text{ depends just on } \mathcal{X}.\end{aligned}$$

We conclude by combining the above with (12) to obtain

$$\mathop{\mathbb{E}}_{(X_{1:n},Y_{1:n})} \|f_Q - f\|^2 \leq \frac{C\left(\sigma_Y^2 + \Delta_f^2\right)}{n(\tau h/2\Delta_{\mathcal{X}})^d} + (1+\alpha)^2\lambda^2 h^2 + \frac{\Delta_f^2}{n}.$$

■

A.2 Choosing a Good h by Empirical Risk Minimization

In this section, we analyze the following simple procedure for choosing a good h:

> Define $H \doteq \{\Delta_{\mathcal{X}} \cdot 2^{-i}/(3/4-\alpha)\}_0^{\lceil \log n \rceil}$. For every $h \in H$, pick the αh-net $Q_{\alpha h}$ over the sample $(X_{1:n}, Y_{1:n})$, and let $f_{Q_{\alpha h}}$ be as previously defined (equation 5). Draw a new sample $(X'_{1:n}, Y'_{1:n})$ of size n. For every $h \in H$, test $f_{Q_{\alpha h}}$ on $(X'_{1:n}, Y'_{1:n})$; let the empirical risk be minimized at $\hat{h}$, i.e. $\hat{h} \doteq \operatorname{argmin}_{h\in H} \frac{1}{n}\sum_{i=1}^n \|f_{Q_{\alpha h}}(X'_i) - Y'_i\|^2$.
> Return $f_{Q_{\alpha\hat{h}}}$ as the final regression estimate.

Proof [Corollary 8] The sample size n is lower-bounded so that there exists a universal constant C_0 such that

$$\tilde{h} \doteq C_0\left(\frac{\Delta_{\mathcal{X}}}{3/4-\alpha}\right)^{d/(2+d)}\left(\frac{\Delta_{\mathcal{Y}}^2}{\lambda^2 n}\right)^{1/(2+d)}$$

is a dyadic number greater than $\Delta_{\mathcal{X}}/n$.

First, let's consider the situation where $h \in H$, i.e. $\tilde{h} \leq \Delta_{\mathcal{X}}/(3/4-\alpha)$. Note that, whenever $n \geq \left(\frac{\Delta_{\mathcal{Y}}(3/4-\alpha)}{\lambda\Delta_{\mathcal{X}}}\right)^2$, we can choose a universal C_0 above so that $\tilde{h} \in H$.

We have by Lemma 14 that for some C, regression with $\tilde{h}$ satisfies

$$\mathop{\mathbb{E}}_{X_{1:n},Y_{1:n}} \left\|f_{Q_{\alpha\tilde{h}}} - f\right\|^2 \leq C'\left(\frac{\lambda\Delta_{\mathcal{X}}}{3/4-\alpha}\right)^{2d/(2+d)}\left(\frac{\Delta_{\mathcal{Y}}^2}{n}\right)^{2/(2+d)}. \quad (13)$$

Applying McDiarmid's to the empirical risk followed by a union bound over H, we have that, with probability at least $1 - 1/\sqrt{n}$ over the choice of $(X'_{1:n}, Y'_{1:n})$, for all $h \in H$

$$\left|\mathop{\mathbb{E}}_{X,Y}\|f_{Q_{\alpha h}}(X) - Y\|^2 - \frac{1}{n}\sum_{i=0}^n \left\|f_{Q_{\alpha h}}(X'_i) - Y'_i\right\|^2\right| \leq \Delta_{\mathcal{Y}}^2\sqrt{\frac{\ln(|H|\sqrt{n})}{n}}.$$

It follows that $\mathbb{E}_{X,Y}\left\|f_{Q_{\alpha\hat{h}}}(X) - Y\right\|^2 \leq \inf_{h\in H}\mathbb{E}_{X,Y}\|f_{Q_{\alpha h}}(X) - Y\|^2 + 2\Delta_{\mathcal{Y}}^2\sqrt{\frac{\ln(|H|\sqrt{n})}{n}}$, which by equation (1) implies

$$\begin{aligned}\left\|f_{Q_{\alpha\hat{h}}} - f\right\|^2 &\leq \inf_{h\in H}\|f_{Q_{\alpha h}} - f\|^2 + 2\Delta_{\mathcal{Y}}^2\sqrt{\frac{\ln(|H|\sqrt{n})}{n}} \quad (14)\\ &\leq \left\|f_{Q_{\alpha\tilde{h}}} - f\right\|^2 + 2\Delta_{\mathcal{Y}}^2\sqrt{\frac{\ln(|H|\sqrt{n})}{n}}\end{aligned}$$

Take the expectation (given the randomness in the two samples) over this last inequality and apply (13) to obtain the first two terms of the bound on $\mathbb{E}\left\|f_{Q_{\alpha\hat{h}}} - f\right\|^2$.

Now consider the case where $\tilde{h} \notin H$, i.e. $\tilde{h} > \Delta_{\mathcal{X}}/(3/4-\alpha)$. Consider any $h \geq \Delta_{\mathcal{X}}/(3/4-\alpha)$. For any such h, and any query $x \in \mathcal{X}$, $0 \leq \rho(x,q) \leq \Delta_{\mathcal{X}} \leq 3h/4$ for all $q \in Q \subset \mathcal{X}$. It follows that for any $q \in Q$, $C_1/n \leq w_q \leq C_2/n$ where C_1, C_2 are as defined in the corollary's statement. In particular, the estimates (at any x) using $\tilde{h}$ or using $l_0 = \Delta_{\mathcal{X}}/(3/4-\alpha)$ differ by at most $(C_2 - C_1)\left\|\bar{Y}\right\| \leq (C_2 - C_1)\Delta_{\mathcal{Y}}$, where $\bar{Y} = \sum_i Y_i/n$. It follows that the difference in errors (fixing $X_{1:n}, Y_{1:n}$) is at most

$$\left\|f_{Q_{\alpha l_0}} - f\right\|^2 \leq \left(\left\|f_{Q_{\alpha\tilde{h}}} - f\right\| + \left\|f_{Q_{\alpha l_0}} - f_{Q_{\alpha\tilde{h}}}\right\|\right)^2 \leq 2\left\|f_{Q_{\alpha\tilde{h}}} - f\right\|^2 + 2(C_2 - C_1)^2\Delta_{\mathcal{Y}}^2.$$

Therefore, whenever $n < \left(\frac{\Delta_{\mathcal{Y}}(3/4-\alpha)}{\lambda\Delta_{\mathcal{X}}}\right)^2$, let $\tilde{h} > \Delta_{\mathcal{X}}/(3/4-\alpha)$, and obtain by (14) that

$$\left\|f_{Q_{\alpha\hat{h}}} - f\right\|^2 \leq 2\left\|f_{Q_{\alpha\tilde{h}}} - f\right\|^2 + 2(C_2 - C_1)^2\Delta_{\mathcal{Y}}^2 + 2\Delta_{\mathcal{Y}}^2\sqrt{\frac{\ln(|H|\sqrt{n})}{n}},$$

with probability at least $1 - 1/\sqrt{n}$. Again take the expectation w.r.t. the two samples and apply (13) to obtain the full corollary's statement. ■

Appendix B. Time Complexity Bounds

B.1 Time Bound of Proposition 1

The proof of Proposition 1 follows easily from considering the mass of balls in a high-dimensional space. The proof is given below.

Proof [Proposition 1] Let for instance P be uniform on a closed ball $\mathcal{X}$ of $\mathbb{R}^D$. Then any ball of radius h centered on the interior of $\mathcal{X}$ has mass $\Omega(h^D)$. Thus for $h = \Omega\left(n^{-1/(2+D)}\right)$, $\mathbb{E}\,|B(x,h) \cap X_{1:n}| = n \cdot P(B(x,h)) = \Omega\left(n^{2/(2+D)}\right)$. Now, let P_n denote the empirical distribution on $X_{1:n}$; it is well known that by Bernstein's bounds (Bousquet et al., 2004), we have with probability at least $1/2$ that

$$P(B(x,h)) \leq P_n(B(x,h)) + \sqrt{C \cdot P_n(B(x,h))/n} + C/n,$$

for some universal constant C. This implies that if $P_n(B(x,h)) < C/n$, then $P(B(x,h)) < 3C/n$. For n sufficiently large, $P(B(x,h)) = \Omega\left(n^{-D/(2+D)}\right) \geq 3C/n$, so we must have $P_n(B(x,h)) \geq C/n$. It then follows by the above Bernstein's inequality that $P(B(x,h)) \leq 3P_n(B(x,h))$, which yields the proposition's statement. ■

B.2 Evaluation Times for Netting

Proof [Proof of Lemma 9] It is well known that an r-packing Q of a ball $B(x,r')$, $r' > r$, has size at most that of an $r/2$-cover of $B(x,r')$ (Clarkson, 2005). This size is bounded by $C(r'/r)^d$, which is seen by applying the definition of doubling dimension recursively. ■

Lemma 15 (Paraphrased from Krauthgamer and Lee (2004)) *Let $\{x_i\}$ be a finite set of points from a metric space $(\mathcal{X}, \rho)$ with doubling dimension d.*

Define the aspect-ratio *of $\{x_i\}$ as $\tau \doteq \max_{i,j} \rho(x_i, x_j) \,/\, \min_{i,j} \rho(x_i, x_j)$. For any query $x \in \mathcal{X}$ and $r > 0$, the range $B(x,r) \cap \{x_i\}$ can be obtained in time at most*

$$C\left(|B(x, c \cdot r) \cap \{x_i\}| + \log \tau\right),$$

for some C depending on d, and a universal constant $c \geq 1$.

Lemma 16 *Assume the conditions of Theorem 7. For any $x \in \mathcal{X}$, the estimate $f_Q(x)$ can be computed in time $C\left(\log(\Delta_{\mathcal{X}}/\alpha h) + \alpha^{-d}\right)$, where C depends on d.*

Proof This is a direct corollary to Lemmas 9 and 15, by noticing that the aspect ratio of Q is at most $\Delta_{\mathcal{X}}/\alpha h$. ■

Appendix C. kd-tree Construction Details

Algorithm 4: kd-tree

Input: dataset $X_{1:n}$.
if $|X_{1:n}| \leq 1$ **then**
  Return leaf.
end if
Let j be the largest-variance coordinate of the data $x \in X_{1:n}$.
Define $t \doteq \text{median}\left\{x^j : x \in X_{1:n}\right\}$.
$X_{\text{left}} \doteq \{x \in X_{1:n} \mid x^j \leq t\}$.
$X_{\text{right}} \doteq \{x \in X_{1:n} \mid x^j > t\}$.
$\text{Tree}_{\text{left}} = k\text{d-tree}(X^{\text{left}}_{1:n})$.
$\text{Tree}_{\text{right}} = k\text{d-tree}(X^{\text{right}}_{1:n})$.
Return $(\text{Tree}_{\text{left}}, \text{Tree}_{\text{right}})$.

C.1 kd-tree Results

The following table lists the error and time ratios (w.r.t. to kernel prediction) achieved at every level of the tree (omitting the last level with no quantization).

Datasets	SARCOS (42k)	CT Slices (51k)	MiniBooNE (128k)
level 0	0.34 - 271.67	0.08 - 2,351.77	0.46 - 412.06
level 1	0.37 - 11.08	0.09 - 260.33	0.46 - 515.86
level 2	0.38 - 10.52	0.09 - 301.92	0.48 - 11.71
level 3	0.39 - 10.77	0.10 - 254.53	0.51 - 12.63
level 4	0.40 - 10.46	0.10 - 209.23	0.54 - 11.96
level 5	0.41 - 10.73	0.11 - 174.25	0.60 - 12.06
level 6	0.43 - 10.22	0.12 - 126.92	0.62 - 11.15
level 7	0.45 - 9.44	0.13 - 90.44	0.64 - 10.51
level 8	0.48 - 8.30	0.15 - 67.72	0.64 - 9.38
level 9	0.52 - 7.77	0.16 - 43.64	0.69 - 8.42
level 10	0.56 - 6.05	0.18 - 27.14	0.71 - 7.21
level 11	0.60 - 4.86	0.20 - 16.25	0.78 - 9.70
level 12	0.65 - 3.23	0.23 - 9.63	0.81 - 7.33
level 13	0.71 - 2.18	0.28 - 6.17	0.87 - 5.84
level 14	0.80 - 1.58	0.34 - 4.22	0.90 - 4.95
level 15	0.93 - 1.42	0.41 - 2.89	0.92 - 3.49
level 16	-	-	0.95 - 2.24

Table 4: kd-tree quantization results

References

A. Agarwal, J.C. Duchi, P.L. Bartlett, and C. Levrard. Oracle inequalities for computationally budgeted model selection. In *Conference on Learning Theory (COLT)*, pages 69–86, 2011.

A. Alaoui and M.W. Mahoney. Fast randomized kernel ridge regression with statistical guarantees. In *Advances in Neural Information Processing Systems (NIPS)*, pages 775–783, 2015.

C. Atkeson, A. Moore, and S. Schaal. Locally weighted learning. *AI Review*, 1997.

J. Audibert and A.B. Tsybakov. Fast learning rates for plug-in classifiers. *The Annals of Statistics*, 35(2):608–633, 2007.

Q. Berthet and P. Rigollet. Complexity theoretic lower bounds for sparse principal component detection. In *Conference on Learning Theory (COLT)*, pages 1046–1066, 2013.

A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbors. In *International Conference on Machine Learning (ICML)*, pages 97–104, 2006.

P. Bickel and B. Li. Local polynomial regression on unknown manifolds. *Technical Report Department of Statistics UC Berkley*, 2006.

A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *Journal of Machine Learning Research (JMLR)*, 6:1579–1619, 2005.

O. Bousquet, S. Boucheron, and G. Lugosi. Introduction to statistical learning theory. In *Advanced Lectures on Machine Learning*, pages 169–207. Springer, 2004.

T. Cai, T. Liang, and A. Rakhlin. Computational and statistical boundaries for submatrix localization in a large noisy matrix. *CoRR, abs/1502.01988*, 2015.

J. Carrier, L. Greengard, and V. Rokhlin. A fast adaptive multipole algorithm for particle simulations. *SIAM Journal on Scientific and Statistical Computing*, 9(4):669–686, 1988.

V. Chandrasekaran and M.I. Jordan. Computational and statistical tradeoffs via convex relaxation. *Proceedings of the National Academy of Sciences (PNAS)*, 110(13):E1181–E1190, 2013.

K.L. Clarkson. Nearest-neighbor searching and metric space dimensions. *Nearest-Neighbor Methods for Learning and Vision: Theory and Practice*, 2005.

K.L. Clarkson and D.P. Woodruff. Low rank approximation and regression in input sparsity time. In *ACM symposium on Theory of Computing (STOC)*, pages 81–90, 2013.

K.L. Clarkson, P. Drineas, M. Magdon-Ismail, M.W. Mahoney, X. Meng, and D.P. Woodruff. The fast cauchy transform and faster robust linear regression. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 466–477, 2013.

B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M.F. Balcan, and L. Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3041–3049, 2014.

S. Dasgupta and Y. Freund. Random projection trees and low dimensional manifolds. In *ACM Symposium on Theory of Computing (STOC)*, pages 537–546, 2008.

L. Devroye, L. Gyorfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

L. Gottlieb, A. Kontorovich, and R. Krauthgamer. Efficient regression in metric spaces via approximate Lipschitz extension. In *Similarity-Based Pattern Recognition*, pages 43–58. Springer, 2013.

F. Graf, H.-P. Kriegel, M. Schubert, S. Poelsterl, and A. Cavallaro. 2d image registration in ct images using radial image descriptors. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 607–614, 2011.

L. Gyorfi, M. Kohler, A. Krzyzak, and H. Walk. *A Distribution Free Theory of Nonparametric Regression*. Springer, New York, NY, 2002.

S. Kpotufe. Fast, smooth and adaptive regression in metric spaces. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1024–1032, 2009.

S. Kpotufe. k-NN regression adapts to local intrinsic dimension. In *Advances in Neural Information Processing Systems (NIPS)*, pages 729–737, 2011.

S. Kpotufe and S. Dasgupta. A tree-based regressor that adapts to intrinsic dimension. *Journal of Computer and System Sciences (JCSS)*, 78(5):1496–1515, 2012.

S. Kpotufe and V. Garg. Adaptivity to local smoothness and dimension in kernel regression. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3075–3083, 2013.

R. Krauthgamer and J. Lee. Navigating nets: Simple algorithms for proximity search. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 798–807, 2004.

J. Lafferty and L. Wasserman. Statistical analysis of semi-supervised regression. In *Advances in Neural Information Processing Systems (NIPS)*, pages 801–808, 2007.

Q. Le, T. Sarlós, and A. Smola. Fastfood – approximating kernel expansions in loglinear time. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2013.

D. Lee and A. Gray. Fast high-dimensional kernel summations using the Monte Carlo multipole method. In *Advances in Neural Information Processing Systems (NIPS)*, pages 929–936, 2008.

M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

V.I. Morariu, B.V. Srinivasan, V.C. Raykar, R. Duraiswami, and L.S. Davis. Automatic online tuning for fast Gaussian summation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1113–1120, 2009.

M. Pilanci and M.J. Wainwright. Iterative Hessian sketch: Fast and accurate solution approximation for constrained least-squares. *CoRR, abs/1411.0347*, 2014.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1177–1184, 2007.

G. Raskutti and M. Mahoney. A statistical perspective on randomized sketching for ordinary least-squares. *arXiv preprint arXiv:1406.5986*, 2014.

C. Rasmussen and C. Williams. Gaussian processes for machine learning - SARCOS dataset. `http://www.gaussianprocess.org/gpml/data/`, 2006.

S. Reddi and B. Poczos. k-NN regression on functional data with incomplete observations. In *Uncertainty in Artificial Intelligence (UAI)*, pages 692–701, 2014.

B.P. Roe, H. Yang, J. Zhu, Y. Liu, I. Stancu, and G. McGregor. Boosted decision trees as an alternative to artificial neural networks for particle identification. *Nuclear Instruments and Methods in Physics Research Section A*, 543:577–584, 2005.

C. Scott and R.D. Nowak. Minimax-optimal classification with dyadic decision trees. *IEEE Transactions on Information Theory*, 52(4):1335–1353, 2006.

D. Shender and J. Lafferty. Computation-risk tradeoffs for covariance-thresholded regression. In *Proceedings of The 30th International Conference on Machine Learning (ICML)*, pages 756–764, 2013.

C.J. Stone. Optimal global rates of convergence for non-parametric estimators. *The Annals of Statistics*, 10:1340–1353, 1982.

A.B. Tsybakov and V. Zaiats. *Introduction to Nonparametric Estimation*. Springer, 2009.

S. Vijayakumar and S. Schaal. LWPR: An $O(n)$ algorithm for incremental real time learning in high dimensional space. pages 1079–1086, 2000.

Y. Zhu and J. Lafferty. Quantized estimation of Gaussian sequence models in Euclidean balls. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3662–3670, 2014.

Journal of Machine Learning Research 18 (2017) 1-21 Submitted 11/16; Revised 3/17; Published 4/17

Asymptotic behavior of Support Vector Machine for spiked population model

Hanwen Huang huanghw@uga.edu
Department of Epidemiology and Biostatistics
University of Georgia
Athens, GA 30602, USA

Editor: Xiaotong Shen

Abstract

For spiked population model, we investigate the large dimension N and large sample size M asymptotic behavior of the Support Vector Machine (SVM) classification method in the limit of $N, M \to \infty$ at fixed $\alpha = M/N$. We focus on the generalization performance by analytically evaluating the angle between the normal direction vectors of SVM separating hyperplane and corresponding Bayes optimal separating hyperplane. This is an analogous result to the one shown in Paul (2007) and Nadler (2008) for the angle between the sample eigenvector and the population eigenvector in random matrix theorem. We provide not just bound, but sharp prediction of the asymptotic behavior of SVM that can be determined by a set of nonlinear equations. Based on the analytical results, we propose a new method of selecting tuning parameter which significantly reduces the computational cost. A surprising finding is that SVM achieves its best performance at small value of the tuning parameter under spiked population model. These results are confirmed to be correct by comparing with those of numerical simulations on finite-size systems. We also apply our formulas to an actual dataset of breast cancer and find agreement between analytical derivations and numerical computations based on cross validation.

Keywords: Asymptotic behavior, Spiked population model, Support Vector Machine

1. Introduction

The Support Vector Machine (SVM) is a state-of-the-art powerful classification method proposed by Vapnik (Vapnik, 1995). It has been widely used in bioinformatics and many other disciplines and has achieved a lot of success. Like other classification methods, SVM may suffer from a loss of generalization ability in high dimensional situations as shown by Figure 1 which displays the application of SVM to a high dimensional two class toy example with class labels $+1$ and -1. The data have dimension $N = 100$, with $M_+ = 45$ data vectors from Class $+1$ represented as circles, and $M_- = 45$ data vectors from Class -1 represented as plus. The two distributions are nearly standard normal except that the mean in the first dimension is shifted to $+\mu$ and $-\mu$ for Class $+1$ and Class -1 respectively. Here $\mu = 1$. Figure 1 shows the projections of the data onto the two-dimensional subspace determined by the first dimension (dashed line) and the normal vector (solid line) of the SVM separating hyperplane. The angle between these two directions can be used to determine the generalization ability of the classifier. A classifier who has good generalization properties should have small angle. For the particular example shown in Figure 1, the angle is 56.6°. Therefore, projection of a new data vector onto the SVM direction cannot be expected to provide effective discrimination. As mentioned by Marron et al. (2007), the reason is that the estimated SVM classifier is driven only by very particular aspects of the realization of the training data at hand. New data will have their own quite different quirks, which will bear no relation to these.

Hall et al. (2005) studied the High Dimensional Low Sample Size (HDLSS) asymptotics of SVM and shown that for fixed sample size $M = M_+ + M_-$, as $N \to \infty$ the angle depends on the signal size μ which is defined as half of the distance between the means of two distributions for this example. Assume that μ increases with N as N^γ, then if $\gamma > 1/2$, SVM is strongly consistent, i.e., the angle approaches to 0°; if $\gamma < 1/2$, SVM is strongly inconsistent, i.e., the angle approaches to 90°; if $\gamma = 1/2$, the angle is between 0° and 90°. Therefore the signal size μ has to be large enough in order to gain some prediction power.

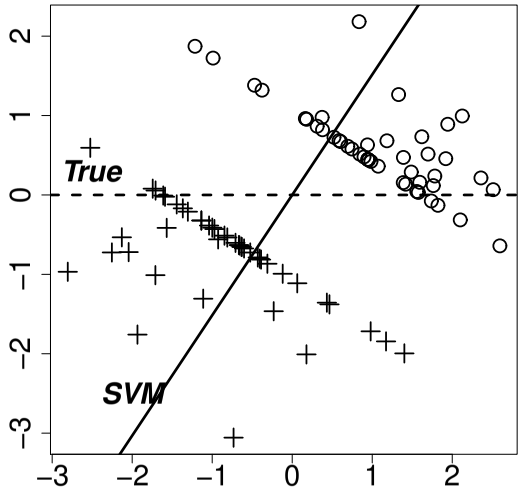


Figure 1: Toy examples, illustrating the performance of SVM on high dimensional data with $N = 100$ and sample size $M = M_+ + M_- = 90$. The circles denote the data from Class $+1$ and and the plus denote the data from Class -1. The dashed line represents the first dimension which is the true difference in the Gaussian means. The solid line represents the normal vector of SVM separating hyperplane. The angle between the solid and dashed lines is 56.6°.

Analogous conclusion has been drawn in the context of unsupervised learning for Principal Component Analysis (PCA). The study of sample covariance matrices is fundamental in multivariate analysis. It is well known that the sample covariance matrix is a consistent estimator of the population covariance matrix for fixed dimension N and sample size $M \to \infty$.

©2017 Hanwen Huang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-564.html.

The PCA consistency in HDLSS context (fixed M and $N \to \infty$) was studied in Jung and Marron (2009); Jung et al. (2012) and it was shown that the asymptotic behavior of the Principal Component (PC) directions of sample covariance matrix depend on the size of the corresponding eigenvalues. Assume that the eigenvalue of the sample covariance matrix λ increases with N in the order of power γ, i.e. $\lambda \sim N^{\gamma}$. Then, if $\gamma>1/2$, the corresponding estimated PC direction is strongly consistent, i.e. the angle between the estimated direction and its population counterpart is $0°$; if $\gamma<1/2$, the corresponding estimated PC direction is strongly inconsistent, i.e. the angle is $90°$; if $\gamma = 1/2$, the angle is random and follows a certain distribution.

On the other hand, with the development of modern high-throughput technologies, it is not uncommon to have data where M is comparable in size to N, or substantially larger. There has been considerable effort to establish asymptotic results for sample eigenvalues and eigenvectors under the assumption that N and M grow at the same rate, that is, $M/N \to \alpha>0$ (see review Bai (1999)). The limiting distribution of eigenvalues of the sample covariance matrix was derived in Marcenko and Pastur (1967). Johnstone (2001) studied the distribution of the largest eigenvalue in PCA. Baik and Silverstein (2006) investigated the convergence of the sample eigenvalues and eigenvectors under the spiked population. The degree of discrepancy in terms of the angle between the directions of sample and population eigenvectors was further derived in Paul (2007); Nadler (2008) for both $0<\alpha<1$ and $\alpha>1$ situations. A phenomenon of retarded learning was observed that the angle goes through a critical phase transition from angle equal to $90°$ for $\lambda<\sqrt{\alpha}$ to angle less than $90°$ for $\lambda>\sqrt{\alpha}$. Therefore, one can only detect signals whose corresponding eigenvalues are larger than the critical value $\sqrt{\alpha}$ in PCA. More general results have been obtained by Hoyle and Rattray (2004) and Hoyle and Rattray (2007); Hoyle (2010) for general population covariance matrix.

In the present work, we study the analogous asymptotic results in the joint limit $N, M \to \infty$ with $M/N = \alpha$ in the supervised learning context for the SVM classification method. We focus on the generalization performance of SVM by deriving analytical results for the angle between the estimated direction and the true direction and investigating how this angle depends on μ, α and other model parameters. We consider a spiked population model and assume that the data from each class are generated from a purely noise model spiked with a few significant eigenvalues. We derive the analytical results using the replica method developed in statistical mechanics and also compare with numerical simulations on finite size systems. To the best of our knowledge, the present paper is the first that provides not just bounds, but sharp predictions of the asymptotic behavior of the SVM estimators in the limit $N, M \to \infty$ at fixed $M/N = \alpha$.

An immediate application of our analytical findings is for tuning parameter selection. SVM is required to solve problem of determining the tuning parameter τ that characterizes the strength of the penalty term. Cross validation (CV) is a practically useful strategy for handling this task; its basic concept is to evaluate the prediction error by examining the data under control. Smaller values of the CV error are expected to be better to express the generative model of the data. The minimum, if it exists, of the CV error when changing τ is thus considered to obtain an optimal value of τ. However, conducting CV through grid search for finding the minimizer of the CV error is rather computationally expensive especially for high dimensional data. Here we propose a new method of selecting optimal value of τ base on analytical evaluation for the angle between the estimated SVM direction and true direction which considerably reduces the computational cost. Under the spiked population assumption, smaller angle indicates smaller test error. A surprising finding is that SVM achieves its best performance at small value of the tuning parameter. All analytical results are confirmed by numerical experiments on finite-size systems and our formula is clarified to work well for moderate-size systems.

The rest of this paper is organized as follows: In Section 2, we state SVM in the context of spike population model. The analytical results for large N, M asymptotics are presented. In Section 3, we show the result of numerical experiments to support our analytical results. An application of the proposed tuning parameter selection method to the breast cancer data is also presented in this section. The last section is devoted to the conclusion.

2. Method

In the classification problem, we are given a training dataset consisting of M observations $(\mathbf{x}_i, y_i)$, for $i = 1, \cdots, M$. Here $\mathbf{x}_i \in R^N$ represents an input vector and $y_i \in \{+1, -1\}$ denotes the corresponding output class label. Each $(\mathbf{x}_i, y_i)$ is an independent random vector distributed according to a joint distribution function $p(\mathbf{x}, y)$. We assume that y has probability p_+ to be $+1$ and probability p_- to be -1 with $p_+ + p_- = 1$. Conditional on $y = +1, -1$, $\mathbf{x}$ follows multivariate distributions $p(\mathbf{x}|y = +1)$, $p(\mathbf{x}|y = -1)$ with mean $\boldsymbol{\mu}_+, \boldsymbol{\mu}_-$ and covariance matrices $\boldsymbol{\Sigma}_+, \boldsymbol{\Sigma}_-$, respectively. Without loss of generality, assume $\boldsymbol{\mu}_+ = -\boldsymbol{\mu}_- = \boldsymbol{\mu}$. Similar to linear discriminant analysis, we make an additional simplifying homoscedasticity assumption $\boldsymbol{\Sigma}_+ = \boldsymbol{\Sigma}_- = \boldsymbol{\Sigma}$. Here $\boldsymbol{\mu} \in R^N$ and $\boldsymbol{\Sigma}$ denotes the $N \times N$ matrix. Based on this setting, the data from two classes are generated from two multivariate distributions with the same covariance but different means. The signal size can be characterized by $\mu = \|\boldsymbol{\mu}\| = \sqrt{\sum_{j=1}^{N} \mu_j^2}$.

We consider a spiked covariance model here. For high dimensional data, typically only few components are biologically important. The remaining structures can be considered as i.i.d. background noise. Therefore, in high-dimensional settings, a collection of data can be modeled by a low-rank signal plus noise structure (Ma, 2013; Liu et al., 2008). We use a factor analysis model to explain correlations between a set of N variables by means of a smaller set of K causal factors. Specifically, we assume the following:

Assumption 1 *Each observation vector* $\mathbf{x}$ *from Class* $+1$ *can be viewed as an independent instantiation of the following generative model*

$$\mathbf{x} = \boldsymbol{\mu} + \sum_{m=1}^{K} \sigma \sqrt{\lambda_m} \mathbf{v}_m z_m + \boldsymbol{\epsilon}. \tag{1}$$

Here $\boldsymbol{\mu}$ *is the mean vector,* $\lambda_m>0$, $\mathbf{v}_m \in R^N$ *are orthonormal vectors, i.e.* $\mathbf{v}_m^T \mathbf{v}_m = 1$ *and* $\mathbf{v}_m^T \mathbf{v}_{m'} = 0$ *for* $m \neq m'$, $\hat{\boldsymbol{\mu}} = \boldsymbol{\mu}/\mu = \mathbf{v}_1$. *The random variables* $z_1, \cdots, z_K \overset{i.i.d}{\sim} N(0,1)$. *The vector* $\boldsymbol{\epsilon} = \{\epsilon_1, \cdots, \epsilon_N\}$ *whose elements* $\epsilon_j s$ *are i.i.d random variables with* $E(\epsilon_j) = 0$, $E(\epsilon_j^2) = \sigma^2$ *and* $E(\epsilon_j^3)<\infty$. *The* $\epsilon_j s$ *are independent of* $z_m s$. *The* $\mathbf{x}$ *from Class* -1 *can be modeled in a similar way with* $\boldsymbol{\mu}$ *replaced by* $-\boldsymbol{\mu}$.

In model (1), λ_m represents the strength of the m-th biological component, and σ^2 represents the level of background noise. The real biology is typically low-dimensional, i.e. $K \ll N$. Considering signal as one of the biological components, without loss of generality, we assume that $\boldsymbol{\mu}$ is in the same direction as $\mathbf{v}_1$, i.e. $\hat{\boldsymbol{\mu}} = \mathbf{v}_1$. Note that the eigenvalue λ_m is not necessarily decreasing in m and λ_1 is not necessarily the largest eigenvalue. From (1), the covariance matrix is

$$\boldsymbol{\Sigma} = \sigma^2 \mathbf{I}_N + \sum_{m=1}^{K} \sigma^2 \lambda_m \mathbf{v}_m \mathbf{v}_m^T, \tag{2}$$

where $\mathbf{I}_N$ is N-dimensional identity matrix. Although the ϵ_js are i.i.d, we didn't impose any parametric form for the distribution of ϵ_j which allows for very flexible covariance structures for $\mathbf{x}$, and thus the results are quite general. The requirement for the finite third order moment is to ensure Berry-Esseen central limit theorem applies. The Assumption 1 is also called spiked population model and has been used in many situations, see Baik and Silverstein (2006); Marcenko and Pastur (1967); Johnstone (2001) for examples. Such a population covariance is a finite rank perturbation of multiple of the identity matrix. In other words, all but finitely many eigenvalues of the population covariance matrix are the same. Examples of spiked data include speech recognition (Trevor Hastie, 1995), mathematical finance (LALOUX et al., 2000), wireless communications (Telatar, 1999), and physics of mixture (Sear and Cuesta, 2003).

The task of linear classification is to construct a hyperplane $\mathbf{x}^T\mathbf{w} = 0$ ($\mathbf{w} \in R^N$) so that the new data vector $\mathbf{x}$ is assigned to Class $+1$ when $\mathbf{x}^T\mathbf{w}>0$ and Class -1 otherwise. If the training data are linearly separable, SVM seeks to find this hyperplane such that the minimal distance between the hyperplane and the data point from each class is maximized. The hard-margin SVM solution can be formulated in terms of the following optimization problem

$$\begin{aligned} &\min_{\mathbf{w}} \left[\mathbf{w}^T\mathbf{w}\right] \\ \text{s.t.} \quad &\frac{y_i \mathbf{x}_i^T \mathbf{w}}{\sqrt{N}} \geq 1, \ i = 1, \cdots, M. \end{aligned} \tag{3}$$

To extend SVM to cases in which the data are not linearly separable, we introduce the slack variables ξ_i for $i = 1, \cdots, M$. The soft-margin SVM solution can be formulated in terms of the following optimization problem

$$\begin{aligned} &\min_{\mathbf{w}} \left[\mathbf{w}^T\mathbf{w} + \tau \sum_{i=1}^{M} \xi_i\right] \\ \text{s.t.} \quad &\frac{y_i \mathbf{x}_i^T \mathbf{w}}{\sqrt{N}} + \xi_i \geq 1, \ \xi_i \geq 0, \ i = 1, \cdots, M, \end{aligned} \tag{4}$$

where the tuning parameter τ determines the trade-off between increasing the margin-size and ensuring that the $\mathbf{x}_i$ lie on the correct side of the margin. For sufficiently large values of τ, the soft-margin SVM will behave identically to the hard-margin SVM. We will show below that, as $\tau \to \infty$, the asymptotic result of soft-margin SVM is the same as the asymptotic result of hard-margin SVM.

For the setting described in Assumption 1, the normal direction vector of the separating hyperplane based on Bayes optimal rule is in the same direction as $\boldsymbol{\mu}$. Therefore, the performance of any classification method can be evaluated by the angle between the normal direction vector of its separating hyperplane and $\boldsymbol{\mu}$. Propositions 1 and 2 provide the sharp prediction of the high-dimensional limiting angles for hard-margin SVM and soft-margin SVM respectively.

Proposition 1 *Under Assumption 1, in the limit $N, M \to \infty$, with fixed $\alpha = M/N$, denote θ the angle between $\boldsymbol{\mu}$ and $\mathbf{w}$ solved from the hard-margin SVM algorithm (3), then $\cos\theta$ converges to ρ that is determined by the following two nonlinear equations*

$$\frac{1-\rho^2}{1+\lambda_1\rho^2} = \alpha \int_{-\infty}^{z_c} Dz(z_c - z)^2, \tag{5}$$

$$\frac{\rho}{\sqrt{1+\lambda_1\rho^2}} = \alpha \int_{-\infty}^{z_c} Dz(z_c - z)\left(\frac{\mu}{\sigma} + \frac{\lambda_1\rho}{\sqrt{1+\lambda_1\rho^2}} z\right). \tag{6}$$

where z_c is an unknown parameter needs to be estimated, μ, σ, λ_1 are defined in (2), and the standard notation $Dz = \frac{dz}{\sqrt{2\pi}} \exp\left(-\frac{z^2}{2}\right)$.

All the proofs are given in the supplementary materials. From equations (5) and (6), we can solve two unknown parameters ρ and z_c given α, μ, σ, and λ_1. It is interesting to note that the results do not depend on $\lambda_2, \cdots, \lambda_K$ which means that only the variance along the signal direction has influence on SVM performance. This observation is also confirmed by extensive simulations in Section 3.1. All the biological components in orthogonal directions have no impact. The nonlinear equations (5) and (6) have no closed form solution. We have to use some numerical algorithms to solve them. As expected, it can be easily checked from the numerical studies in Section 3 that $\cos(\theta)$ increases with α as well as the signal to noise ratio μ/σ, but decreases with λ_1.

Proposition 2 *Under Assumption 1, in the limit $N, M \to \infty$, with fixed $\alpha = M/N$, denote θ the angle between $\boldsymbol{\mu}$ and $\mathbf{w}$ solved from the soft-margin SVM algorithm (4), then $\cos\theta$ converges to ρ that is determined by the following three nonlinear equations*

$$\frac{1-\rho^2}{1+\lambda_1\rho^2} - \alpha q^2 \hat{\tau}^2 \int_{-\infty}^{z_c - q\hat{\tau}} Dz - \alpha \int_{z_c - q\hat{\tau}}^{z_c} Dz(z_c - z)^2 = 0, \tag{7}$$

$$2q - 1 - \alpha q\hat{\tau} \int_{-\infty}^{z_c - q\hat{\tau}} Dzz - \alpha \int_{z_c - q\hat{\tau}}^{z_c} Dz(z_c - z)z = 0, \tag{8}$$

$$\frac{\rho F}{\sqrt{1+\lambda_1\rho^2}} - \frac{\alpha q\hat{\tau}\mu}{\sigma} \int_{-\infty}^{z_c - q\hat{\tau}} Dz - \frac{\alpha\mu}{\sigma} \int_{z_c - q\hat{\tau}}^{z_c} Dz(z_c - z) = 0, \tag{9}$$

where

$$F = 1 - \alpha\lambda_1 q\hat{\tau} \int_{-\infty}^{z_c - q\hat{\tau}} Dzz - \alpha\lambda_1 \int_{z_c - q\hat{\tau}}^{z_c} Dz(z_c - z)z,$$

and

$$z_c = \frac{1/\sqrt{q_0} - \mu\rho}{\sigma\sqrt{1+\lambda_1\rho}}, \qquad \hat{\tau} = \frac{\sigma\tau}{\sqrt{q_0}\sqrt{1+\lambda_1\rho}}.$$

Therefore, given $\alpha, \lambda_1, \mu, \sigma$, and τ, equations (7), (8), (9) can be used to solve three unknown parameters ρ, q_0, and q. The nonlinear equations (7), (8), and (9) have no closed form solution. We have to use some numerical algorithms to solve them. Under Assumption 1, if we further assume that $\boldsymbol{\epsilon}$ in (1) follows a normal distribution, then the SVM test error is $\varepsilon = \Phi\left(-\frac{\rho}{\sqrt{1+\lambda_1\rho^2}}\frac{\mu}{\sigma}\right)$, where $\Phi(\cdot)$ is the cumulative distribution function of $N(0,1)$.

It is interesting to note that, as $\tau \to \infty$, the two equations (7) and (9) are equivalent to (5) and (6) respectively. Therefore, for large τ, the behavior of soft-margin SVM is the same as hard-margin SVM. Our simulation studies in Section 3.2 will also confirm this.

For a given dataset, α, λ_1, μ, and σ can be estimated, therefore Proposition 2 allows us to select optimal tuning parameter τ by studying the dependence of ρ on τ for fixed $\alpha, \lambda_1, \mu, \sigma$.

We now discuss how to estimate λ_1, μ, and σ from the data. To estimate the background noise level σ^2, we use a robust variance estimate based on the full matrix of data values (Liu et al., 2008); that is, for the full set of $M \times N$ entries of the original $M \times N$ data matrix $\mathbf{X}$, we calculate the robust estimate of scale, the median absolute deviation from the median (MAD), to estimate σ as

$$\hat{\sigma} \quad = \quad \frac{\mathrm{MAD}_{\mathbf{X}}}{\mathrm{MAD}_{N(0,1)}}. \tag{10}$$

Here $\mathrm{MAD}_{\mathbf{X}} = \mathrm{median}(|x_{ij} - \mathrm{median}(\mathbf{X})|)$ and $\mathrm{MAD}_{N(0,1)} = \mathrm{median}(|r_i - \mathrm{median}(\mathbf{r})|)$, where $\mathbf{r}$ is a MN-dimensional vector whose elements are i.i.d. samples from $N(0,1)$ distribution.

To estimate λ_1, we use the results from Baik and Silverstein (2006) which shows that in the limit of $M, N \to \infty$, with fixed $\alpha = M/N$, the sample eigenvalue $\tilde{\lambda}_1$ satisfies

$$\tilde{\lambda}_1 \quad \xrightarrow{a.s.} \quad \begin{cases} (\lambda_1+1)\left(1+\frac{1}{\alpha\lambda_1}\right)-1, & \text{for} \quad \lambda_1 > \sqrt{1/\alpha}, \\ (1+\sqrt{1/\alpha})^2 - 1, & \text{for} \quad \lambda_1 \le \sqrt{1/\alpha}. \end{cases} \tag{11}$$

Therefore, for any finite α, $\tilde{\lambda}_1$ is not a consistent estimator of λ_1. We use equation (11) to estimate λ_1 as

$$\hat{\lambda}_1 \quad = \quad \begin{cases} \frac{1}{2}\left(\tilde{\lambda}_1 - \frac{1}{\alpha} + \sqrt{\left(\tilde{\lambda}_1 - \frac{1}{\alpha}\right)^2 - \frac{4}{\alpha}}\right), & \text{for} \quad \tilde{\lambda}_1 > 1/\alpha + 2\sqrt{1/\alpha}, \\ \sqrt{1/\alpha}, & \text{for} \quad \tilde{\lambda}_1 \le 1/\alpha + 2\sqrt{1/\alpha}. \end{cases} \tag{12}$$

To estimate μ, let M_+, M_1 denote the sample sizes of Class $+1$ and Class -1 respectively. Define $\boldsymbol{\mu}_c = \bar{\mathbf{x}}_+ - \bar{\mathbf{x}}_-$, where $\bar{\mathbf{x}}_+$ and $\bar{\mathbf{x}}_-$ represent the sample means for Class $+1$ and Class -1 respectively. The following Proposition describes the relationship between $\boldsymbol{\mu}_c$ and μ.

Proposition 3 *Under Assumption 1, in the limit $N, M \to \infty$, with fixed $\alpha = M/N$, $r_+ = M_+/M$, and $r_- = M_-/M$, then $\|\boldsymbol{\mu}_c\|^2$ converges to*

$$4\mu^2 + \frac{\sigma^2}{\alpha r_+ r_-}. \tag{13}$$

Therefore, we estimate μ as

$$\hat{\mu} = \frac{1}{2}\sqrt{\|\hat{\boldsymbol{\mu}}_c\|^2 - \frac{\hat{\sigma}^2}{\alpha r_+ r_-}},$$

where $\hat{\sigma}$ is given from (10) and $\hat{\boldsymbol{\mu}}_c = \frac{1}{M_+}\sum_{i=1}^{M_+}\mathbf{x}_i - \frac{1}{M_-}\sum_{i=1}^{M_-}\mathbf{x}_i$ is the sample estimation of $\boldsymbol{\mu}_c$.

3. Numerical Results

3.1 Hard-margin SVM

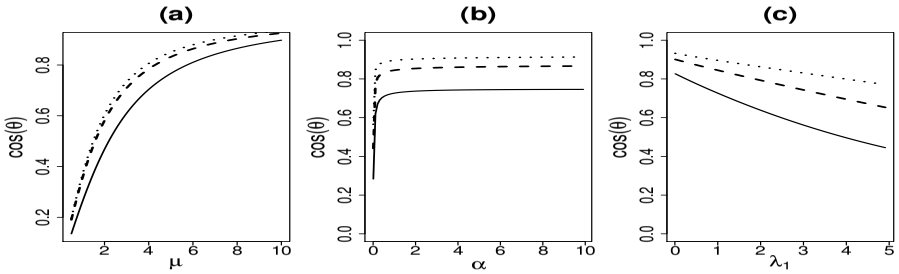


Figure 2: (a) Dependence of $\cos(\theta)$ on μ for fixed $\sigma = 1$, $\lambda_1 = 1$ and $\alpha = 0.1$ (solid), 0.5 (dashed), 1.5 (dotted); (b) Dependence of $\cos(\theta)$ on α for fixed $\lambda_1 = 1$ and $\mu = 3$ (solid), 5 (dashed), 7 (dotted); (c) Dependence of $\cos(\theta)$ on λ_1 for fixed $\alpha = 1$ and $\mu = 3$ (solid), 5 (dashed), 7 (dotted).

Figure 2 shows the dependence of $\cos(\theta)$ on the parameters μ, α, and λ_1 based on numerical solutions of equations (5) and (6). Here θ represents the angle between the directions of SVM separating hyperplane and Bayes optimal separating hyperplane. For spiked population model (1), the normal vector of Bayes optimal separating hyperplane lies in the direction of $\boldsymbol{\mu}$. Discrimination methods whose normal vector $\mathbf{w}/\|\mathbf{w}\|$ lies close to this direction should have good "generalization" properties, i.e., new data will be discriminated as well as possible. Figure 2(a) shows that, for fixed α and λ_1, the classification performance is improved as we increase the signal size μ. Figure 2(b) shows that, for fixed μ and λ_1, $\cos(\theta)$ increases with α, indicating that the classification performance is improved by adding more samples to the training data. For α<1/2, the increasing is faster; for α>2, the increasing becomes slower and saturated. This indicates that, for HDLSS situations, increasing training data can improve the prediction power dramatically; while for situations

when sample size is twice as big as the dimension, adding more samples can not gain too much power. Figure 2(c) shows that, for fixed μ and α, $\cos(\theta)$ decreases with λ_1 as expected.

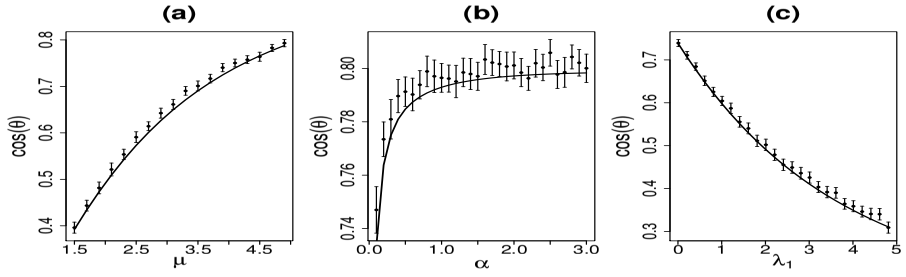


Figure 3: Comparison of analytical calculations with simulation experiments. The solid curves represent the theoretical results, the dots and bars represent the mean and standard error of the estimated $\cos\theta$ by applying SVM algorithm (3) to 100 simulated data sets for each parameter setting. In simulations, the dimension $N = 100$, the background noise $\sigma = 1$. The other parameters are: (a) $\alpha = 1, \lambda_1 = 2$; (b) $\mu = 5, \lambda_1 = 2$; (c) $\alpha = 1, \mu = 2$.

To examine the validity of our analysis and to determine the finite-size effect, Figure 3 provides the comparison with numerical simulations on finite size systems. Similar to Figure 2, we consider the dependence of $\cos(\theta)$ on three parameters μ, α and λ_1 in the plots in Figure 3 (a), (b), and (c) respectively. Here the dimension of the simulated data $N = 100$ and the data are generated according to Assumption 1 with ϵ_j follows i.i.d standard normal distribution. We repeat simulation 100 times for each parameter setting. The mean and standard errors over 100 replications are presented. From Figure 3, we can see that our analytical curves show fairly good agreement with the simulation experiment. Thus our analytical formulas (5) and (6) provide reliable estimates even for moderate system sizes. The benefit of these formulas is their computational ease. We also find that the simulation results for SVM estimators are independent of the choices of orthogonal components $\lambda_{m\geq 2}$ which further confirms that the analytical results described by Proposition 1 are correct.

3.2 Soft-margin SVM

Figure 4 shows the dependence of $\cos(\theta)$ on the parameters μ, α, and λ_1 based on the solution of nonlinear equations (7), (8), and (9) for fixed $\tau = 1$ and $\sigma = 1$. Similar to Figure 2, the $\cos\theta$ increases with μ and α but decreases with λ_1.

To study the the influence of the tuning parameter τ on the performance of the soft-margin SVM classification method (4). Figure 5 shows the dependence of $\cos(\theta)$ as function

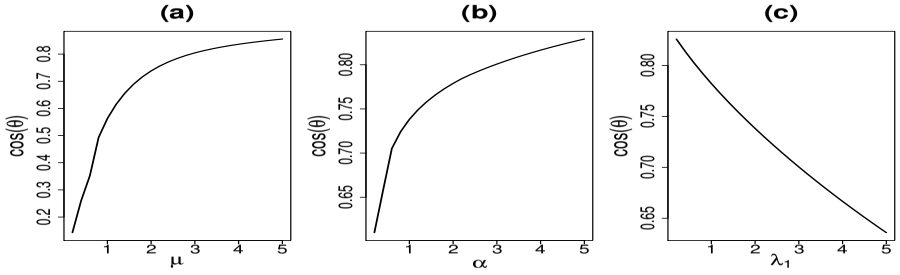


Figure 4: Dependence of $\cos(\theta)$ on μ, α, and λ_1 for $\tau = 1$ and $\sigma = 1$. (a) Dependence of $\cos(\theta)$ on μ for fixed $\lambda_1 = 2$ and $\alpha = 1$; (b) Dependence of $\cos(\theta)$ on α for fixed $\lambda_1 = 2$ and $\mu = 2$; (c) Dependence of $\cos(\theta)$ on λ_1 for fixed $\alpha = 1$ and $\mu = 2$.

of $\log\tau$ for fixed α, μ, λ_1, and σ. Both the analytical solution based on Proposition 2 and numerical experiment based on simulated finite dimensional data are provided and they excellently agree with each other. In simulation, we randomly generate a training set and a test set for the given parameter setting, the test error can be obtained by applying the classifier built from the training set to the test set. The results from the summary over 100 replications are given in Figure 5. From the upper panel, it is interesting to note that $\cos\theta$ reaches a maximum value as one decreases the tuning parameter τ to a threshold value. After that value, further decreasing τ cannot change $\cos\theta$. On the other hand, if we increase τ, $\cos\theta$ will approach the value determined by the hard-margin SVM method as shown by the dashed line in the upper panel of Figure 5. These observations are further confirmed by the dependence of test error ε on $\log\tau$ as shown in the lower panel. The test error reaches a minimum value if we decrease τ to the same threshold value as for $\cos\theta$. From equations (7), (8), and (9), it can be derived that the limiting value of $\cos\theta$ as $\tau \to 0$ is

$$\rho_c = \cos\theta_c = \sqrt{\frac{\alpha\left(\frac{\mu}{\sigma}\right)^2}{1+\alpha\left(\frac{\mu}{\sigma}\right)^2}} \tag{14}$$

which is independent of λ_1. This finding of λ_1 independence is also confirmed by numerical simulations with data on finite size systems. Therefore, if $\boldsymbol{\epsilon}$ in (1) follows a normal distribution, then the best test error we can achieve using the soft-margin SVM classification method (4) is $\Phi\left(-\frac{\rho_c}{\sqrt{1+\lambda_1\rho_c^2}}\frac{\mu}{\sigma}\right)$.

Koo et al. (2008) studied the asymptotic behavior of the coefficients of the linear SVM in the limit of $M \to \infty$ with N fixed. They established a Bahadur type representation of the coefficients and derived their asymptotic normality and statistical variability. Denote $\mathbf{w}^\star$ the minimizer of the population version of the SVM loss function. It was shown in Koo et al. (2008) that the SVM solution $\hat{\mathbf{w}}$ converges to $\mathbf{w}^\star$ which is in the same direction as $\boldsymbol{\mu}$

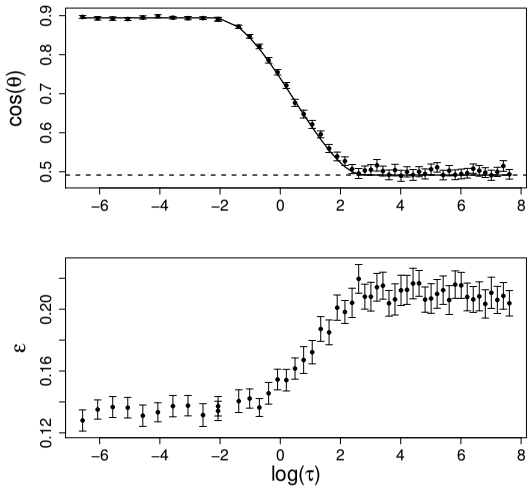


Figure 5: Upper panel: compare theoretical result with simulation experiment for the dependence of $\cos(\theta)$ on tuning parameter $\log(\tau)$ for fixed $\alpha = 1$, $\mu = 2$, $\sigma = 2$, and $\lambda_1 = 2$. The solid line is the theoretical curve, the dots and bars represent the mean and standard error based on 100 simulated data sets at each parameter setting. In simulation, the dimension $N = 100$. The dashed line represents the value based on the hard-margin SVM solution from equations (5) and (6). Lower panel: dependence of the test error ε on $\log(\tau)$ based on simulations.

under the spiked population setting (1). Therefore $\rho \to 1$ as $M \to \infty$ with N fixed. This can be confirmed in (14) by letting $\alpha \to \infty$ on the right hand side. On the other hand, if we let $N \to \infty$ with M fixed, from (14) we get $\rho_c \to 0$ if $\mu/\sqrt{N} \to 0$ and $\rho_c \to 1$ if $\mu/\sqrt{N} \to \infty$. This confirms the results of Hall et al. (2005) for HDLSS setting. Therefore, our asymptotic results are more general with both traditional and HDLSS asymptotics as special cases.

The analytical results in Figure 5 are based on the true values for α, λ_1, μ and σ which ultimately need to be estimated from the given data. In Figure 6 we provide the comparison between the results using the true values and the results using the estimated values for μ, α, λ_1 and σ. For each simulated data, we first estimate μ, α, λ_1 and σ and then use them to derive theoretical results. Figure 6 indicates that the influence of moderate estimation errors in the parameters is small.

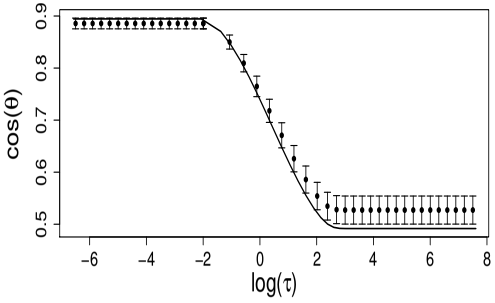


Figure 6: Comparison between the results using the true values and the results using the estimated values for parameters. Here the true parameter values are $\alpha = 1$, $\mu = 2$, $\sigma = 1$, and $\lambda_1 = 2$. The solid curve represents the results derived using the true values. The dots and bars represent the means and standard errors of the $\cos\theta$ values derived using the estimated parameters for 100 simulated data sets.

Although Figure 5 suggests that, for spiked population model, the best performance of SVM is achieved at the smallest value τ, in practice, using too tiny τ could cause difficulties in numerically solving the optimization problem. In order to provide a practical recommendation for the tuning parameter, we need to estimate the threshold value τ_c at which the limiting value $\cos\theta_c$ is almost achieved, i.e. the elbow point in Figure 5. More precisely, τ_c is defined as $\tau_c = \max\{\tau : \rho(\tau) = \rho_c\}$. In practice, we can compute τ_c by numerically finding the largest τ that can give $\cos\theta = \rho_c$ and use it as a guideline for choosing τ. Figure 7 displays the change of $\log\tau_c$ as functions of the parameters μ, α, λ_1. It is shown that $\log\tau_c$ decreases with all three parameters.

3.3 Check the model assumptions

The key assumptions for deriving the results in Propositions 1 and 2 are homoscedasticity (equal covariance) and spiked covariance condition (1). In this section, we use simulation to

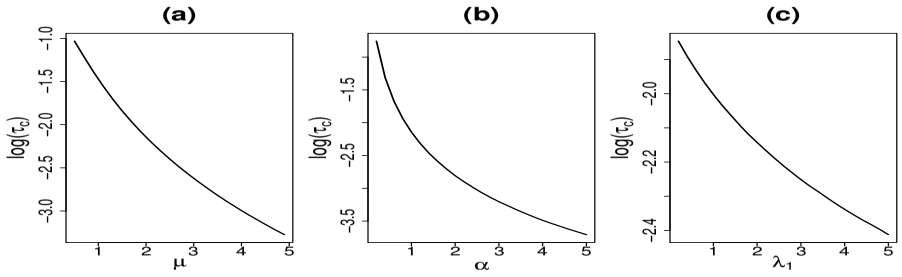


Figure 7: (a) Dependence of $\log(\tau_c)$ on μ for fixed $\lambda_1 = 2$ and $\alpha = 1$; (b) Dependence of $\log(\tau_c)$ on α for fixed $\lambda_1 = 2$ and $\mu = 2$; (c) Dependence of $\log(\tau_c)$ on λ_1 for fixed $\alpha = 1$ and $\mu = 2$.

study the validity of our method in situations where these assumptions are not true. Figure 8 is for situation where the two covariance matrices from the positive and negative classes are different. In simulation, we first generate M samples from $N(0, \sigma^2\mathbf{I}_p)$ distribution. Then $M/2$ of them are shifted by μ in x_1 direction to form the positive class and the remaining $M/2$ are shifted by $-\mu$ in x_1 direction to form the negative class. Both classes are further divided into two subclasses with sample size $M/4$ for each. In the positive class, the two subclasses are separated by shifting in x_2 direction by μ and $-\mu$ respectively. Similarly, the two subclasses in the negative class are separated by shifting in x_3 direction by μ and $-\mu$ respectively. The data generated in this way satisfies the spiked assumption but the two classes have different covariances. Figure 8 shows that the theoretical estimation and direct computation agree fairly well with each other. We have tried several different settings for μ, α and σ and got similar results. Therefore, our method is fairly robust to homoscedasticity as long as the spiked condition (1) holds for the covariance matrices of both classes.

Figures 9, 10, and 11 are for situations where the spiked assumption is violated. In simulation, we first generate M samples from $N(0, \mathbf{\Sigma})$ distribution. Then $M/2$ of them are shifted by μ in x_1 direction to form the positive class and the remaining $M/2$ are shifted by $-\mu$ in x_1 direction to form the negative class. The covariance matrix $\mathbf{\Sigma}$ is diagonal with the ith eigenvalue equal to $i\sigma^2$ for $i \leq K$ and σ^2 for $i > K$. The spiked condition requires $K \ll p$. If we increase K, the spiked condition can be violated. Figures 9, 10, and 11 show the results for $K = 0.1N$, $K = 0.3N$, and $K = 0.5N$ respectively. For situations where the number of uncommon eigenvalues K is less than 10% of the total number of variables N, our method can provide quite accurate estimation for $\cos\theta$ and also bigger $\cos\theta$ corresponds to smaller test error as illustrated in Figure 9. For situations where K is 30% of N, our method can still provide reasonable estimation for $\cos\theta$, but $\cos\theta$ cannot be used as a criterion for choosing τ because bigger $\cos\theta$ does not always correspond to

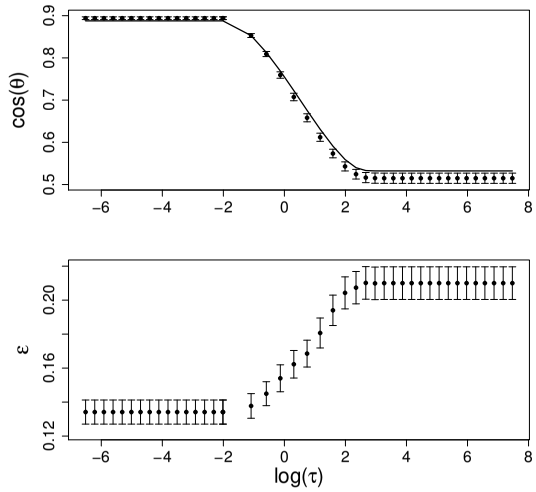


Figure 8: Comparison of theoretical prediction with direct computation for simulated data using parameters $\alpha = 1, \mu = 2, \sigma = 2$, and $\lambda_1 = 2$. In simulation, the covariances are different between the positive class and the negative class.

smaller test error as illustrated in Figure 10. For situations where K is 50% of N, our method cannot provide estimation for $\cos\theta$. Moreover, $\cos\theta$ and ε behavior in an opposite way, i.e. smaller $\cos\theta$ corresponds to smaller test error as illustrated in Figure 11.

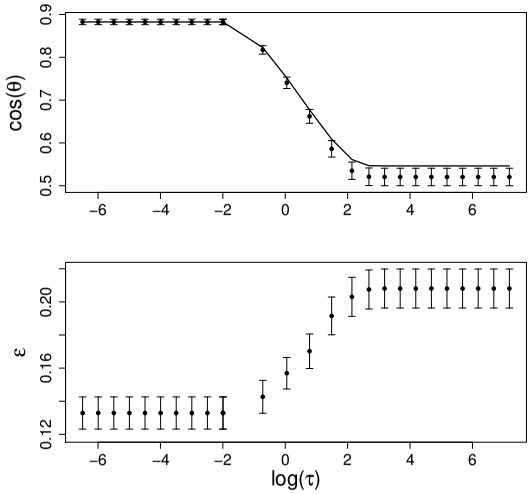


Figure 9: Comparison of theoretical prediction with direct computation for simulated data using parameters $\alpha = 1, \mu = 2, \sigma = 2$, and $\lambda_1 = 2$. In simulation, the number of the uncommon eigenvalues K is equal to 10% of the total number of variables N.

In summary, our simulations indicate that the proposed method depends on the spiked assumption but is not sensitive to the homoscedasticity violation. The spiked assumption is based on factor analysis which is one of the most useful tools for modeling common dependence among all the variables. In genetics, factor analysis modeling appeared to be useful tools to investigate the dependence structure in high-dimensional microarray data. It can fit the data with covariance matrix governed by linkage disequilibrium patterns (Rochat et al., 2007). For data set which cannot be modeled using spiked population, our results indicate that further exploring the data structure is useful for understanding the classification performance.

3.4 Real Data

We apply our methods to a breast cancer dataset from The Cancer Genome Atlas Research Network (Network, 2008) which include two subtypes: LumA and LumB. As in Liu et al. (2008), we filter the genes using the ratio of the sample standard deviation and sample mean of each gene. After gene filtering, the dataset contained 235 patients with 56 genes. Among the 235 samples, there are 154 LumA samples and 81 LumB samples.

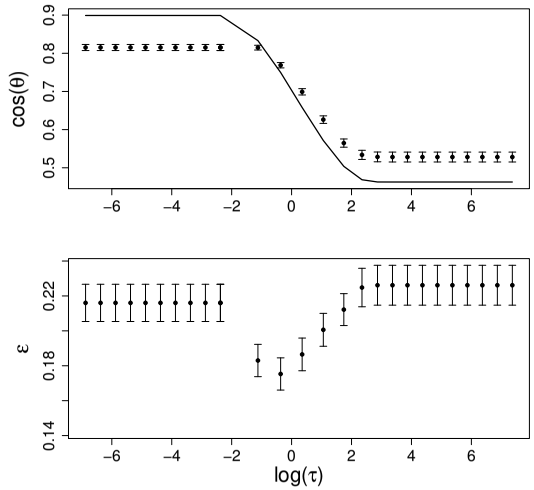


Figure 10: Comparison of theoretical prediction with direct computation for simulated data using parameters $\alpha = 1, \mu = 2, \sigma = 2$, and $\lambda_1 = 2$. In simulation, the number of the uncommon eigenvalues K is equal to 30% of the total number of variables N.

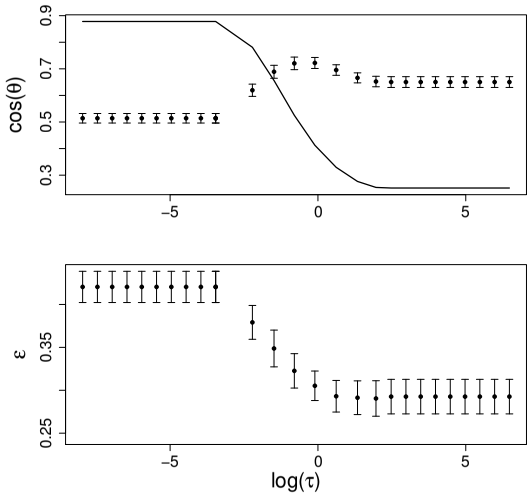


Figure 11: Comparison of theoretical prediction with direct computation for simulated data using parameters $\alpha = 1, \mu = 2, \sigma = 2$, and $\lambda_1 = 2$. In simulation, the number of the uncommon eigenvalues K is equal to 50% of the total number of variables N.

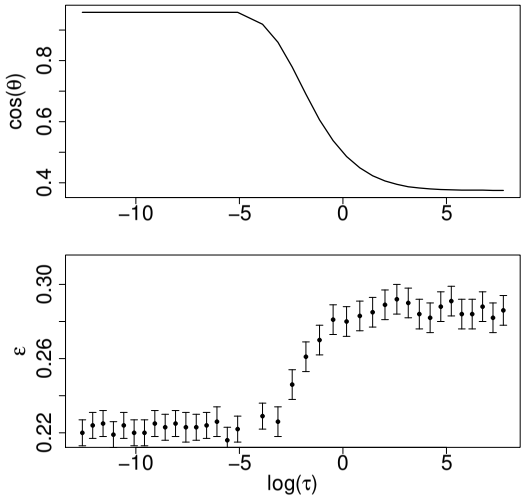


Figure 12: Upper panel: theoretical prediction of the dependence of $\cos(\theta)$ on tuning parameter $\log(\tau)$ based on the solutions from equations (7), (8) and (9) using parameters estimated from the breast cancer data. Lower panel: dependence of cross-validation error ε on tuning parameter $\log(\tau)$. The dots and bars represent the mean and standard error of the cross validation error based on 100 random splittings of the breast cancer data.

We consider LumA as Class +1 and LumB as Class -1. Assume the data are generated based on model (1), using the method discussed in Section 2, we obtain the following parameter estimations: $\hat{\mu} = 3.80$, $\hat{\sigma} = 2.32$, $\hat{\lambda}_1 = 4.06$, $\alpha = 4.20$, $N = 56$, $M = 235$, $M_+ = 154$, $M_- = 81$. The upper panel of Figure 12 shows the analytical curve for the dependence of $\cos\theta$ on τ. It shows that if we choose τ less than 6.19×10^{-3}, we can get the smallest angle. The lower panel of Figure 12 shows the dependence of the cross validation errors as a function of τ. The cross validation errors are computed by randomly splitting the data into two parts, 90% for training and 10% for test. The mean and standard error over 100 random splitting are reported in the lower panel of Figure 12. It shows that the cross validation error can achieve minimum value if τ is less than around 5×10^{-3}. The two results are consistent with each other and similar to the previous simulation results as shown in Figure 5. This indicates that model (1) is a reasonable assumption for this data set.

4. Conclusion

In this study, we examine the asymptotic behavior of SVM in the limit of $N, M \to \infty$ with fixed $\alpha = M/N$. We investigate the estimators of both the hard-margin SVM and the soft-margin SVM methods. Our focus is on the angle between the direction of the estimated separating hyperplane and the Bayes optimal separating hyperplane. Under spiked population model assumption, we analytically evaluate the relation between this angle and the SVM tuning parameter. On the basis of this finding, a new method of selecting tuning parameter is developed for analyzing high dimensional data which significantly reduces the computational cost. The analytical calculations are compared with numerical simulations on finite-size systems and the agreement between the numerical data and the analytical result is fairly good, and thus, our formulas are validated. Although the asymptotic results that we have obtained apply only to the spiked population model, they have shed a new light on the asymptotic behavior of SVM and can also improve the practical use of SVM in various aspects. For situations where the spiked model cannot be applied, one possible solution is to use the generalized spiked population model proposed in Bai and Yao (2012) to re-derive our results. This is one of our future research topics.

It is shown in Figure 1 that a lot of data points are piling up on the two boundaries. This is a phenomenon called data piling which has been studied in Marron et al. (2007) in more details. The reason is that the hinge loss function used in SVM is not continuous differentiable. The consequence of data piling is that the generalization performance is adversely affected. To overcome this problem, Marron et al. (2007) proposed a new classification method call Distance Weighted Discrimination (DWD) which does not have data piling problem. Simulation studies have shown that DWD typically yields better classification performance than SVM in high dimensions, but deeper theoretical evidence is strongly desired. It will be interesting to study the asymptotic property of DWD and compare it with SVM from a analytical point of view. This is another direction that we will pursue in future. The same technique can also be used in other popular classifier that currently heavily relies on cross validation. Examples include the hybrid of DWD and SVM proposed in Qiao and Zhang (2015) and the Large-Margin Unified Machines proposed in Liu et al. (2011).

Acknowledgments

The author would like to thank the Action Editor Professor Xiaotong Shen and three reviewers for their constructive comments and suggestions, which led to substantial improvements of the presentation of this paper.

References

Z. D. Bai. Methodologies in spectral analysis of large-dimensional random matrices, a review. *Statistica Sinica*, 9:611–677, 1999.

Zhidong Bai and Jianfeng Yao. On sample eigenvalues in a generalized spiked population model. *Journal of Multivariate Analysis*, 106:167 – 177, 2012.

Jinho Baik and Jack W. Silverstein. Eigenvalues of large sample covariance matrices of spiked population models. *Journal of Multivariate Analysis*, 97(6):1382 – 1408, 2006.

Peter Hall, J. Marron, and Amnon Neeman. Geometric representation of high dimension, low sample size data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(3):427–444, 2005.

D. C. Hoyle and M. Rattray. Principal-component-analysis eigenvalue spectra from data with symmetry-breaking structure. *Phys. Rev. E*, 69:026124, 2004.

D. C. Hoyle and M. Rattray. Statistical mechanics of learning multiple orthogonal signals: Asymptotic theory and fluctuation effects. *Phys. Rev. E*, 75:016101, 2007.

David C Hoyle. Statistical mechanics of learning orthogonal signals for general covariance models. *Journal of Statistical Mechanics: Theory and Experiment*, 2010(04):P04009, 2010.

Iain M. Johnstone. On the Distribution of the Largest Eigenvalue in Principal Components Analysis. *The Annals of Statistics*, 29(2):295–327, 2001.

S. K. Jung and J. S. Marron. PCA consistency in high dimension, low sample size context. *The Annals of Statistics*, 37:4104–4130, 2009.

Sungkyu Jung, Arusharka Sen, and J.S. Marron. Boundary behavior in high dimension, low sample size asymptotics of PCA. *Journal of Multivariate Analysis*, 109:190 – 203, 2012.

Ja-Yong Koo, Yoonkyung Lee, Yuwon Kim, and Changyi Park. A bahadur representation of the linear support vector machine. *Journal of Machine Learning Research*, 9:1343–1368, June 2008.

LAURENT LALOUX, PIERRE CIZEAU, MARC POTTERS, and JEAN-PHILIPPE BOUCHAUD. Random matrix theory and financial correlations. *International Journal of Theoretical and Applied Finance*, 03(03):391–397, 2000.

Yufeng Liu, David Neil Hayes, Andrew Nobel, and J. S Marron. Statistical significance of clustering for high-dimension, low-sample size data. *Journal of the American Statistical Association*, 103(483):1281–1293, 2008.

Yufeng Liu, Hao Helen Zhang, and Yichao Wu. Hard or soft classification? large-margin unified machines. *Journal of the American Statistical Association*, 106(493):166–177, 2011.

Zongming Ma. Sparse principal component analysis and iterative thresholding. *Ann. Statist.*, 41(2):772–801, 04 2013.

V. A. Marcenko and L. A. Pastur. Distribution of eigenvalues for some sets of random matrices. *Mathematics of the USSR-Sbornik*, 1(4):457–483, April 1967.

J. S. Marron, M. Todd, and J. Ahn. Distance-weighted discrimination. *Journal of the American Statistical Association*, 102:1267–1271, 2007.

Boaz Nadler. Finite sample approximation results for principal component analysis: A matrix perturbation approach. *The Annals of Statistics*, 36(6):2791–2817, 2008.

The Cancer Genome Atlas Research Network. Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature*, 455:1061–1068, 2008.

Debashis Paul. Asymptotics of sample eigenstructure for a large dimensional spiked covariance model. *Statistica Sinica*, 17:1617–1642, 2007.

Xingye Qiao and Lingsong Zhang. Flexible high-dimensional classification machines and their asymptotic properties. *Journal of Machine Learning Research*, 16:1547–1572, 2015.

R. H. Rochat, L. de las Fuentes, G. Stormo, V. G. Davila-Roman, and C. Charles Gu. A novel method combining linkage disequilibrium information and imputed functional knowledge for tagsnp selection. *Hum Hered*, 64(4):243–249, 2007.

Richard P. Sear and José A. Cuesta. Instabilities in complex mixtures with a large number of components. *Phys. Rev. Lett.*, 91:245701, Dec 2003.

E. Telatar. Capacity of multi-antenna Gaussian channels. *Eur. Trans. Telecomm. ETT*, 10 (6):585–596, November 1999.

Robert Tibshirani Trevor Hastie, Andreas Buja. Penalized discriminant analysis. *The Annals of Statistics*, 23(1):73–102, 1995.

V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, NY, 1995.

Journal of Machine Learning Research 18 (2017) 1-22 Submitted 11/16; Revised 4/17; Published 5/17

Distributed Semi-supervised Learning with Kernel Ridge Regression

Xiangyu Chang xiangyuchang@gmail.com
Center of Data Science and Information Quality
School of Management
Xi'an Jiaotong University, Xi'an, China
Shao-Bo Lin* sblin1983@gmail.com
Department of Statistics
Wenzhou University, Wenzhou, China
Ding-Xuan Zhou mazhou@cityu.edu.hk
Department of Mathematics
City University of Hong Kong
Tat Chee Avenue, Kowloon, Hong Kong, China

Editor: Andreas Christmann

Abstract

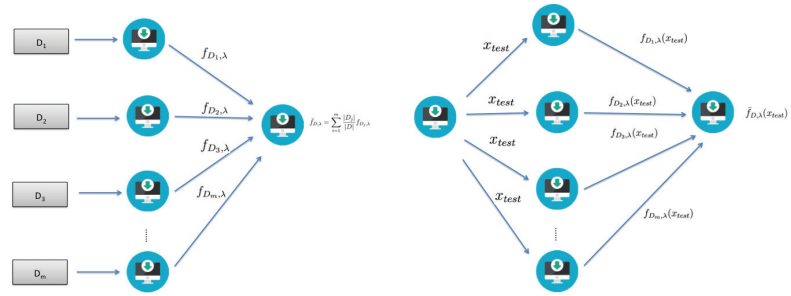
This paper provides error analysis for distributed semi-supervised learning with kernel ridge regression (DSKRR) based on a divide-and-conquer strategy. DSKRR applies kernel ridge regression (KRR) to data subsets that are distributively stored on multiple servers to produce individual output functions, and then takes a weighted average of the individual output functions as a final estimator. Using a novel error decomposition which divides the generalization error of DSKRR into the approximation error, sample error and distributed error, we find that the sample error and distributed error reflect the power and limitation of DSKRR, compared with KRR processing the whole data. Thus a small distributed error provides a large range of the number of data subsets to guarantee a small generalization error. Our results show that unlabeled data play important roles in reducing the distributed error and enlarging the number of data subsets in DSKRR. Our analysis also applies to the case when the regression function is out of the reproducing kernel Hilbert space. Numerical experiments including toy simulations and a music-prediction task are employed to demonstrate our theoretical statements and show the power of unlabeled data in distributed learning.

Keywords: learning theory, distributed learning, kernel ridge regression, semi-supervised learning, unlabeled data, error decomposition

1. Introduction

Data from practical applications in medicine, finance, business and other fields are often stored distributively across multiple servers (called local processors) and may not be shared for reasons of preserving privacy. This requires privacy-preserving machine learning algorithms to discover population-wide patterns of the data without revealing any individual's sensitive information. Distributed learning (Balcan et al., 2012; Yan et al., 2013; Li et al., 2015; Xie et al., 2016) provides a promising way to tackle privacy-preserving learning problems.

Let m be the number of local processors and $D_j = \{(x_{i,j}, y_{i,j})\}_{i=1}^{|D_j|}$ be the data subset stored in the j-th local processor with $1 \le j \le m$ and $D = \bigcup_{j=1}^m D_j$ be the disjoint union of $\{D_j\}_{j=1}^m$, where $|D_j|$ denotes the cardinality of D_j. Distributed learning firstly gets an estimator $f_{D_j,\lambda}$ on each local processor based on D_j and some (regularization) parameter λ, and then launches the individual estimators to a central processor to get a final estimator $\overline{f}_{D,\lambda}$. To get an estimation for a new query point x_{test}, due to the purpose of privacy-preserving, x_{test} is firstly transmitted to each local processor to get an estimation $f_{D_j,\lambda}(x_{test})$, then all $f_{D_j,\lambda}(x_{test})$ with $1 \le j \le m$ are communicated to the central processor to synthesize the final estimation $\overline{f}_{D,\lambda}(x_{test})$. Flows of training and testing for distributed learning are exhibited in Figure 1. Distributed learning can thus be a privacy-preserving strategy in the sense that except for a real number $f_{D_j,\lambda}(x_{test})$, the individual's data information in each local processor is unknown to each other. The aim of statistics and learning theory is to verify that $\overline{f}_{D,\lambda}(x_{test})$ is a good approximation of the unknown but definite output of x_{text}.



(a) Flow of training (b) Flow of testing

Figure 1: Training and testing of distributed learning for privacy-preserving

In this paper, we focus on the distributed kernel ridge regression (DKRR). Let $(\mathcal{H}_K, \|\cdot\|_K)$ be the reproduced kernel Hilbert space (RKHS) induced by a Mercer kernel K on a metric (input) space $\mathcal{X}$. In a standard setting (Mann et al., 2009; Zhang et al., 2013; Lin et al., 2016), DKRR is defined with a regularization parameter $\lambda > 0$ by

$$\overline{f}_{D,\lambda} = \sum_{j=1}^m \frac{|D_j|}{|D|} f_{D_j,\lambda}, \tag{1}$$

where

$$f_{D_j,\lambda} = \arg\min_{f \in \mathcal{H}_K} \left\{ \frac{1}{|D_j|} \sum_{(x,y)\in D_j} (f(x)-y)^2 + \lambda \|f\|_K^2 \right\}, \qquad j = 1, \ldots, m. \tag{2}$$

* Corresponding author

©2017 Chang, Lin and Zhou.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-601.html.

The optimal learning rate for DKRR was presented in (Zhang et al., 2015) under some eigenfunction assumptions which were removed in (Lin et al., 2016) by using a novel integral operator approach. In these existing results on rigorous analysis for DKRR, there are still two critical problems which greatly hinder applications of DKRR in practice. One is that the optimal learning rate is built upon a strict condition bounding the number of local processors, making DKRR infeasible for large m. The other is that the target function called regression function f_ρ is assumed to be in $\mathcal{H}_K$ to achieve the optimal learning rate, which is difficult to verify in practice.

The aim of the present paper is to consider distributed semi-supervised learning with kernel ridge regression (DSKRR) and demonstrate that using additional unlabeled data in a semi-supervised setting can overcome the aforementioned hurdles of DKRR. Let $D_j \cup \tilde{D}_j(x)$ be the subset of the data for semi-supervised learning which is stored on the j-th local processor, where $\tilde{D}_j(x) = \{x_1^j, \ldots, x_{|\tilde{D}_j|}^j\}$. Based on $D_j \cup \tilde{D}_j(x), j = 1, \ldots, m$, we construct a training set associated with $D_j \cup \tilde{D}_j(x)$ as

$$D_j^* = \{(x_i^*, y_i^*)\}_{i=1}^{|D_j^*|}$$

with

$$x_i^* = \begin{cases} x_i, & \text{if } (x_i, y_i) \in D_j, \\ \tilde{x}_i, & \text{if } \tilde{x}_i \in \tilde{D}_j(x), \end{cases} \qquad \text{and} \qquad y_i^* = \begin{cases} \frac{|D_j^*|}{|D_j|} y_i, & \text{if } (x_i, y_i) \in D_j, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

and denote $D^* = \bigcup_{j=1}^m D_j^*$. Define DSKRR by

$$\overline{f}_{D^*,\lambda} = \sum_{j=1}^m \frac{|D_j^*|}{|D^*|} f_{D_j^*,\lambda}. \tag{4}$$

We revise the standard integral operator framework (Smale and Zhou, 2007; Caponnetto and De Vito, 2007; Lin et al., 2016) to analyze the learning performance of algorithm (4). The main tool is a novel error decomposition that decomposes the generalization error of algorithm (4) into the approximation error, sample error and a new term called distributed error. The approximation error, which reflects the difference between a data-free limit of KRR and f_ρ, is standard. The sample error reflects the advantage of weighted averaging in (4) in the sense of scaling the generalization error of individual $f_{D_j^*,\lambda}$, $j = 1, 2, \ldots, m$, with an additional factor $\frac{|D_j^*|}{|D^*|}$. The distributed error describes the difference between the distributed algorithm (4) and KRR processing the whole data in one single processor via presenting a restriction to the number of local processors. We find that additional unlabeled data play crucial roles in deducing a small distributed error and thus relaxing heavily the restriction on m to achieve the optimal learning rate for DKRR. We also prove that DSKRR leads to satisfactory estimates for the sample error when $f_\rho \notin \mathcal{H}_K$, which is beyond the standard setting with $f_\rho \in \mathcal{H}_K$ in (Zhang et al., 2015; Lin et al., 2016). Experimental studies are carried out to verify our theoretical analysis.

2. Theoretical Assessments

The generalization ability of DSKRR will be analyzed in a standard learning theory framework, in which the sample in D is assumed to be independently drawn from ρ, a Borel

probability measure on $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$ with $\mathcal{Y} = \mathbb{R}$ and $\tilde{D}(x) = \cup_{j=1}^m \tilde{D}_j(x)$ from ρ_X, the marginal distribution of ρ. Our purpose is to estimate how the estimator $\overline{f}_{D^*,\lambda}$ based on D^* approximates the regression function $f_\rho(x) := \int_{\mathcal{Y}} y d\rho(y|x)$ with $\rho(\cdot|x)$ being the conditional distribution of ρ induced at $x \in \mathcal{X}$. Throughout this paper, we assume $|y| \le M$ almost surely for some constant $M > 0$ and $\mathcal{X}$ is compact, which implies $\|f_\rho\|_\infty \le M$ almost surely and $\kappa := \sqrt{\sup_{x \in \mathcal{X}} K(x,x)} < \infty$.

As convergence may be as slow as one wants without imposing any restriction on the distribution ρ (Györfy et al., 2002), we introduce the following regularity assumption on f_ρ. Let L_K be the integral operator on $\mathcal{H}_K$ (or $L^2_{\rho_X}$ with norm $\|\cdot\|_\rho\|$) defined by

$$L_K f = \int_{\mathcal{X}} K_x f(x) d\rho_X,$$

where K_x is the function $K(\cdot, x)$ in $\mathcal{H}_K$. The regularity condition in this paper is

$$f_\rho = L_K^r h_\rho, \quad \text{for some } r > 0 \text{ and } h_\rho \in L^2_{\rho_X}, \tag{5}$$

where L_K^r denotes the r-th power of $L_K : L^2_{\rho_X} \to L^2_{\rho_X}$, a compact and positive operator. This regularity condition is standard in learning theory and has been used in a large literature (Bauer et al., 2007; Caponnetto and De Vito, 2007; Smale and Zhou, 2007; Caponnetto and Yao, 2010; Shi et al., 2011; Guo et al., 2016; Hu et al., 2015; Lin and Zhou, 2016).

To derive fast learning rates, we should also present some restrictions on the capacity of $\mathcal{H}_K$. In this paper, we use the effective dimension $\mathcal{N}(\lambda)$ to measure the complexity of $\mathcal{H}_K$ with respect to ρ_X, which is defined to be the trace of the operator $(L_K + \lambda I)^{-1} L_K$, that is

$$\mathcal{N}(\lambda) = \mathrm{Tr}((\lambda I + L_K)^{-1} L_K), \qquad \lambda > 0.$$

We are in a position to present our main results, to be proved in Section 5. We first consider the traditional case of $f_\rho \in \mathcal{H}_K$ by imposing condition (5) with $r \ge 1/2$.

Theorem 1 *Assume $|y| \le M$ almost surely and that (5) holds with $1/2 \le r \le 1$. We have*

$$\begin{aligned} &\max\left\{E\left[\|\overline{f}_{D^*,\lambda} - f_\rho\|_\rho^2\right], \lambda E\left[\|\overline{f}_{D^*,\lambda} - f_\rho\|_K^2\right]\right\} \\ \le\ & C\left[\lambda^{2r} + \sum_{j=1}^m \frac{|D_j^*|}{|D^*|}\left(\frac{\mathcal{A}^2_{D_j^*,\lambda}}{\lambda} + 1\right)^2 \left(\lambda^{2r-1}\mathcal{A}^2_{D_j^*,\lambda} + \frac{|D_j^*|}{|D^*|}\mathcal{A}^2_{D_j,\lambda}\right)\right], \end{aligned} \tag{6}$$

where C is a constant independent of m, $|D_j|$, $|D_j^*|$, or λ and

$$\mathcal{A}_{D,\lambda} = \frac{1}{|D|\sqrt{\lambda}} + \frac{\sqrt{\mathcal{N}(\lambda)}}{\sqrt{|D|}}. \tag{7}$$

To obtain explicit learning rates, we quantify the increment of $\mathcal{N}(\lambda)$ with a parameter $0 < s \le 1$ and a constant $C_0 > 0$ as

$$\mathcal{N}(\lambda) \le C_0 \lambda^{-s}, \qquad \forall \lambda > 0. \tag{8}$$

Condition (8) with $s = 1$ is always satisfied by taking $C_0 = \mathrm{Tr}(L_K) \le \kappa^2$. When $0 < s < 1$, condition (8) is more general than the eigenvalue decaying assumption in the literature (Caponnetto and De Vito, 2007; Steinwart et al., 2009). Based on Theorem 1 and condition (8), we derive the following optimal learning rate of algorithm (4).

Corollary 2 *Assume $|y| \le M$ almost surely, (8) with $0 < s \le 1$ and (5) with $1/2 \le r \le 1$. If $\lambda = |D|^{-\frac{1}{2r+s}}$, $|D_1| = |D_2| = \cdots = |D_m|$, $|D_1^*| = |D_2^*| = \cdots = |D_m^*|$, and*

$$m \le \min\left\{|D|^{\frac{2r+2s-1}{2r+s}}, |D^*||D|^{\frac{-s-1}{2r+s}}\right\}, \tag{9}$$

then

$$E[\|\overline{f}_{D^*,\lambda} - f_\rho\|_\rho^2] = \mathcal{O}\left(|D|^{-\frac{2r}{2r+s}}\right).$$

When $|D^*| = |D|$, the training set D^* defined by (3) for DSKRR is the same as D for DKRR. In such a situation (without using any unlabeled data), the output function $\overline{f}_{D^*,\lambda}$ produced by DSKRR algorithm (4) coincides with $\overline{f}_{D,\lambda}$ generated by DKRR algorithm (1), and the optimal learning rates stated in Corollary 2 are the same as those in (Lin et al., 2016; Guo et al., 2016) achieved under the restriction $m \le |D|^{\frac{2r-1}{2r+s}}$ given by (9). In particular, for the special case of $r = 1/2$ (that is, $f_\rho \in \mathcal{H}_K$), the number of local processors $m = \mathcal{O}(1)$ does not increase as the sample size $|D|$ becomes large, which is very restrictive and limits the use of distributed learning.

Corollary 2 tells us that additional unlabeled data can be used to relax the above restriction on m. For the special case with $r = 1/2$ and $s \ge 1/2$, when $|D^*| = |D|^{1+\frac{1}{1+s}}$ with additional unlabel data of size $|D|^{1+\frac{1}{1+s}} - |D|$, Corollary 2 asserts that the output function $\overline{f}_{D^*,\lambda}$ produced by DSKRR algorithm (4) achieves the optimal learning rates under the restriction $m \le |D|^{\frac{1}{1+s}}$. This allows the number of local processors to increase as the sample size $|D|$ does. Thus our analysis demonstrates the usage of additional unlabeled data in distributed learning, which is the first purpose of this paper.

The second purpose of this paper is to extend the range of r in (5) from $r \ge 1/2$ for the standard setting with $f_\rho \in \mathcal{H}_K$ to $0 < r < 1/2$ for considering the situation $f_\rho \notin \mathcal{H}_K$.

Theorem 3 *Assume $|y| \le M$ almost surely and that (5) holds with $0 < r < 1/2$. Let $0 < \lambda \le 1$. We have*

$$E\left[\|\overline{f}_{D^*,\lambda} - f_\rho\|_\rho^2\right] \le \overline{C}\left[\lambda^{2r} + \sum_{j=1}^m \frac{|D_j^*|}{|D^*|}\left(\frac{\mathcal{A}^2_{D_j^*,\lambda}}{\lambda} + 1\right)^2\left(\lambda^{2r-1}\mathcal{A}^2_{D_j^*,\lambda} + \frac{|D_j^*|}{|D^*|}\mathcal{A}^2_{D_j,\lambda}\right)\right], \tag{10}$$

where $\overline{C}$ is a constant independent of m, $|D_j|$, $|D_j^*|$ or λ.

Theorem 3 yields the following optimal learning rate for algorithm (4) when $f_\rho \notin \mathcal{H}_K$, which has not been given in the literature of distributed learning (Zhang et al., 2015; Lin et al., 2016; Guo et al., 2016).

Corollary 4 *Assume $|y| \le M$ almost surely, (8) with $0 < s \le 1$ and (5) with $0 < r < 1/2$. If $r + s \ge 1/2$, $|D_1| = |D_2| = \cdots = |D_m|$, $|D_1^*| = |D_2^*| = \cdots = |D_m^*|$, $\lambda = |D|^{-\frac{1}{2r+s}}$, $|D^*| \ge |D|^{\frac{s+1}{2r+s}}$ and m satisfies (9), then*

$$E[\|\overline{f}_{D^*,\lambda} - f_\rho\|_\rho^2] = \mathcal{O}\left(|D|^{-\frac{2r}{2r+s}}\right).$$

3. Related Work and Discussion

As a state-of-the-art strategy to reduce the computational burden for some specified algorithms, distributed and parallel computation has triggered enormous research activities in the statistical and machine learning communities (Gillick et al., 2006). Distributed learning with ridge regression (Zhang et al., 2013), gradient descent algorithms (Shamir and Srebro, 2014), online learning (Zinkevich et al., 2010) and spectral algorithms (Guo et al., 2016) were proposed and their learning performances have been observed in many practical applications to be as good as that of a big processor which could handle the whole data, provided the number of servers m is not very large. Some theoretical bounds for m were presented in (Zhang et al., 2015; Lin et al., 2016; Guo et al., 2016; Blanchard and Mücke, 2016), asserting that the range of m depends on the regularity of f_ρ, which is difficult to verify in practice. This key problem makes users select only a small m or take m as a parameter in the learning process.

Compared with these results, there are two novelties of our results in this paper, though adding unlabeled data in the learning process causes additional computations. On one hand, if one takes m as a parameter in the learning process, the data should be re-divided again and again and thus it requires a large amount of communications. Our result avoids these re-division and communications in the sense that for a large range of m, if enough unlabeled data are given, optimal learning rates can be achieved. On the other hand, for some applications, data of small size (e.g. data in hospitals) are stored distributively across a great number of processors and cannot be shared each other for preserving privacy. The existing results (Zhang et al., 2015; Lin et al., 2016; Guo et al., 2016; Blanchard and Mücke, 2016) cannot tackle these problems in the sense that m is out of the range for the quantity to guarantee the optimal learning rate. The result in this paper presents a possibility to conquer this problem, provided there are additional unlabeled data.

We then compare our results with those in two closely related papers (Zhang et al., 2015; Lin et al., 2016). The seminal work (Zhang et al., 2015) considered the learning performance of algorithm (1) when $r = \frac{1}{2}$, i.e., $(f_\rho \in \mathcal{H}_K)$. Using a matrix decomposition approach, (Zhang et al., 2015) derived an optimal learning rate of order $\mathcal{O}(|D|^{-1/(s+1)})$ under the assumption that for some constants $k > 2$ and $A < \infty$, the normalized eigenfunctions $\{\phi_\ell\}_\ell$ of L_K in $L^2_{\rho_X}$ satisfy

$$\|\varphi_\ell\|^{2k}_{L^{2k}_{\rho_X}} = E\left[|\phi_\ell(x)|^{2k}\right] \le A^{2k}, \qquad \forall \ell \in \mathbb{N}. \tag{11}$$

Condition (11) was removed in (Lin et al., 2016), by using a novel integral operator approach based on a second order decomposition of difference of operator inverses. However, the analysis in (Lin et al., 2016) works only for $r > 1/2$. In our Corollary 2, we show that the optimal learning rate for DSKRR can be achieved for all $\frac{1}{2} \le r \le 1$ without assuming condition (11), provided additional unlabeled data are used. At the first glance, our approach in algorithm (4) incurs additional computation due to the unlabeled data. However, it is important for privacy-preserving learning when the data are stored in m (fixed) servers with $m > N^{\frac{2r-1}{2r+s}}$ and cannot be shared. Our result in Corollary 4 is new since optimal learning rates for DKRR when $f_\rho \notin \mathcal{H}_K$ have not been provided in the existing literate of distributed learning (Zhang et al., 2015; Lin et al., 2016; Guo et al., 2016).

Unlabeled data exist widely due to the expensive cost of label evaluation. Originally, unlabeled data are considered to be non-informative and often given up. However, with deeper understanding of semi-supervised learning, researchers recognize that unlabeled data can be useful in some special applications (Zhu and Goldberg, 2009) such as manifold learning (Belkin et al., 2006). In learning theory, it was found in (Caponnetto and Yao, 2010) that using unlabeled data can overcome the limitation that the optimal learning rate for KRR is only achievable for $f_\rho \in \mathcal{H}_K$. It was shown there that optimal learning rates for spectral algorithms might be achieved even when $f_\rho \notin \mathcal{H}_K$, provided enough unlabeled data were added in the training process. Similar results have been deduced for kernel-based conjugate gradient descent in (Blanchard and Krämer, 2010; Blanchard and Mücke, 2016). Results in the present paper show that unlabeled data also benefit distributed learning algorithms by allowing more local processors while achieving optimal learning rates. Furthermore, using some ideas from (Caponnetto and Yao, 2010), we succeed in conquering the bottleneck that optimal learning rates for distributed learning algorithms are achievable only when $f_\rho \in \mathcal{H}_K$. The numerical experiments to be reported in the last section motivated the last two authors (Lin and Zhou, 2016) to study the use of unlabeled data in distributed kernel-based gradient descent algorithms. Our results in the present paper also motivated a recent work (Guo et al., 2017) on error analysis for distributed manifold regularization algorithms.

4. Error Decomposition

The main tool in our analysis is a novel error decomposition for DSKRR. For this purpose, we introduce data-free limit and semi-supervised learning version of $f_{D_j,\lambda}$ as

$$f_\lambda = \arg\min_{f\in\mathcal{H}_K} \left\{ \int_{\mathcal{X}} (f(x) - f_\rho(x))^2 d\rho_X + \lambda \|f\|_K^2 \right\}$$

and

$$f^\diamond_{D_j,\lambda} = E^*[f_{D_j,\lambda}] := E[f_{D_j,\lambda}|D_j(x)].$$

The following proposition gives the error decomposition, whose proof is given at the end of this section.

Proposition 5 *Let $\overline{f}_{D,\lambda}$ be defined by (1). We have*

$$\frac{1}{2}E[\|\overline{f}_{D,\lambda} - f_\rho\|_\rho^2] \le \|f_\lambda - f_\rho\|_\rho^2 + \sum_{j=1}^m \frac{|D_j|^2}{|D|^2} E\left[\|f_{D_j,\lambda} - f_\lambda\|_\rho^2\right] + \sum_{j=1}^m \frac{|D_j|}{|D|} E\left[\left\|f^\diamond_{D_j,\lambda} - f_\lambda\right\|_\rho^2\right], \tag{12}$$

and if $f_\rho \in \mathcal{H}_K$,

$$\frac{1}{2}E[\|\overline{f}_{D,\lambda} - f_\rho\|_K^2] \le \|f_\lambda - f_\rho\|_K^2 + \sum_{j=1}^m \frac{|D_j|^2}{|D|^2} E\left[\|f_{D_j,\lambda} - f_\lambda\|_K^2\right] + \sum_{j=1}^m \frac{|D_j|}{|D|} E\left[\left\|f^\diamond_{D_j,\lambda} - f_\lambda\right\|_K^2\right]. \tag{13}$$

The three terms on the right-hand side of (12) (or (13)) are the approximation error, sample error and distributed error. The approximation error, independent of the sample,

describes the approximation capability of f_λ. The sample error connects the synthesized estimator (1) with the estimator (2). Compared with the sample error of the estimator (2), $E\left[\|f_{D_j,\lambda} - f_\lambda\|_\rho^2\right]$, there is an additional $\frac{|D_j|}{|D|}$ in our error decomposition, which reflects the power of the weighted averaging in (1) and shows the reason why the distributed algorithm (1) possesses similar learning performances as KRR processing the whole data D. Since $E^*[y_i] = f_\rho(x_i)$, it is easy to check that $f^\diamond_{D_j,\lambda}$ is the estimator derived from KRR with the noise-free data $\{(x_i, f_\rho(x_i))\}_{(x_i,y_i)\in D_j}$. This implies

$$f^\diamond_{D_j,\lambda} = \arg\min_{f\in\mathcal{H}_K} \left\{ \frac{1}{|D_j|} \sum_{(x,y)\in D_j} (f(x) - f_\rho(x))^2 + \lambda \|f\|_K^2 \right\}. \tag{14}$$

The distributed error presented in Proposition 5 measures the limitation of the distributed learning algorithm (1). Compared with the sample error $E\left[\|f_{D_j,\lambda} - f_\lambda\|_\rho^2\right]$ of estimator (2), the distributed error $E\left[\left\|f^\diamond_{D_j,\lambda} - f_\lambda\right\|_\rho^2\right]$ focuses on the noise-free data and therefore is smaller than $E\left[\|f_{D_j,\lambda} - f_\lambda\|_\rho^2\right]$. This makes algorithm (1) possess similar learning rates as that of KRR with the whole data D. However, since there are only $|D_j|$ samples, it is impossible to get upper bounds asymptomatically as $|D|^{-2r/(2r+s)}$ for $E\left[\left\|f^\diamond_{D_j,\lambda} - f_\lambda\right\|_\rho^2\right]$ when m is large. Thus, a restriction on m to guarantee the optimal learning rate is necessary.

To deduce a wide range of m, we need a tight bound for the distributed error. Noting that $f^\diamond_{D_j,\lambda}$ is independent of the sample outputs, it motivates us to employ the unlabeled part $\tilde{D}_j(x), j = 1, \ldots, m$ of the data set D_j^* in designing the algorithm (4). In the following, we combine the traditional integral operator approach (Smale and Zhou, 2004, 2005, 2007) with a recently developed second order decomposition of operator inverses (Lin et al., 2016; Guo et al., 2016) to derive a tight estimate for the distributed error of algorithm (4). Denote the sampling operator $S_D : \mathcal{H}_K \to \mathbb{R}^{|D|}$ (or $L^2_{\rho_X} \to \mathbb{R}^{|D|}$) by

$$S_D f := \{f(x_i)\}_{(x_i,y_i)\in D}.$$

Its adjoint $S_D^T : \mathbb{R}^{|D|} \to \mathcal{H}_K$ (or $\mathbb{R}^{|D|} \to L^2_{\rho_X}$) is given by

$$S_D^T \mathbf{c} := \frac{1}{|D|} \sum_{(x_i,y_i)\in D} c_i K_{x_i}, \qquad \mathbf{c} \in \mathbb{R}^{|D|}.$$

Let $L_{K,D}$ be the data-dependent approximation of L_K defined by

$$L_{K,D} f = S_D^T S_D f = \frac{1}{|D|} \sum_{(x,y)\in D} f(x) K_x.$$

Then it is easy to check (Smale and Zhou, 2007; Caponnetto and De Vito, 2007) that

$$f_{D,\lambda} = (L_{K,D} + \lambda I)^{-1} S_D^T y_D, \qquad f_\lambda = (L_K + \lambda I)^{-1} L_K f_\rho \tag{15}$$

and

$$f^\diamond_{D,\lambda} = (L_{K,D} + \lambda I)^{-1} L_{K,D} f_\rho, \tag{16}$$

where I is the identity operator and $y_D := (y_1, \ldots, y_{|D|})^T$. The following proposition presents error estimates for the sample and distributed errors.

Proposition 6 *Let $f_{D_j,\lambda}$, f_λ and $f^\diamond_{D_j,\lambda}$ be defined by (15) and (16). We have*

$$\max\{\|f_{D_j,\lambda}-f_\lambda\|_\rho, \sqrt{\lambda}\|f_{D_j,\lambda}-f_\lambda\|_K\} \le \mathcal{Q}^2_{D_j,\lambda}(\mathcal{P}_{D_j,\lambda}+\mathcal{S}_{D_j,\lambda}\|f_\lambda\|_K), \tag{17}$$

and

$$\max\{\|f^\diamond_{D_j,\lambda}-f_\lambda\|_\rho, \sqrt{\lambda}\|f^\diamond_{D_j,\lambda}-f_\lambda\|_K\} \le \mathcal{Q}^2_{D_j,\lambda}\mathcal{R}_{D_j,\lambda,f_\lambda-f_\rho}, \tag{18}$$

where

$$\mathcal{P}_{D,\lambda} := \left\|(L_K+\lambda I)^{-1/2}(L_Kf_\rho - S_D^Ty_D)\right\|_K,$$
$$\mathcal{Q}_{D,\lambda} := \|(L_K+\lambda I)^{1/2}(L_{K,D}+\lambda I)^{-1/2}\|,$$
$$\mathcal{S}_{D,\lambda} := \left\|(L_K+\lambda I)^{-1/2}(L_K-L_{K,D})\right\|,$$

and with a bounded measurable function g on $\mathcal{X}$,

$$\mathcal{R}_{D,\lambda,g} := \left\|(L_K+\lambda I)^{-1/2}\left(\int_{\mathcal{X}} g(x)K_xd\rho_X - \frac{1}{|D|}\sum_{(x,y)\in D} g(x)K_x\right)\right\|_K.$$

Proof Since

$$\begin{aligned} & f_{D_j,\lambda}-f_\lambda = (L_{K,D_j}+\lambda I)^{-1}S^T_{D_j}y_{D_j} - (L_K+\lambda I)^{-1}L_Kf_\rho \\ = \;& (L_{K,D_j}+\lambda I)^{-1}(S^T_{D_j}y_{D_j}-L_Kf_\rho) + [(L_{K,D_j}+\lambda I)^{-1}-(L_K+\lambda I)^{-1}]L_Kf_\rho, \end{aligned}$$

and

$$\|f\|_\rho = \|L_K^{1/2}f\|_K = \|L_K^{1/2}(L_K+\lambda I)^{-1/2}(L_K+\lambda I)^{1/2}f\|_K \le \|(L_K+\lambda I)^{1/2}f\|_K$$

for any $f\in L^2_{\rho_X}$, it follows from $A^{-1}-B^{-1}=A^{-1}(B-A)B^{-1}$ for positive operators A,B that

$$\begin{aligned} & \max\left\{\|f_{D_j,\lambda}-f_\lambda\|_\rho, \sqrt{\lambda}\|f_{D_j,\lambda}-f_\lambda\|_K\right\} \\ \le \;& \|(L_K+\lambda I)^{1/2}(L_{K,D_j}+\lambda I)^{-1}(S^T_{D_j}y_{D_j}-L_Kf_\rho)\|_K \\ + \;& \|(L_K+\lambda I)^{1/2}[L_{K,D_j}+\lambda I)^{-1}-(L_K+\lambda I)^{-1}]L_Kf_\rho\|_K \\ \le \;& \mathcal{Q}_{D_j,\lambda}\|(L_{K,D_j}+\lambda I)^{-1/2}(S^T_{D_j}y_{D_j}-L_Kf_\rho)\|_K \\ + \;& \|(L_K+\lambda I)^{1/2}(L_{K,D_j}+\lambda I)^{-1}(L_K-L_{K,D_j})(L_K+\lambda I)^{-1}L_Kf_\rho\|_K \\ \le \;& \mathcal{Q}^2_{D_j,\lambda}\|(L_K+\lambda I)^{-1/2}(S^T_{D_j}y_{D_j}-L_Kf_\rho)\|_K \\ + \;& \mathcal{Q}^2_{D_j,\lambda}\|(L_K+\lambda I)^{-1/2}(L_K-L_{K,D_j})\|\|(L_K+\lambda I)^{-1}L_Kf_\rho\|_K \\ \le \;& \mathcal{Q}^2_{D_j,\lambda}(\mathcal{P}_{D_j,\lambda}+\mathcal{S}_{D_j,\lambda}\|f_\lambda\|_K). \end{aligned}$$

This completes the proof of (17). Now we turn to prove (18). Due to (15) and (16), we have

$$\begin{aligned} & f^\diamond_{D_j,\lambda}-f_\lambda = (L_{K,D_j}+\lambda I)^{-1}L_{K,D_j}f_\rho - (L_K+\lambda I)^{-1}L_Kf_\rho \\ = \;& (L_{K,D_j}+\lambda I)^{-1}(L_{K,D_j}-L_K)f_\rho + [L_{K,D_j}+\lambda I)^{-1}-(L_K+\lambda I)^{-1}]L_Kf_\rho \\ = \;& (L_{K,D_j}+\lambda I)^{-1}(L_{K,D_j}-L_K)f_\rho + (L_{K,D_j}+\lambda I)^{-1}[L_K-L_{K,D_j}](L_K+\lambda I)^{-1}L_Kf_\rho \\ = \;& -(L_{K,D_j}+\lambda I)^{-1}[L_K-L_{K,D_j}](f_\rho-f_\lambda). \end{aligned}$$

Then

$$\begin{aligned} & \max\{\|f^\diamond_{D_j,\lambda}-f_\lambda\|_\rho, \sqrt{\lambda}\|f^\diamond_{D_j,\lambda}-f_\lambda\|_K\} \\ \le \;& \|(L_K+\lambda I)^{1/2}(L_{K,D_j}+\lambda I)^{-1}[L_K-L_{K,D_j}](f_\lambda-f_\rho)\|_K \\ \le \;& \mathcal{Q}_{D_j,\lambda}\|(L_{K,D_j}+\lambda I)^{-1/2}(L_{K,D_j}-L_K)(f_\lambda-f_\rho)\|_K \le \mathcal{Q}^2_{D_j,\lambda}\mathcal{R}_{D_j,\lambda,f_\lambda-f_\rho}. \end{aligned}$$

This completes the proof of Proposition 6. ■

If $f_\rho\in\mathcal{H}_K$, we have

$$\mathcal{R}_{D_j,\lambda,f_\lambda-f_\rho} = \|(L_K+\lambda I)^{-1/2}(L_{K,D_j}-L_K)(f_\lambda-f_\rho)\|_K \le \mathcal{S}_{D_j,\lambda}\|f_\lambda-f_\rho\|_K.$$

This implies that the distributed error can be bounded by $\mathcal{Q}^2_{D_j,\lambda}\mathcal{S}_{D_j,\lambda}\|f_\lambda-f_\rho\|_K$. Comparing with the sample error estimate (17) for an individual local processor, there is an additional term $\|f_\lambda-f_\rho\|_K$ in the distributed error estimate, since $\mathcal{P}_{D_j,\lambda}$, and $\mathcal{S}_{D_j,\lambda}$ are asymptotically equal due to Lemma 9 in Section 5. This together with Lemma 8 below shows that the distributed error estimate is essentially smaller than the sample error estimate for local processors under (5) with $r>1/2$, and presents the reason why DSKRR performs similarly as KRR on the whole data set D, provided m is not so large.

Recall the definitions of $\mathcal{Q}_{D_j,\lambda}$, $\mathcal{R}_{D_j,\lambda,f_\lambda-f_\rho}$ and $\mathcal{S}_{D_j,\lambda}$. These three quantities are independent of the outputs. Thus, the distributed error estimate decreases when additional unlabeled data are given. This explains why unlabeled data can enlarge the range of m to guarantee the optimal learning rates for DSKRR. On the other hand, it follows from the definition of $\mathcal{P}_{D,\lambda}$ that the sample error estimate depends heavily on the labels corresponding to unlabeled data. But (3) implies that for each fixed $1\le j\le m$,

$$S^T_{D^*_j}y_{D^*_j} = \frac{1}{|D^*_j|}\sum_{(x,y)\in D^*_j} y^*K_x = \frac{1}{|D^*_j|}\frac{|D^*_j|}{|D_j|}\sum_{(x,y)\in D_j} yK_x = S^T_{D_j}y_{D_j}.$$

Then $\mathcal{P}_{D^*_j,\lambda}=\mathcal{P}_{D_j,\lambda}$ for each $1\le j\le m$, which implies that the sample error estimate does not increase when the unlabeled data are added and shows the necessity of re-weighting of y in our definition in (3).

Thus, the unlabeled data in algorithm (4) can reduce the distributed error estimate without increasing the sample error estimate. The following proposition which can be deduced directly from Propositions 5 and 6, shows the detailed error decomposition for algorithm (4).

Proposition 7 *Let $\overline{f}_{D^*,\lambda}$ be defined by algorithm (4). If condition (5) holds with $0<r\le 1$, then for $1/2\le r\le 1$,*

$$\begin{aligned} & \frac{1}{2}\max\left\{E[\|\overline{f}_{D^*,\lambda}-f_\rho\|^2_\rho], \lambda E[\|\overline{f}_{D^*,\lambda}-f_\rho\|^2_K]\right\} \\ \le \;& \max\{\|f_\lambda-f_\rho\|^2_\rho, \lambda\|f_\lambda-f_\rho\|^2_K\} + \sum_{j=1}^m \frac{|D^*_j|^2}{|D^*|^2}E\left[\mathcal{Q}^4_{D^*_j,\lambda}(\mathcal{P}_{D_j,\lambda}+\mathcal{S}_{D^*_j,\lambda}\|f_\lambda\|_K)^2\right] \\ & + \sum_{j=1}^m \frac{|D^*_j|}{|D^*|}E\left[\mathcal{Q}^4_{D^*_j,\lambda}\mathcal{S}^2_{D^*_j,\lambda}\|f_\lambda-f_\rho\|^2_K\right] \end{aligned} \tag{19}$$

while for $0 < r < 1/2$,

$$E[\|\overline{f}_{D^*,\lambda} - f_\rho\|_\rho^2] \leq \|f_\lambda - f_\rho\|_\rho^2 + \sum_{j=1}^m \frac{|D_j^*|^2}{|D^*|^2} E\left[\mathcal{Q}_{D_j^*,\lambda}^4 (\mathcal{P}_{D_j,\lambda} + \mathcal{S}_{D_j^*,\lambda}\|f_\lambda\|_K)^2\right] + \sum_{j=1}^m \frac{|D_j^*|}{|D^*|} E\left[\mathcal{Q}_{D_j^*,\lambda}^4 \mathcal{R}_{D_j^*,\lambda,f_\lambda - f_\rho}^2\right]. \quad (20)$$

We end this section by proving Proposition 5.

Proof of Proposition 5 For the sake of brevity, we only prove (12), since (13) can be derived by using the same method. Due to the triangle inequality and the elementary inequality $(a+b)^2 \leq 2a^2 + 2b^2$ for $a, b > 0$, we have

$$E[\|\overline{f}_{D,\lambda} - f_\rho\|_\rho^2] \leq 2\|f_\lambda - f_\rho\|_\rho^2 + 2E[\|\overline{f}_{D,\lambda} - f_\lambda\|_\rho^2]. \quad (21)$$

It follows from $\sum_{j=1}^m \frac{|D_j|}{|D|} = 1$ that

$$\|\overline{f}_{D,\lambda} - f_\lambda\|_\rho^2 = \left\|\sum_{j=1}^m \frac{|D_j|}{|D|}(f_{D_j,\lambda} - f_\lambda)\right\|_\rho^2$$

$$= \sum_{j=1}^m \frac{|D_j|^2}{|D|^2}\|f_{D_j,\lambda} - f_\lambda\|_\rho^2 + \sum_{j=1}^m \frac{|D_j|}{|D|}\left\langle f_{D_j,\lambda} - f_\lambda, \sum_{k\neq j}\frac{|D_k|}{|D|}(f_{D_k,\lambda} - f_\lambda)\right\rangle_\rho.$$

Taking expectations gives

$$E\left[\|\overline{f}_{D,\lambda} - f_\lambda\|_\rho^2\right] = \sum_{j=1}^m \frac{|D_j|^2}{|D|^2} E\left[\|f_{D_j,\lambda} - f_\lambda\|_\rho^2\right] + \sum_{j=1}^m \frac{|D_j|}{|D|}\left\langle E[f_{D_j,\lambda}] - f_\lambda, E[\overline{f}_{D,\lambda}] - f_\lambda - \frac{|D_j|}{|D|}\left(E[f_{D_j,\lambda}] - f_\lambda\right)\right\rangle_\rho.$$

But

$$\sum_{j=1}^m \frac{|D_j|}{|D|}\left\langle E_{D_j}[f_{D_j,\lambda}] - f_\lambda, E[\overline{f}_{D,\lambda}] - f_\lambda\right\rangle_\rho = \left\langle E[\overline{f}_{D,\lambda}] - f_\lambda, E[\overline{f}_{D,\lambda}] - f_\lambda\right\rangle_\rho = \left\|E[\overline{f}_{D,\lambda}] - f_\lambda\right\|_\rho^2.$$

We see that $E\left[\|\overline{f}_{D,\lambda} - f_\lambda\|_\rho^2\right]$ equals

$$\sum_{j=1}^m \frac{|D_j|^2}{|D|^2} E\left[\|f_{D_j,\lambda} - f_\lambda\|_\rho^2\right] - \sum_{j=1}^m \frac{|D_j|^2}{|D|^2}\left\|E[f_{D_j,\lambda}] - f_\lambda\right\|_\rho^2 + \left\|E[\overline{f}_{D,\lambda}] - f_\lambda\right\|_\rho^2.$$

Furthermore, by the Schwarz inequality and $\sum_{j=1}^m \frac{|D_j|}{|D|} = 1$, we have

$$\|E[\overline{f}_{D,\lambda}] - f_\lambda\|_\rho^2 = \left\|\sum_{j=1}^m \frac{|D_j|}{|D|}\left(E[f_{D_j,\lambda}] - f_\lambda\right)\right\|_\rho^2 \leq \sum_{j=1}^m \frac{|D_j|}{|D|}\left\|E[f_{D_j,\lambda}] - f_\lambda\right\|_\rho^2.$$

Then

$$E\left[\|\overline{f}_{D,\lambda}] - f_\lambda\|_\rho^2\right] \leq \sum_{j=1}^m \frac{|D_j|^2}{|D|^2} E\left[\|f_{D_j,\lambda} - f_\lambda\|_\rho^2\right] + \sum_{j=1}^m \frac{|D_j|}{|D|}\left\|E[f_{D_j,\lambda}] - f_\lambda\right\|_\rho^2.$$

Inserting the above inequalities into (21), we find

$$\frac{1}{2}E[\|\overline{f}_{D,\lambda} - f_\rho\|_\rho^2] \leq \|f_\lambda - f_\rho\|_\rho^2 + \sum_{j=1}^m \frac{|D_j|^2}{|D|^2} E\left[\|f_{D_j,\lambda} - f_\lambda\|_\rho^2\right] + \sum_{j=1}^m \frac{|D_j|}{|D|}\left\|E[f_{D_j,\lambda}] - f_\lambda\right\|_\rho^2.$$

According to Jensen's inequality, we obtain

$$\left\|E[f_{D_j,\lambda}] - f_\lambda\right\|_\rho^2 \leq E\left[\left\|f_{D_j,\lambda}^\diamond - f_\lambda\right\|_\rho^2\right],$$

which implies

$$\frac{1}{2}E[\|\overline{f}_{D,\lambda} - f_\rho\|_\rho^2] \leq \|f_\lambda - f_\rho\|_\rho^2 + \sum_{j=1}^m \frac{|D_j|^2}{|D|^2} E\left[\|f_{D_j,\lambda} - f_\lambda\|_\rho^2\right] + \sum_{j=1}^m \frac{|D_j|}{|D|} E\left[\left\|f_{D_j,\lambda}^\diamond - f_\lambda\right\|_\rho^2\right].$$

This completes the proof of Proposition 5. ■

5. Proofs

According to Proposition 7, to prove Theorems 1 and 3, we only need to bound $\|f_\lambda - f_\rho\|_\rho$, $\|f_\lambda - f_\rho\|_K$, $\mathcal{Q}_{D_j^*,\lambda}$, $\mathcal{P}_{D_j,\lambda}$, $\mathcal{R}_{D_j^*,\lambda,g}$ and $\mathcal{S}_{D_j^*,\lambda}$. The following two lemmas present bounds for these quantities. The first one can be found in (Smale and Zhou, 2007), which estimates the approximation error of algorithm (4).

Lemma 8 *Assume (5) with* $0 < r \leq 1$. *There holds*

$$\|f_\lambda - f_\rho\|_\rho \leq \lambda^r \|h_\rho\|_\rho. \quad (22)$$

Furthermore, if $1/2 \leq r \leq 1$, *then we have*

$$\|f_\lambda - f_\rho\|_K \leq \lambda^{r-1/2}\|h_\rho\|_\rho. \quad (23)$$

The second lemma presents bounds for $\mathcal{Q}_{D_j^*,\lambda}$, $\mathcal{P}_{D_j,\lambda}$, $\mathcal{R}_{D_j^*,\lambda,g}$ and $\mathcal{S}_{D_j^*,\lambda}$. In particular, (24) was proved in (Guo et al., 2016), (25) and (27) were given in (Lin et al., 2016) and (26) can be found in (Caponnetto and De Vito, 2007). Recall the quantity $\mathcal{A}_{D,\lambda}$ defined by (7).

Lemma 9 *Let* D *be a sample drawn independently according to* ρ *and* $0 < \delta < 1$. *If* $|y| \leq M$ *almost surely, then each of the following estimates holds with confidence at least* $1 - \delta$,

$$\mathcal{Q}_{D,\lambda}^2 \leq 2\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}\log\frac{2}{\delta}}{\sqrt{\lambda}}\right)^2 + 2, \quad (24)$$

$$\mathcal{S}_{D,\lambda} \leq 2(\kappa^2+\kappa)\mathcal{A}_{D,\lambda}\log(2/\delta), \quad (25)$$

$$\mathcal{P}_{D,\lambda} \leq 2M(\kappa+1)\mathcal{A}_{D,\lambda}\log(2/\delta), \quad (26)$$

$$\mathcal{R}_{D,\lambda,g} \leq 2\|g\|_\infty(\kappa+1)\mathcal{A}_{D,\lambda}\log(2/\delta). \quad (27)$$

We can now use Proposition 7 and the above lemmas to prove our main results.

Proof of Theorem 1 As $r \geq 1/2$, Lemma 8 implies that

$$\max\left\{\|f_\lambda - f_\rho\|_\rho, \sqrt{\lambda}\|f_\lambda - f_\rho\|_K\right\} \leq \lambda^r \|h_\rho\|_\rho. \tag{28}$$

For an arbitrary fixed $j \in \{1, \ldots, m\}$, it follows from Lemma 9 that there exist three subsets $\mathcal{Z}_{1,\delta}^{|D_j^*|}$, $\mathcal{Z}_{2,\delta}^{|D_j^*|}$ and $\mathcal{Z}_{3,\delta}^{|D_j^*|}$ of $\mathcal{Z}^{|D_j^*|}$ with measures at least $1-\delta/3$ such that for $D_j^* \in \mathcal{Z}_{1,\delta}^{|D_j^*|} \cap \mathcal{Z}_{2,\delta}^{|D_j^*|} \cap \mathcal{Z}_{3,\delta}^{|D_j^*|}$ there holds

$$\begin{aligned}
\mathcal{Q}^2_{D_j^*,\lambda} &\leq 2\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}\log\frac{6}{\delta}}{\sqrt{\lambda}}\right)^2 + 2,\\
\mathcal{S}_{D_j^*,\lambda} &\leq 2(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}\log\frac{6}{\delta},\\
\mathcal{P}_{D_j,\lambda} &\leq 2M(\kappa+1)\mathcal{A}_{D_j,\lambda}\log\frac{6}{\delta}.
\end{aligned}$$

This together with

$$\|f_\lambda\|_K = \|(L_K+\lambda I)^{-1}L_K f_\rho\|_K \leq \|f_\rho\|_K \leq \|L_K^{r-1/2}\|\|L_K^{1/2}h_\rho\|_K \leq \kappa^{2r-1}\|h_\rho\|_\rho$$

yields that with confidence at least $1-\delta$, there holds

$$\begin{aligned}
&\mathcal{Q}^2_{D_j^*,\lambda}(\mathcal{P}_{D_j,\lambda} + \mathcal{S}_{D_j^*,\lambda}\|f_\lambda\|_K)\\
\leq\ & 16(\kappa+1)\log^3\frac{6}{\delta}\left[\left(\frac{(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]\left[M\mathcal{A}_{D_j,\lambda} + \kappa^{2r}\|h_\rho\|_\rho\mathcal{A}_{D_j^*,\lambda}\right].
\end{aligned}$$

Using the probability to expectation formula

$$E[\xi] = \int_0^\infty \text{Prob}\left[\xi > t\right] dt \tag{29}$$

for nonnegative random variables to $\xi_1 = \mathcal{Q}^4_{D_j^*,\lambda}(\mathcal{P}_{D_j,\lambda} + \mathcal{S}_{D_j^*,\lambda}\|f_\lambda\|_K)^2$ and

$$\text{Prob}\left[\xi_1 > u\right] = \text{Prob}\left[\xi_1^{\frac{1}{2}} > u^{\frac{1}{2}}\right] \leq 6\exp\left\{-[16\mathcal{B}(D_j^*,\lambda)]^{-\frac{1}{3}}u^{\frac{1}{6}}\right\}$$

for $u \geq 256\log^6 6\mathcal{B}(D_j^*,\lambda)^2$, we have

$$\begin{aligned}
&E\left[\mathcal{Q}^4_{D_j^*,\lambda}(\mathcal{P}_{D_j,\lambda} + \mathcal{S}_{D_j^*,\lambda}\|f_\lambda\|_K)^2\right] \leq 256\log^6 6\mathcal{B}(D_j^*,\lambda)^2\\
+\ & 6\int_0^\infty \exp\left\{-[16\mathcal{B}(D_j^*,\lambda)]^{-\frac{1}{3}}u^{\frac{1}{6}}\right\}du = 256(6+\log^6 6)\mathcal{B}^2(D_j^*,\lambda)\int_0^\infty u^{6-1}\exp\{-u\}\,du,
\end{aligned}$$

where

$$\mathcal{B}(D_j^*,\lambda) := (\kappa+1)\left[\left(\frac{(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]\left(M\mathcal{A}_{D_j,\lambda} + \kappa^{2r}\|h_\rho\|_\rho\mathcal{A}_{D_j^*,\lambda}\right). \tag{30}$$

Due to the expression $\int_0^\infty u^{d-1}\exp\{-u\}\,du = \Gamma(d)$ for Gamma functions with $d > 0$, we have

$$E\left[\mathcal{Q}^4_{D_j^*,\lambda}(\mathcal{P}_{D_j,\lambda} + \mathcal{S}_{D_j^*,\lambda}\|f_\lambda\|_K)^2\right] \leq 256(6+\log^6 6)5!\mathcal{B}^2(D_j^*,\lambda). \tag{31}$$

According to Lemma 9, there exist two subsets $\mathcal{Z}_{1',\delta}^{|D_j^*|}$ and $\mathcal{Z}_{2',\delta}^{|D_j^*|}$ of $\mathcal{Z}^{|D_j^*|}$ with measures at least $1-\delta/2$ such that for $D_j^* \in \mathcal{Z}_{1',\delta}^{|D_j^*|} \cap \mathcal{Z}_{2',\delta}^{|D_j^*|}$ there holds

$$\begin{aligned}
&\mathcal{Q}^2_{D_j^*,\lambda}\mathcal{S}_{D_j^*,\lambda}\|f_\lambda - f_\rho\|_K\\
\leq\ & 4(\kappa^2+\kappa)\log\frac{4}{\delta}\left[\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}\log\frac{4}{\delta}}{\sqrt{\lambda}}\right)^2+1\right]\mathcal{A}_{D_j^*,\lambda}\|f_\lambda - f_\rho\|_K.
\end{aligned} \tag{32}$$

Since $r \geq 1/2$, plugging (23) into (32), with confidence at least $1-\delta$, there holds

$$\begin{aligned}
&\mathcal{Q}^2_{D_j^*,\lambda}\mathcal{S}_{D_j^*,\lambda}\|f_\lambda - f_\rho\|_K\\
\leq\ & 4\kappa(\kappa+1)\lambda^{r-1/2}\|h_\rho\|_\rho\log\frac{4}{\delta}\left[\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}\log\frac{4}{\delta}}{\sqrt{\lambda}}\right)^2+1\right]\mathcal{A}_{D_j^*,\lambda}.
\end{aligned}$$

The same method as that in deriving (31) yields

$$\begin{aligned}
&E\left[\mathcal{Q}^4_{D_j^*,\lambda}\mathcal{S}^2_{D_j^*,\lambda}\|f_\lambda - f_\rho\|_K^2\right]\\
\leq\ & 256\kappa^2(\kappa+1)^2(4+\log^6 4)5!\|h_\rho\|_\rho^2\lambda^{2r-1}\left[\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]^2\mathcal{A}^2_{D_j^*,\lambda}.
\end{aligned} \tag{33}$$

Inserting (28), (31), (30) and (33) into (19), we obtain

$$\begin{aligned}
&\max\left\{E[\|\overline{f}_{D^*,\lambda} - f_\rho\|_\rho^2], \lambda E[\|\overline{f}_{D^*,\lambda} - f_\rho\|_K^2]\right\} \leq 2\lambda^{2r}\|h_\rho\|_\rho^2\\
&+512\kappa^2(\kappa+1)^2(4+\log^6 4)5!\|h_\rho\|_\rho^2\lambda^{2r-1}\sum_{j=1}^m\frac{|D_j^*|}{|D^*|}\left[\left(\frac{(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]^2\mathcal{A}^2_{D_j^*,\lambda}\\
&+512(6+\log^6 6)5!(\kappa+1)^2\sum_{j=1}^m\frac{|D_j^*|^2}{|D^*|^2}\left[\left(\frac{(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]^2\\
&\times\left[M\mathcal{A}_{D_j,\lambda} + \kappa^{2r}\|h_\rho\|_\rho\mathcal{A}_{D_j^*,\lambda}\right]^2.
\end{aligned}$$

Then (6) follows from $\mathcal{A}_{D_j^*,\lambda} \leq \mathcal{A}_{D_j,\lambda}$ and the above estimate with

$$C := 2\|h_\rho\|_\rho^2 + 512(6+\log^6 6)5!(\kappa+1)^2[(\kappa+1)^2+1]^2\max\left\{\kappa^2\|h_\rho\|_\rho^2, (M+\|h_\rho\|_\rho\kappa^r)^2\right\}.$$

This completes the proof of Theorem 1. ∎

Proof of Corollary 2 Let $\lambda = |D|^{-\frac{1}{(2r+s)}}$. Since $r+s \geq r \geq 1/2$, we obtain from (7), (8) and $|D_1| = \cdots = |D_m|$ that

$$\mathcal{A}_{D_j,\lambda} \leq m|D|^{-\frac{2r+s-1/2}{2r+s}} + \sqrt{C_0 m}|D|^{-\frac{r}{2r+s}} \qquad \forall j = 1, \ldots, m. \tag{34}$$

Since $|D_1^*| = \cdots = |D_m^*|$, we have

$$\mathcal{A}_{D_j^*,\lambda} \leq m|D^*|^{-1}|D|^{\frac{1}{4r+2s}} + \sqrt{C_0 m}|D^*|^{-1/2}|D|^{\frac{s}{4r+2s}}, \qquad \forall\ j = 1, \ldots, m. \tag{35}$$

This implies

$$\lambda^{-1/2}\mathcal{A}_{D_j^*,\lambda} \leq m|D^*|^{-1}|D|^{\frac{2}{4r+2s}} + \sqrt{C_0 m}|D^*|^{-1/2}|D|^{\frac{s+1}{4r+2s}}.$$

Due to (9), we have

$$\lambda^{-1/2}\mathcal{A}_{D_j^*,\lambda} \leq \sqrt{C_0} + 1. \tag{36}$$

Plugging (34), (35), (36) and (9) into (6) and noticing $\frac{|D_j^*|}{|D^*|} = \frac{1}{m}$ and $m \leq |D|^{\frac{2r+2s-1}{2r+s}}$, we obtain

$$E[\|\overline{f}_{D^*,\lambda} - f_\rho\|_\rho^2] \leq C|D|^{-2r/(2r+s)} + 8C(\sqrt{C_0}+2)^6|D|^{-2r/(2r+s)}.$$

This completes the proof of Corollary 2. ■

Proof of Theorem 3 The proof is almost the same as that of Theorem 1. The only difference is that when $0 < r < 1/2$, we have

$$\|f_\lambda\|_K = \|(L_K+\lambda I)^{-1}L_K f_\rho\|_K \leq \|(L_K+\lambda I)^{-1}L_K^{1/2+r}\|\|L_K^{1/2}h_\rho\|_K \leq \lambda^{r-1/2}\|h_\rho\|_\rho \tag{37}$$

and

$$\|f_\lambda - f_\rho\|_\infty \leq \|f_\lambda\|_\infty + \|f_\rho\|_\infty \leq \kappa\|f_\lambda\|_K + M \leq M + \kappa\lambda^{r-1/2}\|h_\rho\|_\rho. \tag{38}$$

Since $0 < r < 1/2$, we also see from Lemma 8 that

$$\|f_\lambda - f_\rho\|_\rho \leq \lambda^r\|h_\rho\|_\rho. \tag{39}$$

It follows from (37), (27) and the same method as that for deriving (31) that

$$\begin{aligned} E\left[\mathcal{Q}_{D_j^*,\lambda}^4(\mathcal{P}_{D_j,\lambda} + \mathcal{S}_{D_j^*,\lambda}\|f_\lambda\|_K)^2\right] &\leq 512(6+\log^6 6)5!(\kappa+1)^2\left[\left(\frac{(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]^2 \\ &\times \left[M\mathcal{A}_{D_j,\lambda} + \kappa\lambda^{r-1/2}\|h_\rho\|_\rho\mathcal{A}_{D_j^*,\lambda}\right]^2. \end{aligned} \tag{40}$$

Combining (38) and (27) with the same method as that for deriving (33) yields

$$\begin{aligned} &E\left[\mathcal{Q}_{D_j^*,\lambda}^4\mathcal{R}_{D_j^*,\lambda,f_\lambda-f_\rho}^2\right] \\ \leq\ & 512(\kappa+1)^2(4+\log^6 4)5!(M+\kappa\lambda^{r-1/2}\|h_\rho\|_\rho)^2\left[\left(\frac{2(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]^2\mathcal{A}_{D_j^*,\lambda}^2. \end{aligned} \tag{41}$$

Inserting (39), (40) and (41) into (20), we have

$$\begin{aligned} &E\left[\|\overline{f}_{D^*,\lambda} - f_\rho\|_\rho^2\right] \leq 2\lambda^{2r}\|h_\rho\|_\rho^2 + 1024(\kappa+1)^2(4+\log^6 4)5!(M+\kappa\lambda^{r-1/2}\|h_\rho\|_\rho)^2 \\ &\times\sum_{j=1}^m\frac{|D_j^*|}{|D^*|}\left[\left(\frac{(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]^2\mathcal{A}_{D_j^*,\lambda}^2 \\ &+1024(6+\log^6 6)5!(\kappa+1)^2\sum_{j=1}^m\frac{|D_j^*|^2}{|D^*|^2}\left[\left(\frac{(\kappa^2+\kappa)\mathcal{A}_{D_j^*,\lambda}}{\sqrt{\lambda}}\right)^2+1\right]^2 \\ &\times\left[M\mathcal{A}_{D_j,\lambda} + \kappa\lambda^{r-1/2}\|h_\rho\|_\rho\mathcal{A}_{D_j^*,\lambda}\right]^2. \end{aligned}$$

Together with the restrictions $0 < \lambda \leq 1$ and $2r < 1$, this yields (10) with

$$\overline{C} := 2\|h_\rho\|_\rho^2 + 1024(6+\log^6 6)5!(\|h_\rho\|_\rho + M)^2(\kappa+1)^4[(\kappa^2+\kappa)^2+1]^2.$$

This completes the proof of Theorem 3. ■

Proof of Corollary 4 Due to $r+s \geq 1/2$ and (9), we get (34), (35) and (36). Inserting (34), (35), (36) and (9) into (10), we obtain from $|D^*| \geq |D|^{\frac{1+s}{2r+s}}$ that

$$\begin{aligned} &E[\|\overline{f}_{D^*,\lambda} - f_\rho\|_\rho^2] \leq \overline{C}|D|^{-2r/(2r+s)} \\ +\ & \overline{C}((\sqrt{C_0}+1)^2+1)^2\sum_{j=1}^m\frac{|D_j^*|}{|D|}\left[\frac{1}{m}\mathcal{A}_{D_j,\lambda}^2 + \lambda^{2r-1}\mathcal{A}_{D_j^*,\lambda}^2\right] \\ \leq\ & [\overline{C} + 2\overline{C}((\sqrt{C_0}+1)^2+1)^2((1+\sqrt{C_0})^2))]|D|^{-\frac{2r}{2r+s}}, \end{aligned}$$

where we have used $|D^*| \geq |D|^{\frac{2r+s+1}{2r+s}}$ in the last inequality. This completes the proof of Corollary 4. ■

6. Experimental Verifications

In this section, we report experimental studies to justify the statements in Section 2. We employ two criteria for the comparison. The first criterion is the *global error* (GE) which is the mean square error of a testing set with $N = |D|$ examples used in the training flow. GE provides a baseline to assess the performance of DSKRR. The second criterion is the *average error* (AE) which is the mean square error of algorithm (4). Regularization parameters in all experiments are selected by the 5-fold cross-validation.

6.1 Toy simulations

In this part, we carry out three simulations to verify our theoretical statements. The first two simulations are devoted to verifying Corollaries 2 and 4, respectively. The last simulation focuses on the relation between the generalization performance of algorithm (4) and the size of unlabeled data to show the power of unlabeled data in distributed learning.

Simulation 1: We generate $N = 5000$ examples for training. The inputs $\{x_i\}_{i=1}^N$ are independently drawn according to the uniform distribution on the (hyper-)cube $[0,1]^d$

with $d = 1$ or $d = 3$. The corresponding outputs are generated from the regression models $y_i = g_j(x_i)+\varepsilon_i, i = 1, \ldots, N, j = 1, 2$, where ε_i is the independent Gaussian noise $\mathcal{N}(0, 1/5)$,

$$g_1(x) := \begin{cases} x, & 0 < x \le 0.5, \\ 1 - x, & 0.5 < x \le 1, \end{cases} \tag{42}$$

and

$$g_2(x) := h_2(\|x\|_2) := \begin{cases} (1 - \|x\|_2)^6(35\|x\|_2^2 + 18\|x\|_2 + 3), & 0 < \|x\|_2 \le 1, x \in \mathbb{R}^3, \\ 0, & \|x\|_2 > 1. \end{cases} \tag{43}$$

We also generate 500 test examples $\{(x_i', y_i')\}_{i=1}^{500}$ with $\{x_i'\}$ drawn independently according to the uniform distribution and $y_i' = g_j(x_i'), j = 1, 2$. The number m of local processors varies from 2 to 60. The $SN \in \{0, 2000, 4000, 6000, 8000, 10000\}$ unlabeled examples $\{\tilde{x}_i\}_{i=1}^{SN}$ are independently drawn according to the uniform distribution on the (hyper-)cube $[0, 1]^d$. It can be found in (Wu, 1995; Schaback and Wendland, 2006) that $g_1 \in W_1^1$ and $g_2 \in W_3^4$, where W_d^α denotes the α-order Sobolev space on $[0, 1]^d$. Furthermore, it is easy to see that $g_1 \notin W_1^2$ and $g_2 \notin W_3^5$. If we define $K_1(x, x') = 1 + \min(x, x')$ and $K_2(x, x') = h_3(\|x - x'\|_2)$ with

$$g_3(x) := h_3(\|x\|_2) := \begin{cases} (1 - \|x\|_2)^4(4\|x\|_2 + 1), & 0 < \|x\|_2 \le 1, x \in \mathbb{R}^3, \\ 0, & \|x\|_2 > 1, \end{cases} \tag{44}$$

then we know (Wu, 1995; Schaback and Wendland, 2006) that K_1 and K_2 are reproducing kernels for W_1^1 and W_3^2, respectively. Obviously, $g_1 \in \mathcal{H}_{K_1}$ and $g_2 \in \mathcal{H}_{K_2}$. The $N + SN$ sample points are evenly distributively stored in m local processors and algorithm (4) is applied to the training set. The testing results of GEs and AEs are recorded and shown in Figure 2.

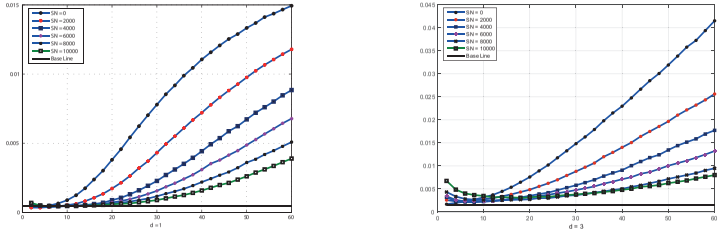


Figure 2: Performances of algorithm (4) with different scale of $|D^*|$ when $f_\rho \in \mathcal{H}_K$

In Figure 2, AE curves are recorded by different SN. When m is not too large, as shown in Figure 2, AEs are always comparable to GEs. Furthermore, there exists an upper bound of the number of local processors, m_{SN} (e.g., $m_{100,00} \approx 30$), lager than which AE curves increase dramatically. This result confirms Corollary 2 which indicates that DSKRR can achieve the optimal learning rate as long as m is not too large. Moreover, when we add more

unlabeled data into the training set, i.e., increasing SN, the upper bound m_{SN} increases and AEs decrease (e.g., the AE curve for $SN = 100, 00$ is almost below all the other curves). This shows the power of adding unlabeled data into training data, as condition (9) shows.

Simulation 2: We generate a training set from the regression model $y_i = g_3(x_i)+\varepsilon_i, i = 1, \ldots, N$ with $N = 5000$, where the inputs $\{x_i\}_{i=1}^N$ are independently drawn according to the uniform distribution on $[0, 1]^3$. We also generate 500 test examples $\{(x_i', y_i')\}_{i=1}^{500}$ with x_i' drawn independently according to the uniform distribution and $y_i' = g_3(x_i')$. The number of servers m varies from 2 to 90 and the number of unlabeled examples is $SN \in \{0, 2000, 4000, 6000, 8000, 10000\}$. We utilize $K_3(x, x') = h_2(\|x - x'\|_2)$ as the kernel. It can be found in (Wu, 1995; Schaback and Wendland, 2006) that K_3 is a reproducing kernel for W_3^4. Obviously, $g_3 \notin \mathcal{H}_{K_3}$. GEs and AEs are recorded and shown in Figure 3.

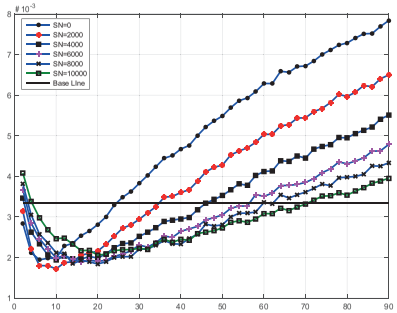


Figure 3: Performances of algorithm (4) with different scale of $|D^*|$ when $f_\rho \notin \mathcal{H}_K$

This simulation shows similar results as Simulation 1. The only difference is that the regression function $g_3 \notin \mathcal{H}_{K_3}$. This demonstrates the statement of Corollary 4, saying that even when the regression function is not in the RHKS, the optimal learning rate of algorithm (4) can be achieved. Based on Simulations 1 and 2, we find that adding unlabeled data into the training set can essentially improve the learning performance of algorithm (1).

Simulation 3: The experimental setting of this simulation is the same as that in Simulation 1. We generate $N = 500$ observations as a training set and 50 observations as a testing set. Here, we fix the size of local processors to be $m = 10$ and vary the number of unlabeled data SN from 50 to 10000. The aim is to describe the relation between the generalization ability of algorithm (4) and the size of unlabeled data. The simulation results are reported in Figure 4.

In Figure 4, we construct two straight lines (black and blue lines) as base lines for comparisons. The black straight line is the GE, while the blue line is the AE value with $SN = 0$. We change the size of unlabeled data and apply algorithm (4) on the training set. It is shown in Figure 4 that the AE's curve decreases dramatically and can achieve the GE value when we add about $10N$ unlabeled data. However, when we add more unlabeled data

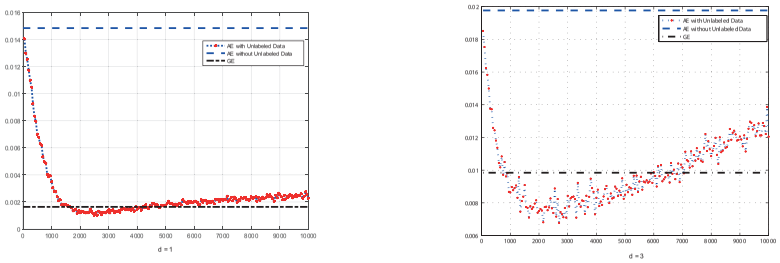


Figure 4: Relation between generalization error and the size of unlabeled data

into the training data, the AE curve begins to increase slowly. This phenomenon indicates two important observations. On one hand, adding unlabeled data into the training set can essentially improve the learning performance of DSKRR. On the other hand, if the size of unlabeled data is too large, the optimal learning rate has reasonable probabilities to be broken out. From our simulations, we suggest that $SN \leq 10N$. Here, we need to mention that for stabilizing the AE's curve, the simulation is repeated 10 times. The drawn curve is the average values of AE.

All these simulations verify our theoretical statements in Section 2 and show the power of unlabeled data in distributed semi-supervised learning.

6.2 Real data experiment

In this part, we focus on the Million Song data (Bertin-Mahieux et al., 2011) that describes a learning task of predicting the year in which a song is released based on audio features associated with the song. The dataset consists of $463,715$ training examples and $51,630$ test examples. Each example is a song released between 1922 and 2011, and the song is represented as a vector of timbre information computed about the song. Each sample point consists of a pair $(x_i, y_i) \in [0,1]^d \times [1922, 2011]$ with $d = 90$.

For the Million Song data, we normalize the feature vectors x so that the timbre signals have mean 0 and standard deviation 1. We also give a feature weight vector $W = (w_1, \ldots, w_d)^\top$ for setting $x_{ij} := w_j x_{ij}$. Here we choose $w_j = 1$ if $j \leq 12$ and $w_j = 0.2$ if $12 < j \leq 90$. Finally, we use the Gaussian kernel $K(x, x') = \exp\left\{-\frac{\|x-x'\|_2^2}{2\beta^2}\right\}$ in our experiments with bandwidth parameter $\beta = 6$ and regularization parameter $\lambda = N^{-1}/2$.

For each feature $X_j = (x_{1j}, \ldots, x_{Nj})^\top$, we denote $x_j^{min} = \min\{x_{1j}, \ldots, x_{Nj}\}$ and $x_j^{max} = \max\{x_{1j}, \ldots, x_{Nj}\}$. Then we generate the unlabeled data $\tilde{x}_{ij}, j = 1, \ldots, SN$, independently from the uniform distribution $U[x_j^{min}, x_j^{max}]$. It should be noted that the distributions for the labeled data and unlabeled data are different, making the learning task a so-called mismatch problem. Algorithm (4) is applied to the training examples with six partitions $m \in \{300, 500, 700, 900, 1100, 1300\}$. Finally, we plot AE curves of DSKRR with 3 different sizes of unlabeled samples in Figure 5. As exhibited in Figure 5, DSKRR

has better performance when SN increases if $m \geq 300$. This phenomenon confirms the observation that adding unlabeled data into training examples can improve the order of m, as condition (9) shows.

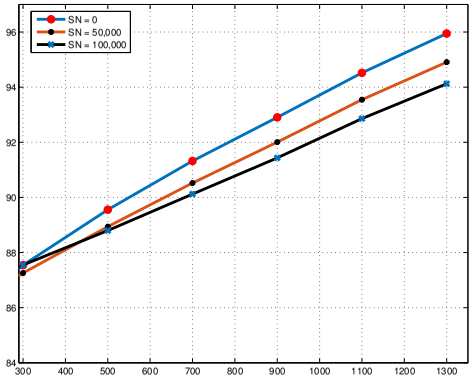


Figure 5: Performance of DSKRR with unlabeled data on Million Song data

Acknowledgments

The work described in this paper is supported partially by the Research Grants Council of Hong Kong [Project No. CityU 11304114] and the National Natural Science Foundation of China (Grant Numbers 61502342, 11401462, 11461161006, 11471292). This paper was written when the last author visited Shanghai Jiaotong University (SJTU). The hospitality and sponsorships from SJTU and Ministry of Education are greatly appreciated. The authors would like to thank the referees for constructive suggestions and Dr. Yucheng Zhang for discussing with the numerical simulations.

References

M. Balcan, A. Blum, S. Fine, and Y. Mansour. Distributed learning, communication complexity, and privacy. *Conference on Learning Theory (COLT)*, 23: 26.1-26.22, 2011.

F. Bauer, S. Pereverzev, and L. Rosasco. On regularization algorithms in learning theory. *Journal of Complexity*, 23: 52-72, 2007.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: a geometric framework for learningform labeled and unlabeled examples. *Journal of Machine Learning Research*,

7: 2339-2434, 2006.

T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere. The million song dataset. In Proceedings of the 12th International conference on Music Information Retrieval, 18, 2011.

G. Blanchard and N. Krämer. Optimal learning rates for kernel conjugate gradient regression. *Advances in Neural Information Processing Systems*, 226-234, 2010.

G. Blanchard and N. Krämer. Convergence rates of kernel conjugate gradient for random design regression. *Analysis and Applications*, 14: 763-794, 2016.

G. Blanchard and N. Mücke. Parallelizing spectral algorithms for kernel learning. arXiv preprint arXiv:1610.07487, 2016.

A. Caponnetto and E. DeVito. Optimal rates for the regularized least squares algorithm. *Foundations of Computational Mathematics*, 7: 331-368, 2007.

A. Caponnetto and Y. Yao. Cross-validation based adaptation for regularization operators in learning theory. *Analysis and Applications*, 8: 161-183, 2010.

D. Gillick, A. Faria, and J. DeNero. Mapreduce: Distributed computing for machine learning. Berkley, December 18, 2006.

Z. C. Guo, S. B. Lin, and D. X. Zhou. Distributed learning with spectral algorithms. *Inverse Problems*, Minor revision under review, 2016.

Z. C. Guo, S. B. Lin and L. Shi. Distributed learning with multi-penalty regularization. Submitted, 2017.

L. Györfy, M. Kohler, A. Krzyzak, H. Walk. A Distribution-Free Theory of Nonparametric Regression. Springer-Verlag, Berlin, 2002.

T. Hu, J. Fan, Q. Wu, and D. X. Zhou. Regularization schemes for minimum error entropy principle. *Analysis and Applications*, 13:437–455, 2015.

C. Li, P. Zhou, and T. Jiang. Differential privacy and distributed online learning for wireless big data. *Wireless Communications & Signal Processing (WCSP), 2015 International Conference on. IEEE*, 1-5, 2015.

S. B. Lin, X. Guo, and D. X. Zhou, Distributed learning with regularized least squres. *Journal of Machine Learning Research*, Revision under review, (arXiv 1608.03339), 2016.

S. B. Lin and D. X. Zhou, Distributed kernel gradient descent algorithms. *Constructive Approximation*, Minor revision under review, 2016.

G. Mann, R. McDonald, M. Mohri, N. Silberman, and D. Walker. Efficient large-scale distributed training of conditional maximum entropy models. *Advances in Neural Information Processing Systems*, 1231-1239, 2009.

R. Schaback and H. Wendland. Kernel techniques: from machine learning to meshless methods. *Acta Numerica*, 15: 543-639, 2006.

O. Shamir and N. Srebro. Distributed stochastic optimization and learning. *In 52nd Annual Allerton Conference on Communication, Control and Computing*, 2014.

L. Shi, Y. L. Feng and D. X. Zhou. Concentration estimates for learning with l^1-regularizer and data dependent hypothesis spaces. *Applied and Computational Harmonic Analysis*, 31: 286-302, 2011.

S. Smale and D. X. Zhou. Shannon sampling and function reconstruction from point values. *Bulletin of the American Mathematical Society*, 41: 279-305, 2004.

S. Smale and D. X. Zhou. Shannon sampling II: Connections to learning theory. *Appllied and Computational Harmonic Analysis*, 19: 285-302, 2005.

S. Smale and D.X. Zhou. Learning theory estimates via integral operators and their approximations. *Constructive Approximation*, 26: 153-172, 2007.

I. Steinwart, D. Hush, and C. Scovel. Optimal rates for regularized least squares regression. *Conference on Learning Theory (COLT Dasgupta and A. Klivans, eds.)*, 22: 79-93, 2009.

Z. M. Wu. Compactly supported positive definite radial functions. *Advances in Computational Mathematics*, 4: 283-292, 1995.

L. Xie, S. Plis, and A. D. Sarwate. Data-weighted ensemble learning for privacy-preserving distributed learning. 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2016.

F. Yan, S. Sundaram, S. Vishwanathan, and Y. Qi. Distributed autonomous online learning: Regrets and intrinsic privacy-preserving properties. *IEEE Transaction on Knowledge and Data Engineering*, 25: 2483-2493, 2013.

Y. C. Zhang, J. Duchi, and M. Wainwright. Communication-efficient algorithms for statistical optimization. *Journal of Machine Learning Research*, 14: 3321-3363, 2013.

Y. C. Zhang, J. Duchi, and M. Wainwright. Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates. *Journal of Machine Learning Research*, 16: 3299-3340, 2015.

M. Zinkevich, M. Weimer, A. Smola, and L. Li. Parallelized stochastic gradient descent. *Advances in neural information processing systems*, 2595-2603, 2010.

X. Zhu and A. Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning,* 3: 1-130, 2009.

Journal of Machine Learning Research 18 (2017) 1-43 Submitted 5/15; Revised 11/16; Published 5/17

On Markov chain Monte Carlo methods for tall data

Rémi Bardenet REMI.BARDENET@GMAIL.COM
Université de Lille, CNRS, Centrale Lille, Inria,
UMR 9189 - CRIStAL,
59651 Villeneuve d'Ascq, France

Arnaud Doucet DOUCET@STATS.OX.AC.UK
Chris Holmes CHOLMES@STATS.OX.AC.UK
Department of Statistics, University of Oxford,
Oxford OX1 3TG, United Kingdom

Editor: Robert McCulloch

Abstract

Markov chain Monte Carlo methods are often deemed too computationally intensive to be of any practical use for big data applications, and in particular for inference on datasets containing a large number n of individual data points, also known as tall datasets. In scenarios where data are assumed independent, various approaches to scale up the Metropolis-Hastings algorithm in a Bayesian inference context have been recently proposed in machine learning and computational statistics. These approaches can be grouped into two categories: divide-and-conquer approaches and, subsampling-based algorithms. The aims of this article are as follows. First, we present a comprehensive review of the existing literature, commenting on the underlying assumptions and theoretical guarantees of each method. Second, by leveraging our understanding of these limitations, we propose an original subsampling-based approach relying on a control variate method which samples under regularity conditions from a distribution provably close to the posterior distribution of interest, yet can require less than $\mathcal{O}(n)$ data point likelihood evaluations at each iteration for certain statistical models in favourable scenarios. Finally, we emphasize that we have only been able so far to propose subsampling-based methods which display good performance in scenarios where the Bernstein-von Mises approximation of the target posterior distribution is excellent. It remains an open challenge to develop such methods in scenarios where the Bernstein-von Mises approximation is poor.

Contents

1 Introduction **2**

2 Bayesian inference, MCMC, and tall data **3**
2.1 Bayesian inference . . . 3
2.2 The Metropolis-Hastings algorithm . . . 4
2.3 Running examples . . . 5

3 Divide-and-conquer approaches **5**
3.1 Representing the posterior as a combination of batch posteriors . . . 5
3.2 Replacing the posterior by a geometric combination of batch posteriors . . . 7

4 Exact subsampling approaches: Pseudo-marginal MH **8**
4.1 Pseudo-marginal Metropolis-Hastings . . . 8
4.2 Unbiased estimation of the likelihood using unbiased estimates of the log-likelihood . . . 9
4.3 Building $\hat{\gamma}(\theta)$ with auxiliary variables . . . 10

5 Other exact approaches **13**
5.1 Forgetting about acceptance probability: stochastic approximation approaches 14
5.2 Delayed acceptance . . . 16
5.3 Piecewise deterministic Markov chain Monte Carlo methods . . . 16

6 Approximate subsampling approaches **17**
6.1 Naive subsampling . . . 17
6.2 Relying on the CLT . . . 19
6.2.1 A pseudo-marginal approach with variance reduction under Gaussian assumption . . . 19
6.2.2 Adaptive subsampling with T-tests . . . 20
6.3 Exchanging acceptance noise for subsampling noise . . . 21
6.4 Confidence samplers . . . 24

7 An improved confidence sampler **25**
7.1 Introducing proxies in the confidence sampler . . . 25
7.2 An example proxy: Taylor expansions . . . 30
7.2.1 Taylor expansions . . . 30
7.2.2 Drop proxies along the way . . . 31
7.2.3 A heuristic on the subsampling gain . . . 31

8 Experiments **33**
8.1 Logistic regression . . . 33
8.1.1 A Taylor proxy for logistic regression . . . 33
8.1.2 A toy example that requires $\mathcal{O}(1)$ likelihood evaluations . . . 33
8.1.3 The *covtype* dataset . . . 34
8.2 Gamma linear regression . . . 35
8.2.1 A Taylor proxy for gamma regression . . . 35
8.2.2 The *covtype* dataset . . . 36

9 Discussion **37**

©2017 Rémi Bardenet, Arnaud Doucet, and Chris Holmes.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-205.html.

1. Introduction

Performing inference on tall datasets, that is datasets containing a large number n of individual data points, is a major aspect of the big data challenge. Statistical models, and Bayesian methods in particular, commonly demand Markov chain Monte Carlo (MCMC) algorithms to make inference, yet running MCMC on such tall datasets is often far too computationally intensive to be of any practical use. Indeed, MCMC algorithms such as the Metropolis-Hastings (MH) algorithm require at each iteration to sweep over the

whole dataset. Frequentist or variational Bayes approaches are thus usually preferred to a fully Bayesian analysis in the tall data context on computational grounds. However, they might be difficult to put in practice or justify in scenarios where the likelihood function is complex; e.g. non-differentiable (Chernozhukov and Hong, 2003). Moreover, some applications require precise quantification of uncertainties and a full Bayesian approach might be preferable in those instances. This is the case for example for applications from experimental sciences, such as cosmology (Trotta, 2006) or genomics (Wright, 2014), were such big data problems abound. Consequently, much efforts have been devoted over recent years to develop scalable MCMC algorithms. These approaches can be broadly classified into two groups: divide-and-conquer approaches and subsampling-based algorithms. Divide-and-conquer approaches divide the initial dataset into batches, run MCMC on each batch separately, and then combine these results to obtain an approximation of the posterior: Subsampling approaches aim at reducing the number of individual data point likelihood evaluations necessary at each iteration of the MH algorithm.

After briefly reviewing the limitations of MCMC for tall data, introducing our notation and two running examples in Section 2, we first review the divide-and-conquer literature in Section 3. The rest of the paper is devoted to subsampling approaches. In Section 4, we discuss pseudo-marginal MH algorithms. These approaches are exact in the sense that they target the correct posterior distribution. In Section 5, we review other exact approaches, before relaxing exactness in Section 6. Throughout, we focus on the assumptions and guarantees of each method. We also illustrate key methods on two running examples. Finally, in Section 7, we improve over our so-called confidence sampler in (Bardenet et al., 2014), which samples from a controlled approximation of the target. We demonstrate these improvements yield significant reductions in computational complexity at each iteration in Section 8. In particular, our improved confidence sampler can break the $\mathcal{O}(n)$ barrier of number of individual data point likelihood evaluations per iteration in favourable cases. Its main limitation is the requirement for cheap-to-evaluate proxies for the log-likelihood, with a known error. We provide examples of such proxies relying on Taylor expansions.

All examples can be rerun or modified using the companion IPython notebook[1] to the paper.

2. Bayesian inference, MCMC, and tall data

In this section, we describe the inference problem of interest and the associated MH algorithm. We also detail the two running examples on which we benchmark key methods in Section 4, 5 and 6.

2.1 Bayesian inference

Consider a dataset

$$\mathcal{X} = \{x_1, ..., x_n\} \subset \mathbb{X} \subset \mathbb{R}^d, \tag{1}$$

and a parameter space Θ. We assume the data are conditionally independent with associated likelihood $\prod_{i=1}^n p(x_i|\theta)$ given a parameter value θ and we denote $\ell(\theta)$ the associated average log-likelihood

$$\ell(\theta) = \frac{1}{n}\sum_{i=1}^n \log p(x_i|\theta) = \frac{1}{n}\sum_{i=1}^n \ell_i(\theta). \tag{2}$$

We follow a Bayesian approach where one assigns a prior $p(\theta)$ to the unknown parameter, so that inference relies on the posterior distribution

$$\pi(\theta) = p(\theta|x) \propto \gamma(\theta) \triangleq p(\theta)e^{n\ell(\theta)}, \tag{3}$$

where γ denotes an unnormalized version of π. In most applications, π is intractable and we will focus here on Markov chain Monte Carlo methods (MCMC; Robert and Casella, 2004) and, in particular, on the Metropolis-Hastings (MH) algorithm to approximate it.

1. https://github.com/rbardenet/2017-JMLR-MCMCForTallData

2.2 The Metropolis-Hastings algorithm

A standard approach to sample approximately from $\pi(\theta)$ is to use MCMC algorithms. To illustrate the limitation of MCMC in the tall data context, we focus here on the MH algorithm (Robert and Casella, 2004, Chapter 7.3). The MH algorithm simulates a Markov chain $(\theta_k)_{k\geq 0}$ of invariant distribution π. Then, under weak assumptions, see e.g. (Douc et al., 2014, Theorem 7.32), the following central limit theorem holds for suitable test functions h

$$\sqrt{N_{\text{iter}}}\left[\frac{1}{N_{\text{iter}}}\sum_{k=0}^{N_{\text{iter}}} h(\theta_k) - \int h(\theta)\,\pi(\theta)d\theta\right] \to \mathcal{N}(0, \sigma^2_{\text{lim}}(h)), \tag{4}$$

where convergence is in distribution.

MH$\big(\gamma(\cdot),\, q(\cdot|\cdot),\, \theta_0,\, N_{\text{iter}}\big)$

1 **for** $k \leftarrow 1$ **to** N_{iter}
2 $\quad\theta \leftarrow \theta_{k-1}$,
3 $\quad\theta' \sim q(.|\theta)$,
4 $\quad u \sim \mathcal{U}_{(0,1)}$
5 $\quad\alpha(\theta,\theta') \leftarrow \frac{\gamma(\theta')}{\gamma(\theta)} \times \frac{q(\theta|\theta')}{q(\theta'|\theta)}$
6 $\quad$**if** $u < \alpha(\theta,\theta')$
7 $\quad\quad\theta_k \leftarrow \theta'$ ▷ *Accept*
8 $\quad$**else** $\theta_k \leftarrow \theta$ ▷ *Reject*
9 **return** $(\theta_k)_{k=1,...,N_{\text{iter}}}$

Figure 1: Pseudocode of the MH algorithm targeting the distribution π. Note that π is only involved in ratios, so that one only needs to know an unnormalized version γ of π.

The pseudocode of MH targeting a generic distribution π is given in Figure 1. In the case of Bayesian inference with independent data (3), Step 5 is equivalent to setting

$$\log\alpha(\theta,\theta') \;=\; \log\left[\frac{p(\theta')}{p(\theta)}\frac{q(\theta|\theta')}{q(\theta'|\theta)}\right] + n\left[\ell(\theta') - \ell(\theta)\right]. \tag{5}$$

When the dataset is tall ($n \gg 1$), evaluating the log likelihood ratio in (5) is too costly an operation and rules out the applicability of such a method. As we shall see, two possible options are to either divide the dataset into tractable batches, or approximate the acceptance ratio in (5) using only part of the dataset.

2.3 Running examples

We will evaluate some of the described approaches on two illustrative simple running examples. We fit a one-dimensional normal distribution $p(\cdot|\mu,\sigma) = \mathcal{N}(\cdot|\mu,\sigma^2)$ to 10^5 i.i.d. points drawn according to $X_i \sim \mathcal{N}(0,1)$ and lognormal observations $X_i \sim \log\mathcal{N}(0,1)$, respectively. The latter example illustrates a misspecification of the model. We assign a flat prior $p(\mu, \log\sigma) \propto 1$. For all algorithms, we start the chain at the maximum a posteriori (MAP) estimate. The MH proposal is an isotropic Gaussian random walk, whose stepsize is first set proportional to $1/\sqrt{n}$ and then adapted during the first 1 000 iterations so as to reach 50% acceptance. When applicable, we also display the number of likelihood evaluations per iteration, and compare it to the n evaluations required at each iteration by the MH algorithm.

In Figure 2, we illustrate the results of 10 000 iterations of vanilla MH on each of the two datasets. MH does well, as the posterior coincides with that of a longer reference run of 50 000 iterations in each case, and the autocorrelations show a fast exponential decrease. The Bernstein-von Mises approximation (van der Vaart, 2000, Chapter 10.2), a Gaussian centered at the true value, with covariance minus the scaled inverse Fisher information, is a very good approximation to the posterior in both cases. We are thus in simple cases of heavy concentration of the posterior, where subsampling should help a lot if it is to be of any help in tackling tall data problems.

3. Divide-and-conquer approaches

A natural way to tackle tall data problems is to divide the data into batches, run MH on each batch separately, and then combine the results.

3.1 Representing the posterior as a combination of batch posteriors

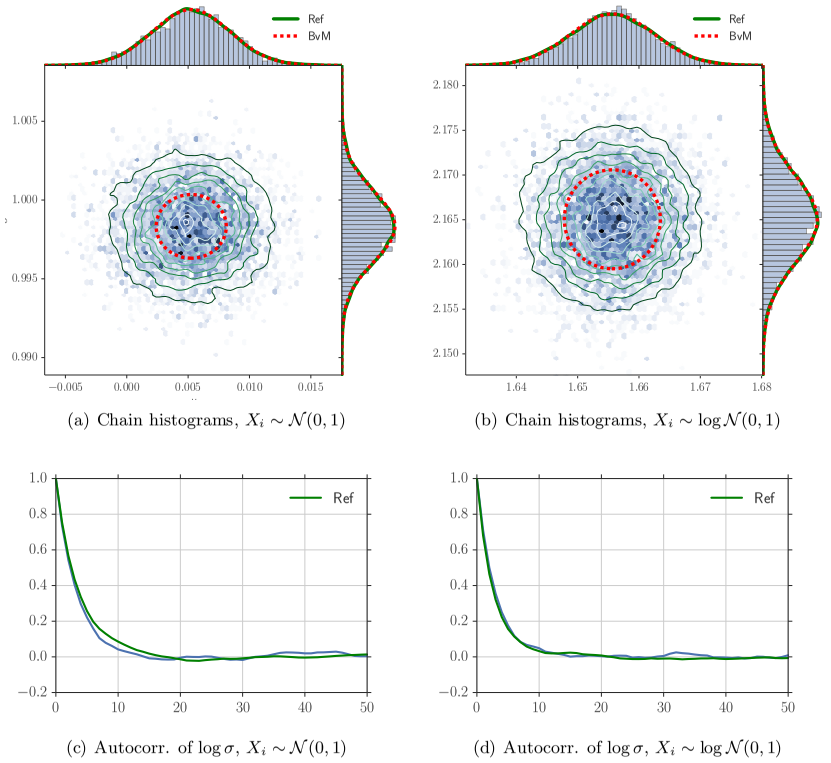
Assume data $\mathcal{X}$ are divided in B batches $\mathbf{x}_1, \ldots, \mathbf{x}_B$. Relying on the equality

$$p(\theta|\mathcal{X}) \propto \prod_{i=1}^{B} p(\theta)^{1/B} p(\mathbf{x}_i|\theta), \tag{6}$$

Huang and Gelman (2005) propose to combine the batch posterior approximations using Gaussian approximations or importance sampling. Scott et al. (2013) propose to average samples across batches, noting this is exact under Gaussian assumptions. Neiswanger et al. (2014) propose to run an MCMC chain on each batch $\mathbf{x}_i$ targeting an artificial batch posterior

$$\pi_i(\theta) \propto p(\theta)^{1/B} p(\mathbf{x}_i|\theta),$$

fit a smooth approximation to each batch posterior, and multiply them. These methods are however theoretically justified only when batch posteriors are Gaussian, or when the size *of*



(a) Chain histograms, $X_i \sim \mathcal{N}(0,1)$

(b) Chain histograms, $X_i \sim \log\mathcal{N}(0,1)$

(c) Autocorr. of $\log\sigma$, $X_i \sim \mathcal{N}(0,1)$

(d) Autocorr. of $\log\sigma$, $X_i \sim \log\mathcal{N}(0,1)$

Figure 2: Results of 10 000 iterations of vanilla MH fitting a Gaussian model to one-dimensional Gaussian and lognormal synthetic data, on the left and right panel, respectively. Figures 2(a) and 2(b) show the chain histograms, joint and marginals; the x-axis corresponds to the mean of the fitted Gaussian, the y-axis to the standard deviation. We have superimposed a kernel density estimator of a long MH chain for reference in green and the Bernstein-von Mises Gaussian approximation in red. Figures 2(c) and 2(d) show the marginal autocorrelation of $\log\sigma$ in blue. The green curves are baselines that correspond to the long MH reference run depicted in green in the top panel; although the green autocorrelation functions are of limited interest when comparing them to vanilla MH, we use them as reference in all similar later figures.

each batch goes to infinity, to guarantee that the used smooth approximation of each batch posterior is accurate.

There are few results available on how the properties of combined estimators scale with the number of batches B. Neiswanger et al. (2014) fit a kernel density estimator to the samples of each batchwise chain, and multiply the resulting kernel density estimators. A sample from this mixture approximation to π is then obtained through an additional MCMC step. Under simplifying assumptions (all MCMC chains are assumed being independent draws from their targets, for example), a bound on the MSE of the final estimator is obtained. However, this bound explodes as the kernel bandwidth goes to zero, and more importantly, it is exponential in the number of batches B. In a tall data context, the number of batches is expected to grow with n to ensure that the size of each batch is less than $\mathcal{O}(n)$. Thus, the proposed bound is currently not informative for tall data.

As pointed out by Wang and Dunson (2013), if the supports of the π_i are almost disjoint, then the product of their approximations will be a poor approximation to π. To improve the overlap between the approximations of the π_i's, Wang and Dunson (2013) propose to replace the posterior in (6) by the product of the Weierstrass transforms of each batch posterior. When the approximation of π_i is an empirical measure, its Weierstrass transform corresponds to a kernel density estimator. The product of the Weierstrass transforms can be interpreted as the marginal distribution of an extended distribution on Θ^{B+1}, where the first B copies of θ are associated to the B batches, and the remaining copy is conditionally Gaussian around a weighted mean of the first B copies. Unfortunately, sampling from the posterior of this artificial model is difficult when one only has access to approximate samples of each π_i.

Although it is not strictly speaking a Monte Carlo method, we note that Xu et al. (2014) and Gelman et al. (2014) propose an expectation-propagation-like algorithm that similarly tackles the issue of disjoint approximate batch posterior supports. Each batch of data points is represented by its individual likelihood times a *cavity* distribution. The cavity distribution is itself the product of the prior and a number of terms that represent the contributions of other batches to the likelihood. The algorithm iterates between 1) simulating from each batchwise likelihood times a batch-specific cavity distribution, and 2) fitting each batch-specific cavity component. Again, while these approaches are computationally feasible and appear to perform well experimentally, it is difficult to assert the characteristics of the proposed approximation of the posterior and there is no convergence guarantee available for this iterative algorithm.

3.2 Replacing the posterior by a geometric combination of batch posteriors

Another avenue of research consists in avoiding multiplying the batch posteriors by replacing the target by a different combination of the latter.

By introducing a suitable metric on the space of probability measures such as the Wasserstein metric, it is for example possible to define the barycenter or the median of a set of probability measures. Minsker et al. (2014) propose to output the median of the batch posteriors, while Srivastava et al. (2015) use the Wasserstein barycenter, which can be computed efficiently in practice using the techniques developed by Cuturi and Doucet (2014). While this idea has some appeal, the statistical meaning of these median or mean measures

is unclear, and the robustness of the median estimate advocated in (Minsker et al., 2014) may also be a drawback, as in some circumstances valuable information contained in some batches may be lost.

To conclude, divide-and-conquer techniques appear as a natural approach to handle tall data. However, the crux is how to efficiently combine the batch posterior approximations. The main issues are that the batch posterior approximations potentially have disjoint supports, that the multiplicative structure of the posterior (3) leads to poor scaling with the number of batches, that theoretical guarantees are often asymptotic in the batch size, and that cheap-to-sample combinations of batch posteriors are difficult to interpret.

4. Exact subsampling approaches: Pseudo-marginal MH

Pseudo-marginal MH (Lin et al., 2000; Beaumont, 2003; Andrieu and Roberts, 2009) is a variant of MH, which relies on unbiased estimators of an unnormalized version of the target. Pseudo-marginal MH is useful to help understand several potential approaches to scale up MCMC. We start by describing pseudo-marginal MH in Section 4.1. Then, we present two pseudo-marginal approaches to tall data in Section 4.2 and Section 4.3.

4.1 Pseudo-marginal Metropolis-Hastings

Assume that instead of being able to evaluate $\gamma(\theta)$, we have access to an unbiased, almost-surely *non-negative* estimator $\hat{\gamma}(\theta)$ of the unnormalized target $\gamma(\theta)$. Pseudo-marginal MH substitutes a realization of $\hat{\gamma}(\theta')$ to $\gamma(\theta')$ in Step 5. Similarly, it replaces $\gamma(\theta)$ in Step 5 by the realization of $\hat{\gamma}(\theta)$ that was computed when the parameter value θ was accepted. Pseudo-marginal MH is of considerable practical importance, with applications such as particle marginal MH (Andrieu et al., 2010) and MCMC versions of the approximate Bayesian computation paradigm (Marin et al., 2012). It is thus worth investigating its use in the context of tall data problems.

The possibility to use an unbiased estimator of γ comes at a price: first, the asymptotic variance σ^2_{lim} in (4) of an MCMC estimator based on a pseudo-marginal chain will always be larger than that of an estimator based on the underlying "marginal" MH (Andrieu and Vihola, 2015). Second, the qualitative properties of the underlying MH may not be preserved, meaning that the rate of convergence to the invariant distribution may go from geometric to subgeometric, for instance; see Andrieu and Roberts (2009) and Andrieu and Vihola (2015) for a detailed discussion. In practice, if the variance of $\hat{\gamma}(\vartheta)$ is large for some value $\vartheta \in \Theta$, then an MH move to ϑ might be accepted while $\hat{\gamma}(\vartheta)$ largely overestimates $\gamma(\vartheta)$. In that case, it is difficult for the chain to leave ϑ, and pseudo-marginal MH chains thus tend to get stuck if the variance of the involved estimators is not controlled. When some tunable parameter allows to control this variance, Doucet et al. (2015) show that, in order to minimize the variance of MCMC estimates for a fixed computational complexity, the variance of the log-likelihood estimator should be kept around 1.0 when the ideal MH having access to the exact likelihood generates quasi-i.i.d samples from π; or set to around 3.0 when it exhibits very large integrated autocorrelation times. In practice, the integrated autocorrelation times of averages under the ideal MH are unknown as this algorithm cannot be implemented. In this common scenario, Doucet et al. (2015) recommend keeping the variance around 1.5 as this is a value which ensures a small penalty in performance even in

scenarios where 1.0 or 3.0 are actually optimal. They also show that the penalty incurred for having a variance too small (i.e. inferior to 0.2) or too large (i.e. superior to 10) is very large. When mentioning pseudo-marginal MH algorithms, we will thus comment on the variance of the logarithm of the involved estimators $\hat{\gamma}(\theta)$, or, if not available, of their relative variance.

4.2 Unbiased estimation of the likelihood using unbiased estimates of the log-likelihood

As described in Section 4.1, pseudo-marginal MH requires an almost-surely nonnegative unbiased estimator $\hat{\gamma}(\theta)$ of the unnormalized posterior at θ, for any θ in Θ. It is easy to check that, by sampling $x_1^*, \ldots, x_t^*$ from the dataset $\mathcal{X}$ with or without replacement, we obtain the following unbiased estimator of the log-likelihood $n\ell(\theta)$

$$n\hat{\ell}(\theta) = \frac{n}{t}\sum_{i=1}^{t} \log p(x_i^*|\theta). \tag{7}$$

We denote by $\hat{\ell}(\theta)$ the subsampling estimate of the average log-likelihood and denote by $\sigma_t(\theta)^2$ its variance. Obviously, exponentiating (7) does not provide an unbiased estimate of the likelihood $e^{n\ell(\theta)}$. However, an interesting question is whether one can design a procedure which outputs an unbiased, almost-surely nonnegative estimate of $e^{n\ell(\theta)}$ using unbiased estimates of $n\ell(\theta)$ such as $n\hat{\ell}(\theta)$. Without making any further assumption about $n\hat{\ell}(\theta)$, it was recently shown by Jacob and Thiery (2015) that it is not possible. However, this can be done if one further assumes, for instance, that there exists $a(\theta)$ such that $\ell_i(\theta) > a(\theta)$ for all i, see (Jacob and Thiery, 2015, Section 3.1) who rely on a technique by Rhee and Glynn (2015) generalizing (Bhanot and Kennedy, 1985). Unfortunately, as we shall see, the resulting estimator $\hat{\gamma}(\theta)$ typically has a very large relative variance, resulting in very poor performance of the associated pseudo-marginal chain.

We apply (Rhee and Glynn, 2015, Theorem 1) to build an unbiased non-negative estimator of $\gamma(\theta)/p(\theta)$, which is equivalent to defining $\hat{\gamma}(\theta)$. For $j \geq 1$, let

$$D_j^* = \frac{n}{t}\sum_{i=1}^{t} \log p(x_{i,j}^*|\theta) - na(\theta), \tag{8}$$

be an unbiased estimator of $n\left(\ell(\theta) - a(\theta)\right)$, where the $x_{i,j}^*$'s are drawn with replacement from $\mathcal{X}$ for each i, and are further independent across j. In (Jacob and Thiery, 2015, Section 3.1), N is an integer-valued random variable whose tails do not decrease too fast, in the sense that $\mathbb{P}(N \geq k) \geq C(1+\epsilon)^{-k}$. To ease computations, we take N to be geometric with parameter $\epsilon/(1+\epsilon)$. This corresponds to the lightest tails allowed by (Jacob and Thiery, 2015, Section 3.1), since $\mathbb{P}(N \geq k) = (1+\epsilon)^{-k}$. Finally, let

$$Y \triangleq e^{na(\theta)}\left[1 + \sum_{k=1}^{N} \frac{1}{\mathbb{P}(N \geq k)}\frac{1}{k!}\prod_{j=1}^{k} D_j^*\right]. \tag{9}$$

By (Rhee and Glynn, 2015, Theorem 1), Y is a non-negative unbiased estimator of the likelihood $e^{n\ell(\theta)}$. As mentioned in Section 4.1, it is crucial, if we want to plug $\hat{\gamma}(\theta) = Y \times p(\theta)$ in a pseudo-marginal algorithm, to control the variance of its logarithm. The variance of $\log Y$ is difficult to compute, so we use here the relative variance of Y as a proxy.

Proposition 1 *Let $\theta \in \Theta$ and Y be the almost surely non-negative estimator of $e^{n\ell(\theta)}$ defined in* (9)*. Then its relative variance satisfies*

$$\frac{\mathrm{Var}Y}{e^{2n\ell(\theta)}} \geq \frac{e^{-2n(\ell(\theta)-a(\theta))+2n\sqrt{(1+\epsilon)[\sigma_t(\theta)^2+(\ell(\theta)-a(\theta))^2]}}}{n\sqrt{(1+\epsilon)[\sigma_t(\theta)^2+(\ell(\theta)-a(\theta))^2]}} + \mathcal{O}(1). \tag{10}$$

The proof of Proposition 1 can be found in Appendix A. It follows from (10) that in order for the relative variance of Y not to increase exponentially with n, it is necessary that $n\sigma_t(\theta)$ is of order 1. But $\sigma_t(\theta)$ is of order $t^{-1/2}$, so that the batchsize t would have to be of order n^2, which is impractical. It is also necessary that $\sqrt{1+\epsilon}$ is of order $1+n^{-1}$ to control the term in $(\ell(\theta)-a(\theta))$. This means that ϵ should be taken of order n^{-1}, but then the mean $(1+\epsilon)\epsilon^{-1}$ of the geometric variable N will be of order n. This entails that the number of terms in the randomly truncated series (9) should be of order n, which defeats the purpose of using this estimator.

Hence for the reasons outlined in Section 4.1, we expect the pseudo-marginal MH relying on Y to be highly inefficient. Indeed, we have not been able to obtain reasonably mixing chains even on our Gaussian running example. We have experimented with various choices of ϵ, and with various values of t, but none yielded satisfactory results. We conclude that this approach is not a viable solution to MH for tall data.

We note that Strathmann et al. (2015) have recently proposed a different way to exploit the methodology of Rhee and Glynn (2015) in the context of tall data. However, their methodology does not provide unbiased estimates of the posterior expectations of interest. It only provides unbiased estimates of some biased MCMC estimates of these expectations, these biased MCMC estimates corresponding to running an MCMC kernel on the whole dataset for a finite number of iterations. Strathmann et al. (2015) suggest that it might be possible to combine their algorithm with the recent scheme of Glynn and Rhee (2014) to obtain unbiased estimates of the posterior expectations. It is unclear whether this could be achieved under realistic assumptions on the MCMC kernel.

4.3 Building $\hat{\gamma}(\theta)$ with auxiliary variables

In (MacLaurin and Adams, 2014), the authors propose an alternative MCMC to sample from π which, similarly to the method described previously, only requires evaluating the likelihood of a subset of the data at each iteration. Assume a bound $\ell_i(\theta) \geq b_i(\theta)$ is available for each i and θ. For simplicity, we further assume that $b_i(\theta) = b(\theta, x_i)$ only depends on i through x_i. This is the case in the experiments of (MacLaurin and Adams, 2014), as well as ours. Note also that in Section 4.2, we used a bound that was uniform in the data index i; we could have used similarly a non-uniform bound, but this would have made the derivation of Proposition 1 unnecessarily heavy.

As noted in (MacLaurin and Adams, 2014), we can then define the following extended target $\tilde{\pi}$ distribution on $\Theta \times \{0,1\}^n$

$$\begin{aligned}\tilde{\pi}(\theta, z) &\propto p(\theta) \prod_{i=1}^{n} \left[\exp\left(\ell_i(\theta)\right) - \exp\left(b_i(\theta)\right)\right]^{z_i} \exp\left(b_i(\theta)\right)^{1-z_i} \\ &= p(\theta) \prod_{i=1}^{n} \exp\left(b_i(\theta)\right) \prod_{i=1}^{n} \left[\exp\left(\ell_i(\theta) - b_i(\theta)\right) - 1\right]^{z_i}. \end{aligned} \tag{11}$$

This distribution satisfies two important features: it admits $\pi(\theta)$ as a marginal distribution, and its pointwise evaluation only requires to evaluate $\ell_i(\theta)$ for those i's for which $z_i = 1$. Note that evaluating $\tilde{\pi}(\theta, z)$ however requires to evaluate $\prod_{i=1}^{n} \exp(b_i(\theta))$, and the bounds $b_i(\theta)$ thus must be chosen so that this computation is cheap. This is the case for the lower bound of the logistic regression log-likelihood model discussed in (MacLaurin and Adams, 2014), which is a quadratic form in $t_i \theta^T x_i$, where t_i is the ± 1 label of datum x_i. The idea of replacing the evaluation of the target by a Bernoulli draw and the evaluation of a lower bound has been exploited previously in the physics literature; see e.g. (Mak, 2005).

Any MCMC sampler could be used to sample from $\tilde{\pi}$. MacLaurin and Adams (2014) propose an MH-within-Gibbs sampler that leverages the known conditional $\tilde{\pi}(z|\theta)$. The expected cost of one conditional MH iteration on θ at equilibrium, that is the average number of indices i such that $z_i = 1$, is $\mathcal{O}(n)$, and the constant is related to the expected relative tightness of the bound, see (MacLaurin and Adams, 2014, Section 3.1). The number of likelihood evaluations for an update of z conditional on θ is explicitly controlled in (MacLaurin and Adams, 2014) by either specifying a maximum number of attempted flips, or implicitly specifying the fraction of flips to 1.

The authors of MacLaurin and Adams (2014) remarked that their methodology is related to pseudo-marginal techniques but did not elaborate. We show here how it is indeed possible to exploit the extended target distribution $\tilde{\pi}$ in (11) to obtain an unbiased estimate of an unnormalized version of π. More precisely, we have

$$p(x_i|\theta) = \sum_{z_i \in \{0,1\}} p\left(x_i, z_i|\theta\right)$$

where $p\left(z_i|\theta, x_i\right) = \{1 - \exp(b_i(\theta) - \ell_i(\theta))\}^{z_i} \exp(b_i(\theta) - \ell_i(\theta))^{1-z_i}$. Hence, the marginal distribution of z_i under this extended model is given by

$$\begin{aligned} p(z_i = 1|\theta) &= \int p(z_i = 1, x_i|\theta) dx_i \\ &= \int \left[\exp(\ell_i(\theta)) - \exp(b_i(\theta))\right] dx_i \\ &= 1 - I_\theta, \end{aligned} \tag{12}$$

where $I_\theta \triangleq \int \exp(b(\theta, x)) dx$. Using Bayes' theorem, we obtain accordingly

$$p(x_i|\theta, z_i = 1) = \frac{\exp(\ell_i(\theta)) - \exp(b_i(\theta))}{1 - I_\theta},\ p(x_i|\theta, z_i = 0) = \frac{\exp(b_i(\theta))}{I_\theta}.$$

An obvious unbiased estimator of the unnormalized posterior is thus given by

$$\hat{\gamma}(\theta) = p(\theta) \prod_{i=1}^{n} p(x_i|\theta, z_i) \tag{13}$$

where each z_i is drawn independently given θ from (12). Note that in the case of logistic regression, if $b_i(\theta)$ is chosen to be the quadratic lower bound given in (MacLaurin and Adams, 2014), its integral I_θ is a Gaussian integral and can thus be computed. Finally, similarly to the Firefly algorithm of MacLaurin and Adams (2014), the number of evaluations of the likelihood per iteration is nI_θ, loosely speaking.

Although the pseudo-marginal variant of Firefly we propose has the disadvantage of requiring the integrals I_θ to be tractable, it comes with two advantages. First, the sampling of z does not require to evaluate the likelihood at all. If computing all bounds does not become a bottleneck, this avoids the need to explicitly state a resampling fraction at the risk of augmenting the variance of the likelihood estimator. Second, the properties of this variant are easier to understand, as it is a 'standard' pseudo-marginal MH and hence the results from Section 4.1 apply. In particular, although it has the correct target distribution, the asymptotic variance of ergodic averages is inflated compared to the ideal algorithm.

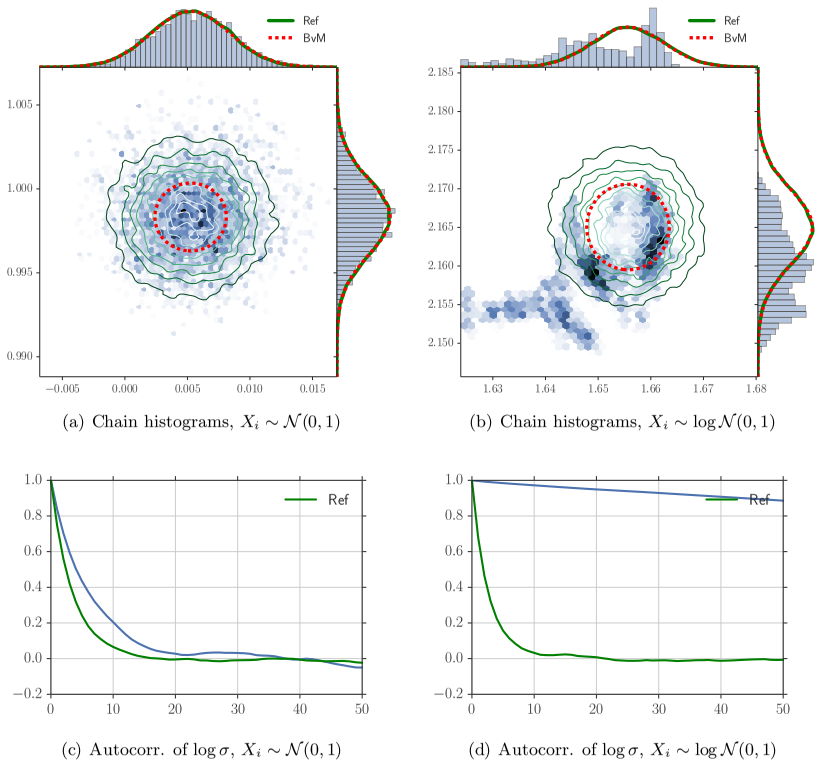
As explained in Section 4.1, we consider the variance of the log likelihood estimator.

Proposition 2 *Let $\theta \in \Theta$. With the notations introduced in Section 4.3,*

$$\mathrm{Var}_z \left[\sum_{i=1}^{n} \log p(x_i|\theta, z_i)\right] = I_\theta(1 - I_\theta) \sum_{i=1}^{n} \log^2 \left[\frac{I_\theta}{1 - I_\theta} \left(e^{\ell_i(\theta) - b_i(\theta)} - 1\right)\right] \tag{14}$$

The proof of Proposition 2 can be found in Appendix B. Proposition 2 can be interpreted as follows: the variance is related to how tight the bound is. In general, obtaining a variance of order 1 will only be possible if *most* bounds $b_i(\theta)$ are very tight, and the bigger n, the tighter the bounds have to be. These conditions will typically not be met when a fixed fraction of "outlier" x_i's correspond to untight bounds.

We give the results of the original Firefly MH on our running Gaussian and log normal examples in Figure 3. We bound each $\ell_i(\theta)$ using a 2nd order Taylor expansion at the MLE and the Taylor-Lagrange inequality, see Section 7.2.1 for further details. This bound is very tight in both cases, so that we are in the favourable case where only a few components of z are 1 at each iteration, and the number of likelihood evaluations per full joint iteration is thus roughly the fraction of points for which z_i has been resampled. We chose the fraction of resampled points to be 10% here, and initialized z to have 10% of ones. Trying smaller fractions led to very slowly mixing chains even for the Gaussian example. Estimating the number of likelihood evaluations per full joint iteration as the sum of the number of resampled z_i's and the number of "bright" points, we obtained in both the Gaussian and lognormal case an almost constant number of likelihood evaluations close to 10%, so that only a few points are bright. This can be explained by the tightness of the Taylor bound, which leads Firefly MH to almost exclusively replace the evaluation of the likelihood by that of the Taylor bound. Finally, unlike the other algorithms we applied, we observed that a bad choice of the initial value of z can easily take θ out of the posterior mode. To be fair, we thus discarded the first 1 000 iterations as a burn-in before plotting.



(a) Chain histograms, $X_i \sim \mathcal{N}(0,1)$

(b) Chain histograms, $X_i \sim \log\mathcal{N}(0,1)$

(c) Autocorr. of $\log\sigma$, $X_i \sim \mathcal{N}(0,1)$

(d) Autocorr. of $\log\sigma$, $X_i \sim \log\mathcal{N}(0,1)$

Figure 3: Results of 10 000 iterations of Firefly MH (MacLaurin and Adams, 2014) on our Gaussian and lognormal running examples. See Section 4.3 and the caption of Figure 2 for details.

As expected, the algorithm behaves erratically in the lognormal case, as failure to attempt a flip of each z_i draws the μ-component of the chain towards the few large values of $(x_i - \mu)^2$ which are bright. Since the bright points are rarely updated, the chain mixes very slowly.

5. Other exact approaches

Other exact approaches have been proposed, which do not rely on pseudo-marginal MH.

5.1 Forgetting about acceptance probability: stochastic approximation approaches

Welling and Teh (2011) proposed an algorithm based on stochastic gradient Langevin dynamics (SGLD). This is an iterative algorithm which at iteration $k+1$ uses the following update rule

$$\theta_{k+1} = \theta_k + \frac{\epsilon_{k+1}}{2}\left[\nabla \log p(\theta) + \frac{n}{t}\sum_{i=1}^{t} \nabla \log p(x^*_{i,k}|\theta)\right] + \sqrt{\epsilon_{k+1}}\eta_{k+1}, \tag{15}$$

(ϵ_k) is a sequence of time steps, (η_k) are independent $\mathcal{N}(0, I_d)$ vectors and

$$\frac{n}{t}\sum_{i=1}^{t} \nabla \log p(x^*_{i,k}|\theta)$$

is an unbiased estimate of the score computed at each iteration using a random subsample $\left\{x^*_{i,k}\right\}$ of the observations. This approach is reminiscent of the Metropolis-adjusted Langevin algorithm (MALA; Robert and Casella 2004, Section 7.8.5), where the proposal given by

$$\theta' = \theta + \frac{\epsilon}{2}\left[\nabla \log p(\theta) + \sum_{i=1}^{n} \nabla \log p(x_i|\theta)\right] + \sqrt{\epsilon}\eta,$$

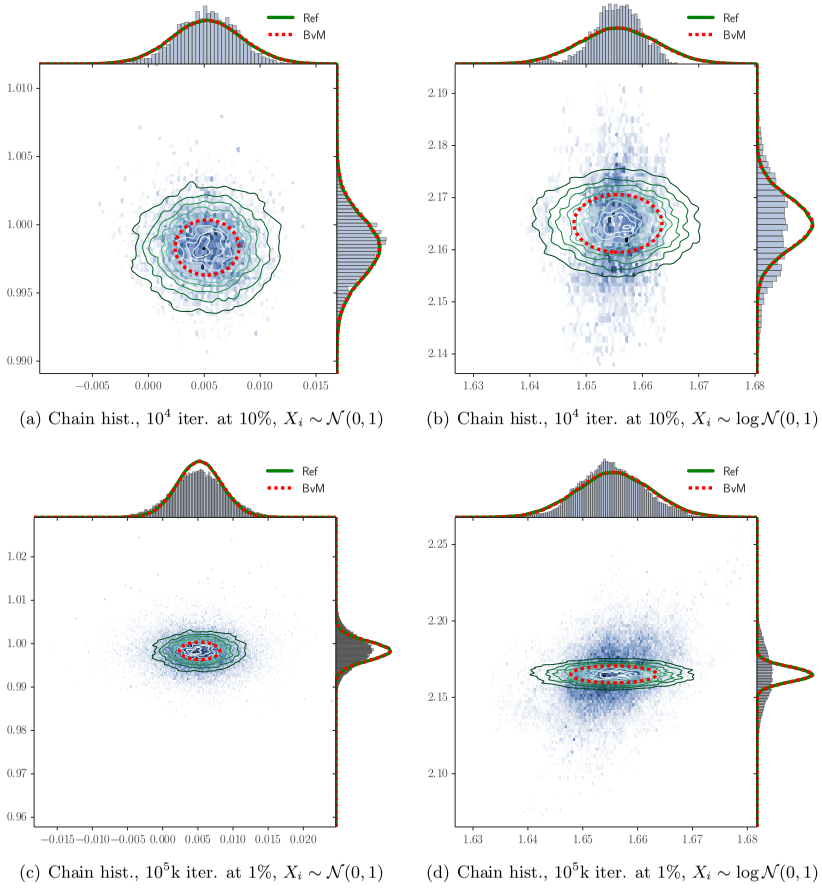
is used in an MH acceptance step, where $\epsilon \sim \mathcal{N}(0, I_d)$. The point of Welling and Teh (2011) is that if one suppresses the MH acceptance step, computes an unbiased estimate of the score but introduces a sequence of stepsizes (ϵ_k) that decreases to zero at the right rate, then

$$\left(\sum_{k=0}^{N_{\text{iter}}} \epsilon_k\right)^{-1} \sum_{k=0}^{N_{\text{iter}}} \epsilon_k \delta_{\theta_k}$$

is an approximation to π. The algorithm has been analyzed recently in (Teh et al., 2016), where it has been established that it provides indeed a consistent estimate of the target. Additionally, a central limit theorem holds with convergence rate $N_{\text{iter}}^{-1/3}$, which is slower than the traditional Monte Carlo rate $N_{\text{iter}}^{-1/2}$. It is yet unclear how SGLD compares to other subsampling schemes in theory: it may require a smaller fraction of the dataset per iteration, but more iterations are needed to reach the same accuracy.

In practice, we show the results of SGLD on our two running examples in Figure 5.1. The stepsize ϵ_k is chosen proportional to $k^{-1/3}$, following the recommendations of Teh et al. (2016). We show the results of two choices for the subsample size t: 10% and 1% of the data, with respectively 10 000 and 100 000 iterations, so that both runs amount to the same 10% fraction of the budget of the vanilla MH in Figure 2. Both runs are still far from convergence on the lognormal example: subsampling draws the chain away from the support of the posterior, and one has to wait for smaller stepsizes to avoid overconfident moves. But then, the variance of the final estimate gets bigger. Constant stepsizes lead to comparable results (not shown).

Finally, we note that subsampling for Hamiltonian Monte Carlo (HMC; Duane et al., 1987) has also been recently considered. Chen et al. (2014) propose a modification of



(a) Chain hist., 10^4 iter. at 10%, $X_i \sim \mathcal{N}(0,1)$

(b) Chain hist., 10^4 iter. at 10%, $X_i \sim \log\mathcal{N}(0,1)$

(c) Chain hist., 10^5k iter. at 1%, $X_i \sim \mathcal{N}(0,1)$

(d) Chain hist., 10^5k iter. at 1%, $X_i \sim \log\mathcal{N}(0,1)$

Figure 4: Results of SGLD (Welling and Teh, 2011) on our Gaussian and lognormal running examples. See Section 5.1 and the caption of Figure 2 for details.

HMC that is inspired by the SGLD with decreasing stepsize of Welling and Teh (2011), while Betancourt (2015) explores why naive approaches suffer from unacceptable biases. The algorithm of Chen et al. (2014) is however a heuristic, which further relies on the subsampling noise being Gaussian. As demonstrated in (Bardenet et al., 2014) and in Section 6.2, relying on a Gaussian noise assumption can yield arbitrarily poor performance when this assumption is violated.

5.2 Delayed acceptance

Banterle et al. (2015) remarked that if we decompose the acceptance ratio in a product of positive functions

$$\alpha(\theta,\theta') = \prod_{i=1}^{B} \rho_i(\theta,\theta')$$

then the MH-like algorithm that accepts the move from θ to θ' with probability

$$\prod_{i=1}^{B} \min\left[\rho_i(\theta,\theta'),1\right]$$

still admits π as invariant distribution. In practice, in the case of tall data, we can divide the dataset into B batches and use for example

$$\rho_i\left(\theta,\theta'\right) = \frac{p(\theta')^{1/B} p(\mathbf{x}_i|\theta') q(\theta|\theta')}{p(\theta)^{1/B} p(\mathbf{x}_i|\theta) q(\theta'|\theta)}.$$

This allows us to reject candidate θ' without having to compute the full likelihoods and the calculations of $\rho_i(\theta,\theta')$ can be done in parallel. However, as remarked by Banterle et al. (2015), the resulting Markov chain has a larger asymptotic variance σ^2_{lim} in (4) than the original MH, and it does not necessarily inherit the ergodicity of the original MH. Furthermore, by construction, every accepted point has to be evaluated on the whole dataset, and the average proportion of data points used is thus lower bounded by the acceptance rate of the algorithm, which in practice is often around 25%. Overall, it is an easy-to-implement feature that does not add any bias, but its benefits are inherently limited, and speed of convergence might be affected.

5.3 Piecewise deterministic Markov chain Monte Carlo methods

The Bouncy Particle Sampler (BPS) is an MCMC algorithm where the target distribution of interest is explored using a continuous-time piecewise-deterministic process. In other words, the state of the Markov process moves along straight lines around the space and, when facing a high energy barrier, it is not rejected but its path is modified by bouncing against this barrier. The BPS was originally proposed in the physics literature by Peters and de With (2012). Several generalizations of this method have then been proposed by Bouchard-Côté et al. (2015). In the context of tall data, in particular, Bouchard-Côté et al. (2015) show that one can use subsampling and preserve the correct target distribution. This has also been noticed by Bierkens et al. (2016), who propose an algorithm related to BPS

but using an alternative bouncing mechanism. However, these approaches also suffer from severe limitations.

First, for subsampling to preserve the invariant distribution, the statistical model under consideration needs to satisfy strong regularity conditions, which have only been established for logistic regression; see (Bouchard-Côté et al., 2015) and (Bierkens et al., 2016). These conditions can be relaxed, but at the price of introducing a bias that is difficult to control (Pakman et al., 2016). Second, it is shown in (Bouchard-Côté et al., 2015, Section 4.6) that, although these subsampling BPS algorithms might have a cost of $\mathcal{O}(1)$ per iteration instead of $\mathcal{O}(n)$ for standard MH, the asymptotic variance of the corresponding averages is empirically of order $\mathcal{O}(n)$ instead of $\mathcal{O}(1)$ for a correctly scaled random walk MH. Hence the gains brought by these approaches over a correctly scaled random walk MH are debatable.

6. Approximate subsampling approaches

In this Section, we consider again subsampling approaches where, at each MH iteration, a subset of data points is used to approximate the acceptance ratio (5). As mentioned in Section 4.2, it is very simple to obtain an unbiased estimator of the log-likelihood $n\ell(\theta)$ based on random samples $x_1^*, \ldots, x_t^*$ from the dataset $\mathcal{X}$; see (7). Similarly, one can also easily obtain an unbiased estimator of the average log likelihood ratio $[\ell(\theta') - \ell(\theta)]$

$$\Lambda_t^*(\theta, \theta') \triangleq \frac{1}{t} \sum_{i=1}^{t} \log \frac{p(x_i^*|\theta')}{p(x_i^*|\theta)}. \tag{16}$$

Note that unlike the exact approaches of Sections 4 and 5, the methods reviewed in Section 6 do not attempt to sample exactly from π, but just from an approximation to π.

6.1 Naive subsampling

The first approach one could try is to only use a random fixed proportion of data points to estimate π at any newly drawn θ. We highlight that this leads to a nontrivial mixture target that is hard to interpret, where all subsampled posteriors appear, suitably rescaled. Assume that at each new θ drawn in an MH run, we draw n independent Bernoulli variables and let

$$\hat{\ell}(\theta) = \frac{1}{n} \sum_{i=1}^{n} \frac{z_i}{\lambda} \ell_i(\theta) \tag{17}$$

be an unbiased estimator of the average log likelihood $\ell(\theta)$, where $z_i \sim B(1, \lambda)$ i.i.d. Now one could think of plugging estimates $\hat{\gamma}(\theta) = p(\theta) e^{n\hat{\ell}(\theta)}$ in Steps 2 and 3 of MH in Figure 1. However, as $\hat{\gamma}(\theta)$ is not an unbiased estimator of $\gamma(\theta)$, this algorithm samples from a target distribution which is proportional to $p(\theta)\mathbb{E}e^{n\hat{\ell}(\theta)} \neq \gamma(\theta)$, where the expectation is w.r.t the distributions of the Bernoulli random variables $\{z_i\}$. Now

$$\begin{aligned} \mathbb{E}e^{n\hat{\ell}(\theta)} &= \prod_{i=1}^{n} \left[\lambda e^{\ell_i(\theta)/\lambda} + (1-\lambda)\right] \\ &= \sum_{r=0}^{n} \sum_{\#I_r = r} \lambda^r (1-\lambda)^{n-r} p(x_{I_r}|\theta)^{1/\lambda}, \end{aligned}$$

where I_r denotes a set of r distinct indices in $\{1, \ldots, n\}$, $x_{I_r} = \{x_i; i \in I_r\}$, and with the convention $p(x_\emptyset|\theta) = 1$. Each subsampled likelihood contributes to the target, exponentiated to the power $1/\lambda$, resulting in a nontrivial mixture of rescaled data likelihood terms. To further simplify, assume $p(x_{I_r}|\theta) \approx p_r(\theta)$ for each set of indices I_r, that is, the variance of the likelihood under subsampling is small, then

$$\mathbb{E}e^{n\hat{\ell}(\theta)} \approx \sum_{r=0}^{n} C_n^r \lambda^r (1-\lambda)^{n-r} p_r(\theta)^{1/\lambda} = \mathbb{E}_{R\sim B(n,\lambda)} p_R^{1/\lambda}(\theta), \tag{18}$$

where $B(n, \lambda)$ denotes the binomial distribution with parameters n and λ. Noticing that $p_r(\theta)$ is roughly exponentially decreasing in r, the values or r that are larger than the mode of the binomial probability mass function are unlikely to contribute a lot to (18). The largest subsample size contributing to (18) is thus roughly $n\lambda$, and the power $1/\lambda$ makes this term of the same scale as $p(x_1, \ldots, x_n|\theta)$. Broadly speaking, subsampling has a "broadening" effect due to the contribution of the likelihoods of small subsamples.

Alternately, if one starts with the biased estimator of the average log likelihood

$$\tilde{\ell}(\theta) = \frac{1}{n} \sum_{i=0}^{n} z_i \ell_i(\theta), \tag{19}$$

instead of (17) one ends up with

$$\begin{aligned} \mathbb{E}e^{n\tilde{\ell}(\theta)} &= \prod_{i=1}^{n} \left[\lambda e^{\ell_i(\theta)} + (1-\lambda)\right] \\ &= \sum_{r=0}^{n} \sum_{\#I_r = r} \lambda^r (1-\lambda)^{n-r} p(x_{I_r}|\theta) \\ &\approx \mathbb{E}_{R\sim B(n,\lambda)} p_R(\theta). \end{aligned} \tag{20}$$

Again, all subsampled likelihoods contribute, but without being further exponentiated. Still, the result is a much broadened target, as values of r that are larger than $n\lambda$ are unlikely to contribute a lot. In this case, the broadening effect of subsampling is not only due to the contribution of small subsamples, but also to the absence of bias correction in (19).

We have thus seen that naive subsampling is nontrivial tempering, so that the target is not preserved. Additionally, as mentioned in Section 4.1, the variance of the log likelihood estimator needs to be kept around a constant, 1 or 3 depending on hypotheses, in order for pseudo-marginal MH to be efficient. This means that λ should be such that

$$\frac{(1-\lambda)}{\lambda} \sum_{i=1}^{n} \ell_i(\theta)^2$$

is of order 1 in the case of (17). This entails that λ should be close to 1, so that there is no substantial gain in terms of number of likelihood evaluations. In the case of (19),

$$\lambda(1-\lambda) \sum_{i=1}^{n} \ell_i(\theta)^2$$

can be of order 1 if $\lambda \sim n^{-1}$. But then the leading terms in the mixture target (20) will be the subsampled likelihoods corresponding to small subsamples, so that the target will be very different from the actual target π.

Overall, naive subsampling is a very poor approach. However, it allows us to identify the main issues a good subsampling approach should tackle: guaranteeing its target, not loosing too much convergence speed compared to the MH algorithm based on the true likelihood, and cutting the likelihood evaluation budget. As shown in (Bardenet et al., 2014), the first point is an algorithmic design issue, while the last two points are related to controlling the variance of the log likelihood ratios.

6.2 Relying on the CLT

Several authors have appealed to the central limit theorem to justify their assumption that the average subsampled log likelihoods and log likelihood ratios in (7) and (16) are Gaussianly distributed.

If the distribution of the log likelihood ratio estimate minus the true log-likelihood ratio was normal with known variance σ^2 and mean $-\sigma^2/2$, Ceperley and Dewing (1999) have proposed an MH with a corrected acceptance ratio that is exact, i.e., that still targets π. However, this distribution is never exactly normal and the variance of the noise is unknown. In Nicholls et al. (2012), it is proposed to estimate the noise variance and the author propose a heuristic argument to explain why this inexact chain might give reasonable approximate results, but the Gaussian assumption remains crucial there. As shown in (Bardenet et al., 2014) and in Figures 5 and 6 of this paper, this assumption can be easily violated when subsampling tall data if the log likelihood ratios $\ell_i(\theta') - \ell_i(\theta)$ are heavy-tailed. Missing log likelihoods in the tails will lead to erroneous decisions, and uncontrolled results.

6.2.1 A pseudo-marginal approach with variance reduction under Gaussian assumption

Quiroz et al. (2014) propose a methodology to use MH for tall data which also relies on the assumption that the log-likelihood estimator is Gaussian with mean $\ell(\theta)$, for every θ. By introducing a bias correction providing an approximately unbiased estimate of the likelihood, this corresponds to a pseudo-marginal MH algorithm whose target distribution is proportional to $p(\theta)\mathbb{E}e^{n\hat{\ell}(\theta)-\hat{b}(\theta)}$, where $\hat{b}(\theta)$ is an estimate of the bias $b(\theta)$ satisfying $\mathbb{E}e^{n\hat{\ell}(\theta)} = e^{n\ell(\theta)+b(\theta)}$. They rightly notice that, ideally, if one wants to keep the variance of average subsampled log likelihoods small, one should not subsample data points with or without replacement, but one should rather perform importance sampling with the weight of data point i being proportional to $|\ell_i(\theta)|$. While this variance reduction approach obviously defeats the purpose of subsampling, Quiroz et al. (2014) propose to use as weights an approximation of the log-likelihood, based e.g. on a Gaussian process or splines trained on a small subset of computed likelihoods $\ell_i(\theta)$. Finally, a heuristic to adaptively choose the size of the total subsample so as to keep the variance of the log likelihood controlled is proposed. The method is demonstrated to work on a bivariate probit model using only 10% of the full dataset. However, as a general purpose method, it suffers from two limitations. First, it is based on Gaussian assumptions, which can be unreasonable as noted above and it is unclear whether it will be robust to these CLT approximations not being valid. Second,

the proposed importance sampling step requires to learn a good proxy for $x \mapsto p(x|\theta)$ for each θ drawn during the MCMC run. The fitted proxies should thus be cheap to train and evaluate, but at the same time accurate if any variance reduction is to be obtained.

6.2.2 Adaptive subsampling with T-tests

Still assuming the noise of the log likelihood is Gaussian, given a drawn $\theta \in \Theta$, one can try to adaptively choose the size of the subsample $\{x_1^*, \dots, x_t^*\}$ to be used in the unbiased estimators (7) or (16), so as to take the correct acceptance decision with high probability. Upon noting that the MH acceptance decision is equivalent to deciding whether $\log \alpha(\theta, \theta') > u$, or equivalently

$$\frac{1}{n}\sum_{i=1}^{n} \log \frac{p(x_i|\theta')}{p(x_i|\theta)} > \frac{1}{n}\log u - \frac{1}{n}\log\left[\frac{p(\theta')q(\theta|\theta')}{p(\theta)q(\theta'|\theta)}\right] \tag{21}$$

with $u \sim \mathcal{U}_{[0,1]}$ drawn beforehand, statistical tests can be used to assert whether (21) holds with a given level of "confidence". As far as we are aware, Bulgak and Sanders (1988) were the first to consider such a procedure. They used it in a simulated annealing algorithm maximizing a function defined as an expectation w.r.t a probability distribution, and approximated using Monte Carlo. Simulated annealing is a simple non-homogeneous variant of the MH algorithm where the the target distribution is annealed over the iterations. The same application received more attention later (Alkhamis et al., 1999; Wang and Zhang, 2006). Applied to the standard MH, the method has been considered by Singh et al. (2012), and more recently by Korattikara et al. (2014) specifically for tall data. Korattikara et al. (2014) propose an MH-like algorithm called *Austerity MH* that incorporates a sequential T-test to check (21) for each pair (θ, θ'), thus relying on *several* CLTs. They demonstrate dramatic reductions in the average number of subsamples used per MCMC iteration on particular applications. However, as noted in (Korattikara et al., 2014; Bardenet et al., 2014), the results can be arbitrarily far from the original MH when the CLT approximations are not valid.

We show the results of 10 000 iterations of Austerity MH on our two running examples in Figure 5. The parameters are $\epsilon = 0.05$, corresponding to the p-value threshold in the aforementioned T-test, and an initial subsample size of 100 at each iteration. In the Gaussian case, the posterior is rightly centered, but is slightly too wide. This is a tempering effect due to too small subsamples, while the CLT-based Student approximation seems reasonable, as shown in Figure 6. In the lognormal case, the departure of the chain from the actual posterior is more remarkable, and relatedly the CLT approximations of Austerity MH are inaccurate for the chosen initial subsample size of 100, as we demonstrate in Figure 6. This explains the strong mismatch of the chain and the posterior in Figure 5(b). The standard deviation of the fitted Gaussian is largely underestimated, due to small subsamples which do not include enough of the tails of the log likelihood ratios, which coincide with the tails of $\mathcal{X}$. Finally, the reductions in the number of samples needed per iteration are quite interesting: half of the iterations require less than 4% of the dataset for the lognormal case, but this is at the price of a large error in the posterior approximation. Augmenting the initial size of the subsample will likely make the CLT approximations tighter, but there is no generic answer as to which size to choose: any fixed choice will fail on an example where the log likelihood ratios have heavy enough tails. In both the Gaussian and the

lognormal example, it is actually safer to go with the Bernstein-von Mises approximation, which costs little more than a run of stochastic gradient descent, and only requires one CLT approximation, for a sample of size $n \gg 1$. This illustrates the danger of using CLT-based approximations for small sample sizes, which is related to asymptotic arguments on small batches in Section 3.

Overall, CLT-based approaches to MH with tall data lead to heuristics with interesting reductions in the number of samples used, but they have little theoretical backing so far and they are not robust to the involved CLT approximations being inaccurate. We note also that the CLT is assumed to provide a good approximation for the log likelihood or log likelihood ratio for *any* $\theta, \theta' \in \Theta$, which amounts to more than one Gaussian assumption. The approaches in this section should thus be applied with care. As a minimal sanity-check, we recommend using tests of Gaussianity across $\Theta \times \Theta$ to make sure the CLT assumptions are realistic. Note that even then, there is no guarantee the above algorithms have π for target, if any.

6.3 Exchanging acceptance noise for subsampling noise

This section is an original contribution, which illustrates a way to obtain subsampling algorithms with guarantees under weaker assumptions than Gaussianity. This approach is impractical, but it is of methodological and illustrative interest. First it illustrates a potentially useful technique to *take advantage of* subsampling noise. Second, it is our first illustration of the seemingly inevitable $\mathcal{O}(n)$ average number of subsamples required per MCMC iteration as soon as we do not use any CLT-based approximation and require theoretical guarantees.

Let $\theta, \theta' \in \Theta$, and let $x_1^*, \ldots, x_t^*$ be drawn independently with replacement from $\mathcal{X}$. Let $\Lambda_t^*(\theta, \theta')$ be the average subsampled log likelihood ratio defined in (16). Now, we remark that MH has some inherent noise in its acceptance decision (21), encapsulated by the uniform variable $u \sim \mathcal{U}[0, 1]$. Why, then, not rely on the subsampling noise to guarantee exploration, and accept a move if and only if

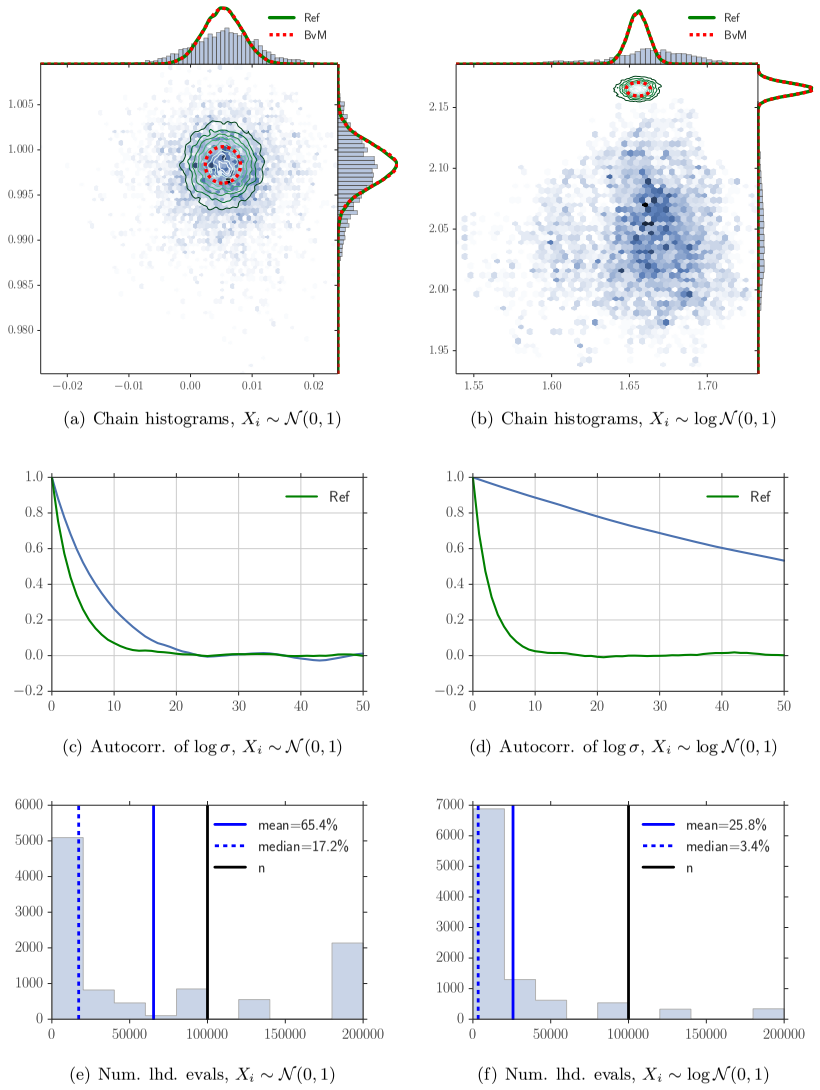
$$\Lambda_t^*(\theta, \theta') + \frac{1}{n} \log \left[\frac{p(\theta')q(\theta|\theta')}{p(\theta)q(\theta'|\theta)} \right] > 0 \tag{22}$$

instead of (21)? This idea has been first used by Branke et al. (2008) to develop heuristics for simulated annealing in the presence of noise. We formalize this argument here in the context of subsampling. For the sake of simplicity, assume for a moment we have a flat prior and a symmetric proposal, so that (22) becomes

$$\Lambda_t^*(\theta, \theta') > 0.$$

We do not assume that the $\Lambda_t^*(\theta, \theta')$'s are Gaussianly distributed, but we make the parametric assumption that the second and third absolute moments σ^2 and ρ of $-\log p(x_i^*|\theta') + \log p(x_i^*|\theta)$ are known and independent of θ, θ'. Applying the Berry-Esseen inequality (van der Vaart and Wellner, 1996) to the variables $-\log p(x_i^*|\theta') + \log p(x_i^*|\theta)$ yields

$$\left| \mathbb{P}(-\Lambda_t^*(\theta, \theta') \leq u) - \Phi\left(\frac{u + \Lambda_n(\theta, \theta')}{\sigma/\sqrt{t}} \right) \right| \leq \frac{K(\sigma, \rho)}{\sqrt{t}} \tag{23}$$



(a) Chain histograms, $X_i \sim \mathcal{N}(0, 1)$

(b) Chain histograms, $X_i \sim \log\mathcal{N}(0, 1)$

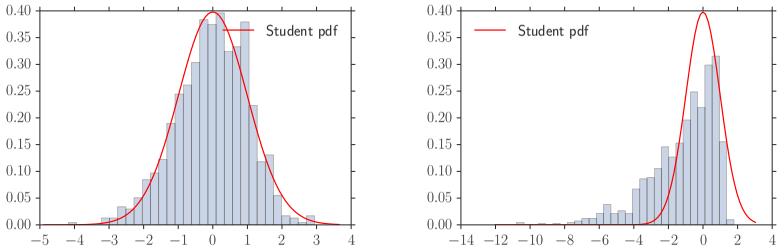
(c) Autocorr. of $\log\sigma$, $X_i \sim \mathcal{N}(0, 1)$

(d) Autocorr. of $\log\sigma$, $X_i \sim \log\mathcal{N}(0, 1)$

(e) Num. lhd. evals, $X_i \sim \mathcal{N}(0, 1)$

(f) Num. lhd. evals, $X_i \sim \log\mathcal{N}(0, 1)$

Figure 5: Results 10 000 iterations of Austerity MH (Korattikara et al., 2014). See Section 6.2 and the caption of Figure 2 for details.



(a) Student around MAP, $X_i \sim \log\mathcal{N}(0,1)$ (b) Student around MAP, $X_i \sim \log\mathcal{N}(0,1)$

Figure 6: Histogram of 1000 realizations of the Student statistic required in Austerity MH, taken at $\theta = \theta_{\text{MAP}}$ and $\theta' \sim q(\cdot|\theta)$. The theoretical Student pdf is plotted in red.

for any $u \in \mathbb{R}$, where Φ is the cdf of a $\mathcal{N}(0,1)$ variable, and

$$\Lambda_n(\theta,\theta') \triangleq \frac{1}{n}\sum_{i=1}^{n} \log \frac{p(x_i|\theta')}{p(x_i|\theta)}$$

is the average log likelihood ratio. When $u = 0$, (23) yields

$$\left|\mathbb{P}(\Lambda_t^*(\theta,\theta') \geq 0) - \Phi\left(\frac{\Lambda_n(\theta,\theta')}{\sigma/\sqrt{t}}\right)\right| \leq \frac{K(\sigma,\rho)}{\sqrt{t}}. \tag{24}$$

Now let $C, \lambda > 0$ be such that for any $x \in \mathbb{R}$,

$$\left|\Phi(x) - \frac{1}{1+e^{-\lambda x}}\right| \leq C.$$

Bowling et al. (2009) for instance, empirically found $C = 0.0095$ and $\lambda = 1.702$. Combining this bound with (24), we obtain

$$\left|\mathbb{P}(\Lambda_t^*(\theta,\theta') \geq 0) - \frac{1}{1+e^{-\frac{\lambda\Lambda_n(\theta,\theta')}{\sigma/\sqrt{t}}}}\right| \leq C + \frac{K(\sigma,\rho)}{\sqrt{t}}.$$

Hence, the acceptance probability of an algorithm that would accept the move from θ to θ' by checking whether $\Lambda^*(\theta,\theta') > 0$ is close to the acceptance probability of an MCMC algorithm with a Baker acceptance criterion (Robert and Casella, 2004, Section 7.8.1) that targets π^β with temperature $\beta = \frac{\lambda\sqrt{t}}{n\sigma}$. Arguments such as (Bardenet et al., 2014, Lemma 3.1, Proposition 3.2) could then help concluding that the distance between the kernels of both Markov chains is controlled, which would yield positive ergodicity results, in the line of (Bardenet et al., 2014, Proposition 3.2). This reasoning shows again a close relation between

subsampling and tempering, as in Section 6.1, with a clear link between the variance of the subsampled log likelihood ratios and the temperature.

Now, from a practical point of view, in simple applications such as logistic regression, σ is of the order of $\|\theta - \theta'\|$, which in turn should be of order $\mathcal{O}_p(n^{-1/2})$ if the MCMC proposal is a Gaussian random walk with covariance similar to that of π, see Bardenet et al. (2014). This means that t has to be of order n for the temperature β to be of order 1, and this approach is thus bound to use $\mathcal{O}(n)$ subsamples per iteration! In conclusion, robustness to non-Gaussianity leads to requiring a fixed proportion of the whole dataset on average, even in the favourable case when one controls the first three moments of the subsampling noise and one swaps subsampling noise for the inherent MCMC acceptance noise.

6.4 Confidence samplers

In (Bardenet et al., 2014), we proposed a controlled approximation of the acceptance decision (21). Indeed, let us fix θ, θ' and momentarily assume that $x \mapsto \log[p(x|\theta')/p(x|\theta)]$ was Lipschitz with known constant. Then, having observed the log likelihood ratio at some points $\{x_i^*, i = 1, \ldots, t\} \subset \mathcal{X}$, one could build a lower and an upper bound for the complete log likelihood ratio

$$\frac{1}{n}\sum_{i=1}^{n} \log\left[\frac{p(x_i|\theta')}{p(x_i|\theta)}\right],$$

simply by associating each x_i with the nearest point among $\{x_1^*, \ldots, x_t^*\}$. These bounds could be refined by augmenting the set of observed log likelihoods ratios, until eventually one knows for sure whether (21) holds.

Now, concentration inequalities allow softer bounds and require less than this Lipschitz assumption. If one knows a bound for the range

$$C_{\theta,\theta'} \triangleq \max_{i=1}^{n} \left|\log\left[\frac{p(x_i|\theta')}{p(x_i|\theta)}\right]\right|, \tag{25}$$

then concentration inequalities such as Hoeffding's or Bernstein's, yield confidence bounds $c_t(\delta)$ such that

$$\mathbb{P}\left(\left|\frac{1}{t}\sum_{i=1}^{t} \log\left[\frac{p(x_i^*|\theta')}{p(x_i^*|\theta)}\right] - \frac{1}{n}\sum_{i=1}^{n} \log\left[\frac{p(x_i|\theta')}{p(x_i|\theta)}\right]\right| > c_t(\delta)\right) \geq 1 - \delta, \tag{26}$$

where the probability is taken over $x_1^*, \ldots, x_t^*$ drawn uniformly from $\mathcal{X}$, with or without replacement. Borrowing from the bandit literature, we explain in (Bardenet et al., 2014) how to leverage such confidence bounds to automatically select a subsample size T such that the right MH acceptance decision is taken with a user-specified probability $1 - \delta$. Note that for our algorithm to bring any improvement over the ideal MH, the range (25) must be cheap to compute, i.e. cheaper than $\mathcal{O}(n)$. This is the case for logistic regression, for example, but it is the major limitation of the approach in Bardenet et al. (2014). We showed in (Bardenet et al., 2014, Proposition 3.2) that if the ideal MH sampler is uniformly ergodic then the resulting algorithm inherits the uniform ergodicity of the ideal MH sampler, with a convergence speed that is within $\mathcal{O}(\delta)$ of that of the ideal MH. Importantly, we showed that our sampler then admits a limiting distribution, which is also within $\mathcal{O}(\delta)$ of π. Uniform

ergodicity is a very strong assumption and it would be worth extending these results to the geometrically ergodic scenario. There has recently been work in this direction (Alquier et al., 2016; Pillai and Smith, 2014; Rudolf and Schweizer, 2015).

On the negative side, we demonstrated in (Bardenet et al., 2014) that vanilla confidence samplers still require $\mathcal{O}(n)$ samples at each iteration at equilibrium, where the proportionality constant is the variance of the log likelihood ratio under subsampling. This statement relies on the leading term in $c_t(\delta)$ being of order $t^{-1/2}$. In practice, the results of the vanilla confidence sampler on our running examples are shown in Figure 7. We set $\delta = 0.1$ and we place ourselves in the favourable scenario where the algorithm has access to the actual range of each log likelihood ratio. The number of likelihood evaluations is estimated as follows: we take by default *twice* the detected value T for the subsample size in general, but only *once* when the previous iteration required computing all n likelihoods at the current state of the chain. Still, even in these favourable conditions, the algorithm basically requires essentially the whole dataset at each iteration.

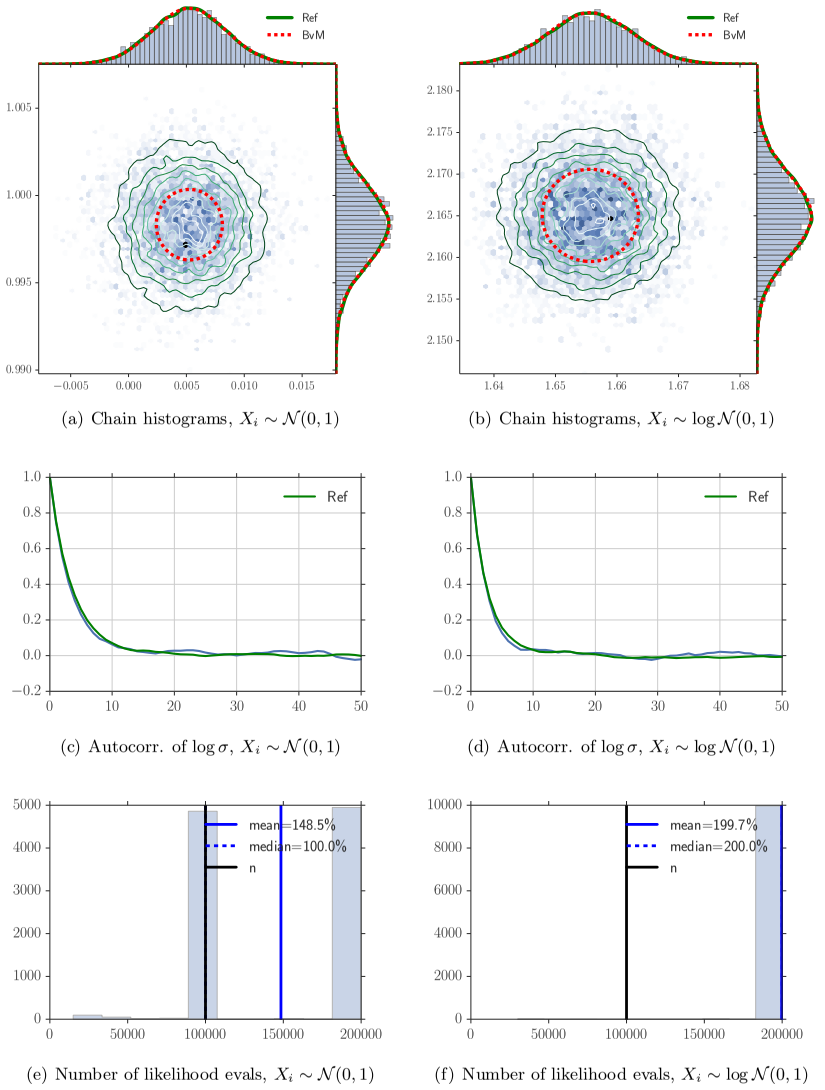
Concentration inequalities are "worst-case" guarantees, and the theoretical results come at the price of a smaller reduction in the number of samples required. When the target is locally Gaussian, e.g. when Bernstein-von Mises yields a good approximation, there is potentially a lot to be gained, as empirically demonstrated by Korattikara et al. (2014), for example. In the current paper, we propose in Section 7 a modified confidence sampler that can leverage concentration of the target to yield dramatic empirical gains while not sacrificing any theoretical guarantee of the confidence sampler. The basic tool is a cheap proxy for the log likelihood ratio that acts as a control variate in the concentration inequality (26). Using a 2nd order Taylor expansion centered at the maximum of the likelihood – obtained with a stochastic gradient descent for example – allows to replace many likelihood evaluations by the evaluation of this Taylor expansion. Figure 8 shows the results of this new confidence sampler with proxy on our running Gaussian and lognormal examples. Our algorithm outperforms all preceding methods, using almost no sample in the Gaussian case where it *automatically* detects that a quadratic form is enough to represent the log likelihood ratio. Finally, we demonstrate in Sections 7.2.3 and 8 that this new algorithm can require less than $\mathcal{O}(n)$ likelihood evaluations per iteration. Combined with the statements in Bardenet et al. (2014) that each iteration is almost as efficient as the ideal MH, which is further supported by the match of the autocorrelation functions in Figures 8(c) and 8(d), this opens up big data horizons. We give full details on the confidence algorithm with proxy in Section 7.

7. An improved confidence sampler

In this section, we build upon the confidence sampler in (Bardenet et al., 2014) by introducing likelihood proxies, which act as control variates for the individual likelihoods.

7.1 Introducing proxies in the confidence sampler

We start by recalling the pseudocode of the confidence sampler in (Bardenet et al., 2014) in Figure 9, using sampling with replacement and a generic empirical concentration bound $c_t(\delta)$. In practice, one can think of the empirical Bernstein bound of Audibert et al. (2009)



(a) Chain histograms, $X_i \sim \mathcal{N}(0,1)$

(b) Chain histograms, $X_i \sim \log\mathcal{N}(0,1)$

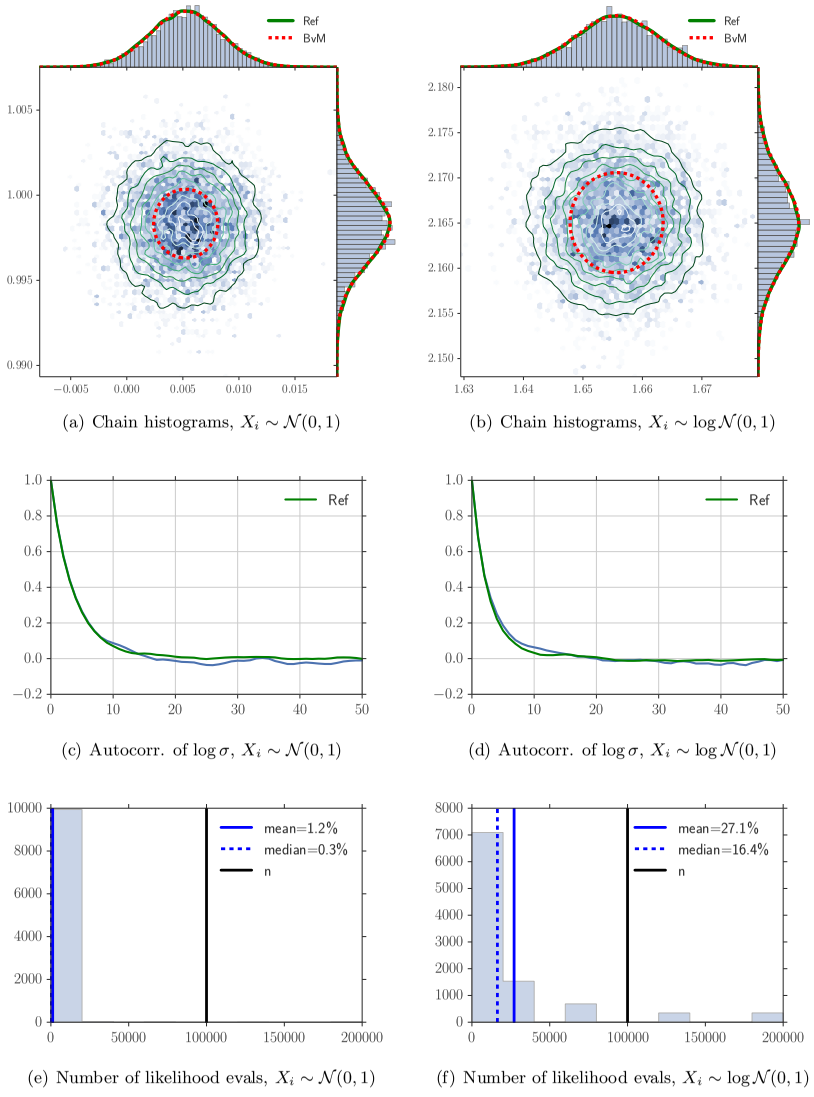
(c) Autocorr. of $\log\sigma$, $X_i \sim \mathcal{N}(0,1)$

(d) Autocorr. of $\log\sigma$, $X_i \sim \log\mathcal{N}(0,1)$

(e) Number of likelihood evals, $X_i \sim \mathcal{N}(0,1)$

(f) Number of likelihood evals, $X_i \sim \log\mathcal{N}(0,1)$

Figure 7: Results of 10 000 iterations of the vanilla confidence sampler (Bardenet et al., 2014), see Section 6.4 and the caption of Figure 2 for details.



(a) Chain histograms, $X_i \sim \mathcal{N}(0,1)$

(b) Chain histograms, $X_i \sim \log\mathcal{N}(0,1)$

(c) Autocorr. of $\log\sigma$, $X_i \sim \mathcal{N}(0,1)$

(d) Autocorr. of $\log\sigma$, $X_i \sim \log\mathcal{N}(0,1)$

(e) Number of likelihood evals, $X_i \sim \mathcal{N}(0,1)$

(f) Number of likelihood evals, $X_i \sim \log\mathcal{N}(0,1)$

Figure 8: Results of 10 000 iterations of the confidence sampler of Section 7 with a single 2nd order Taylor proxy at θ_{MAP}.

$$c_t(\delta) = \hat{\sigma}_{t,\theta,\theta'}\sqrt{\frac{2\log(3/\delta)}{t}} + \frac{6C_{\theta,\theta'}\log(3/\delta)}{t}, \tag{27}$$

where $\hat{\sigma}_{t,\theta,\theta'}$ is the sample standard deviation of the log likelihood ratio

$$\left\{\log\left[\frac{p(x_i^*|\theta')}{p(x_i^*|\theta)}\right], i = 1,\dots,t\right\},$$

and $C_{\theta,\theta'}$ is their range, defined in (25). We emphasize that other choices of sampling procedure and concentration inequalities are valid, as long as they guarantee a concentration like (26). We refer the reader to (Bardenet et al., 2014) for a proof of the correctness of the confidence sampler and implementation details.

The bottleneck for the performance of the confidence sampler was identified in (Bardenet et al., 2014) as the expectation w.r.t. $\pi(\theta)q(\theta'|\theta)$ of the variance of the log likelihood ratio $\log p(x|\theta')/p(x|\theta)$ w.r.t. to the empirical distribution of the observations. We now propose a control variate technique inspired from the Firefly MH of MacLaurin and Adams (2014) to lower this variance down when an accurate and cheap proxy of the log likelihood is known.

We require a proxy for the log likelihood ratio that may decrease the variance of the log likelihood ratio or its range. More precisely, let $\wp_i(\theta,\theta')$ be such that for any $\theta,\theta' \in \Theta$,

1. $\wp_i(\theta,\theta') \approx \ell_i(\theta') - \ell_i(\theta)$
2. $\sum_{i=1}^n \wp_i(\theta,\theta')$ can be computed cheaply.
3. $|\ell_i(\theta') - \ell_i(\theta) - \wp_i(\theta,\theta')|$ can be bounded uniformly in $1 \le i \le n$, and the bound is cheap to compute.

We now simply remark that the acceptance decision (21) in MH is equivalent to checking whether

$$\frac{1}{n}\sum_{i=1}^n\left[\log\frac{p(x_i|\theta')}{p(x_i|\theta)} - \wp_i(\theta,\theta')\right] > \frac{1}{n}\log u - \frac{1}{n}\log\left[\frac{p(\theta')q(\theta|\theta')}{p(\theta)q(\theta'|\theta)}\right] - \frac{1}{n}\sum_{i=1}^n\wp_i(\theta,\theta'). \tag{28}$$

Building the confidence sampler on (28) leads to the same pseudocode as in Figure 9, except that Step 14 is replaced by

$$\Lambda^* \leftarrow \frac{1}{b}\left(t\Lambda^* + \sum_{i=t+1}^b\left[\log\frac{p(x_i^*|\theta')}{p(x_i^*|\theta)} - \wp_i(\theta,\theta')\right]\right),$$

the condition in Step 19 is replaced by

$$\left|\Lambda^* + \frac{1}{n}\sum_{i=1}^n\wp_i(\theta,\theta') - \psi(u,\theta,\theta')\right| \ge c,$$

and the condition in Step 21 becomes

$$\Lambda^* > \psi(u,\theta,\theta') - \frac{1}{n}\sum_{i=1}^n\wp_i(\theta,\theta').$$

ConfidenceSampler$\left(p(x|\theta),\, p(\theta),\, q(\theta'|\theta),\, \theta_0,\, N_{\text{iter}},\, \mathcal{X},\, (\delta_t),\, C_{\theta,\theta'},\right)$

1 **for** $k \leftarrow 1$ **to** N_{iter}
2 $\quad\theta \leftarrow \theta_{k-1}$
3 $\quad\theta' \sim q(.|\theta)$, $u \sim \mathcal{U}_{(0,1)}$,
4 $\quad\psi(u,\theta,\theta') \leftarrow \frac{1}{n}\log\left[u\frac{p(\theta)q(\theta'|\theta)}{p(\theta')q(\theta|\theta')}\right]$
5 $\quad t \leftarrow 0$
6 $\quad t_{\text{look}} \leftarrow 0$
7 $\quad\Lambda^* \leftarrow 0$
8 $\quad\mathcal{X}^* \leftarrow \emptyset$ ▷ *Keeping track of points already used*
9 $\quad b \leftarrow 1$ ▷ *Initialize batchsize to 1*
10 $\quad$Done $\leftarrow$ False
11 $\quad$**while** Done = False **do**
12 $\quad\quad x^*_{t+1}, \ldots, x^*_b \sim_{\text{w/o repl.}} \mathcal{X} \setminus \mathcal{X}^*$ ▷ *Sample new batch without replacement*
13 $\quad\quad\mathcal{X}^* \leftarrow \mathcal{X}^* \cup \{x^*_{t+1}, \ldots, x^*_b\}$
14 $\quad\quad\Lambda^* \leftarrow \frac{1}{b}\left(t\Lambda^* + \sum_{i=t+1}^{b}\log\left[\frac{p(x_i^*|\theta')}{p(x_i^*|\theta)}\right]\right)$
15 $\quad\quad t \leftarrow b$
16 $\quad\quad c \leftarrow c_t(\delta_{t_{\text{look}}})$ ▷ *Use a valid concentration inequality*
17 $\quad\quad t_{\text{look}} \leftarrow t_{\text{look}} + 1$
18 $\quad\quad b \leftarrow n \wedge \lceil \gamma t \rceil$ ▷ *Increase batchsize geometrically, stop when all data have been seen*
19 $\quad\quad$**if** $|\Lambda^* - \psi(u,\theta,\theta')| \geq c$ **or** $t = n$
20 $\quad\quad\quad$Done $\leftarrow$ True
21 $\quad$**if** $\Lambda^* > \psi(u,\theta,\theta')$
22 $\quad\quad\theta_k \leftarrow \theta'$ ▷ *Accept*
23 $\quad$**else** $\theta_k \leftarrow \theta$ ▷ *Reject*
24 **return** $(\theta_k)_{k=1,\ldots,N_{\text{iter}}}$

Figure 9: Pseudocode of the confidence MH from (Bardenet et al., 2014). Note that although we present the algorithm with uniform subsampling *without replacement*, batches can be drawn with or without replacement at Step 12, as long as a valid concentration inequality is used at Step 16. Our contribution is a modification of Steps 14, 19 and 21 to introduce proxies for the log likelihood ratios, see Section 7.

For completeness, we restate here in Proposition 3 that the vanilla confidence sampler inherits the uniform ergodicity of the underlying MH sampler, that its target is within $\mathcal{O}(\delta)$ of π, and that the difference in speed of convergence is also controlled by δ. Let P be the underlying MH kernel, and $\tilde{P}$ the kernel of the confidence sampler described in this section.

Proposition 3 *Let P be uniformly geometrically ergodic, i.e., there exists an integer m, a probability measure ν on $(\Theta, \mathcal{B}(\Theta))$ and $0 \leq \rho < 1$ such that for all $\theta \in \Theta$, $P^m(\theta,\cdot) \geq (1-\rho)\,\nu(\cdot)$. Hence there exists $A < \infty$ such that*

$$\forall\theta \in \Theta, \forall k > 0,\ \|P^k(\theta,\cdot) - \pi\|_{TV} \leq A\rho^{\lfloor k/m \rfloor}. \tag{29}$$

Then there exists $B < \infty$ and a probability distribution $\tilde{\pi}$ on $(\Theta, \mathcal{B}(\Theta))$ such that for all $\theta \in \Theta$ and $k > 0$,

$$\|\tilde{P}^k(\theta,\cdot) - \tilde{\pi}\|_{TV} \leq B[1-(1-\delta)^m(1-\rho)]^{\lfloor k/m \rfloor}. \tag{30}$$

Furthermore, $\tilde{\pi}$ satisfies

$$\|\pi - \tilde{\pi}\|_{TV} \leq \frac{Am\delta}{1-\rho}. \tag{31}$$

Even in the presence of proxies, the proofs of (Bardenet et al., 2014, Lemma 3.1, Proposition 3.2) apply with straightforward modifications, so that we can extend Proposition 3 to the proxy case. The major advantage of this new algorithm is that the sample standard deviation $\hat{\sigma}_{t,\theta,\theta'}$ and range $C_{\theta,\theta'}$ in the concentration inequality (27) are replaced by those of

$$\left\{\log\left[\frac{p(x_i^*|\theta')}{p(x_i^*|\theta)}\right] - \wp_i(\theta,\theta'), i = 1,\ldots,t\right\}.$$

If $\wp_i(\theta,\theta')$ is a good proxy for the log likelihood ratio, one can thus expect significantly more accurate confidence bounds, leading in turn to reduction in the number of samples used.

7.2 An example proxy: Taylor expansions

In general, the choice of proxy $\wp$ will be problem-dependent, and the availability of a good proxy at all is a *strong* assumption, although not as strong as our previous requirement in Bardenet et al. (2014) that the range (25) can be computed cheaply, which basically corresponds to $\wp_i(\theta,\theta')$ being identically zero for all i. Indeed, we show in this section that all models that possess up to third derivatives can typically be tackled using Taylor expansions as proxies. In Section 8, we detail the case of logistic regression and gamma linear regression.

7.2.1 Taylor expansions

We expand ℓ_i around some reference value $\theta_\star$ to obtain an estimate

$$\hat{\ell}_i(\theta) = \ell_i(\theta_\star) + g_{i,\star}^T(\theta - \theta_\star) + \frac{1}{2}(\theta-\theta_\star)^T H_{i,\star}(\theta-\theta_\star),$$

where $g_{i,\star}$ and $H_{i,\star}$ are respectively the gradient and the Hessian of ℓ_i at $\theta_\star$. The choice of $\theta_\star$ is deferred to Section 7.2.2. Let us now define $\wp_i(\theta,\theta') = \hat{\ell}_i(\theta') - \hat{\ell}_i(\theta)$. The average $\frac{1}{n}\sum_{i=1}^n \wp_i(\theta,\theta')$ can be computed in $\mathcal{O}(1)$ time if one has precomputed

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^n g_{i,\star}$$

and

$$\hat{S} = \frac{1}{n}\sum_{i=1}^{n} H_{i,\star}.$$

Indeed, the following holds

$$\frac{1}{n}\sum_{i=1}^{n} \wp_i(\theta,\theta') = \hat{\mu}^T(\theta'-\theta) + \frac{1}{2}(\theta'-\theta)^T\hat{S}(\theta+\theta'-2\theta_\star).$$

Finally, assuming

$$\left.\frac{\partial \ell_i}{\partial\theta^{(j)}\partial\theta^{(k)}\partial\theta^{(l)}}\right|_{\theta_\star}$$

can be bounded uniformly in i,j,k,l, the absolute difference $\ell_i(\theta') - \ell_i(\theta) - \wp_i(\theta,\theta')$ can be bounded using the Taylor-Lagrange inequality. To conclude, all conditions of Section 7.1 are satisfied by the proxy $\wp_i$.

7.2.2 Drop proxies along the way

When the mass of the posterior is concentrated around the maximum likelihood estimator θ_{MLE}, a single proxy – say a Taylor proxy centered at $\theta_\star = \theta_{\text{MLE}}$ – can represent the target quite accurately. This is the proxy we used in the running examples of Section 6, see Figure 8. When the posterior does not concentrate, or the proposal is not local enough, such a proxy will be inaccurate, potentially resulting in insufficient subsampling gains. There are various tricks that can be applied. One can either precompute proxies across Θ if one has an idea where the modes of π are, and then use the closest proxy to the current state of the chain at each iteration. Alternately, if one agrees to look at the whole dataset every α iterations, we can *drop proxies along the way*, i.e. set $\theta_\star$ to the current state of the chain every α MH iterations. The whole dataset needs to be browsed at each change of the reference point $\theta_\star$, since there is typically some preprocessing to do in order to compute later bounds. In the case of 2nd order Taylor expansions, for example, one has to compute the full gradient, Hessian, and any other quantity needed to bound the third derivatives. What the user should aim at is to sacrifice a proportion α of the budget of the ideal MH to make the remaining iterations cheaper. The proof of Proposition 3 easily generalizes to the case of proxies dropped every constant number of iterations. We demonstrate the empirical performance of such an approach in Sections 8.1.3 and 8.2.2.

7.2.3 A heuristic on the subsampling gain

In (Bardenet et al., 2014), we presented a heuristic that showed the original confidence sampler required $\mathcal{O}(n)$ likelihood evaluations per iteration. At the time, it seemed every attempt at marrying subsampling and MH was fundamentally $\mathcal{O}(n)$. We first repeat here the heuristic from (Bardenet et al., 2014), before arguing that the contributions of this paper can lower this budget to $o(n)$, even $\mathcal{O}(1)$ up to polylogarithmic factors in very favourable conditions.

Assuming a symmetric proposal and a flat prior, the stopping rule of the **while** loop in the original confidence sampler in Figure 9 is met whenever

$$\frac{1}{t}\sum_{i=1}^{t}\log\left[\frac{p(x_i^*|\theta')}{p(x_i^*|\theta)}\right] - \frac{1}{n}\log u$$

is of the same order as the confidence bound $c_t(\delta)$, that is, when $c_t(\delta)$ is of order $1/n$. We consider the Bernstein bound in (27) and we assume the range $C_{t,\theta,\theta'}$ grows with n strictly slower than $\sqrt{n}$. The latter assumption is realistic: $C_{t,\theta,\theta'}$ is often dominated by some power of $\max\|x_i\|_\infty$, and if $x_1,\dots,x_n$ are drawn i.i.d. from a subgaussian distribution, then $\mathbb{E}\max_{i=1}^n x_i = \mathcal{O}(\sqrt{\log n})$ (Cesa-Bianchi and Lugosi, 2006, Lemma A.13). The leading term of the Bernstein bound is proportional to $\hat{\sigma}_{t,\theta,\theta'}/\sqrt{t}$. In simple models such as logistic regression, $\hat{\sigma}_{t,\theta,\theta'}$ is proportional to $\|\theta-\theta'\|$.

Assuming n is large enough that standard asymptotics apply and the target is approximately Gaussian, the results of Roberts and Rosenthal (2001) lead to choose the covariance matrix of the proposal such that $\|\theta-\theta'\|$ is of order $n^{-1/2}$. Summing up, we exit the **while** loop when

$$\frac{1}{n} \sim \frac{1}{\sqrt{t}\sqrt{n}},$$

which leads to $t \sim n$.

Now consider the confidence sampler with second-order Taylor proxies introduced in Section 7.2.1. $\hat{\sigma}_{t,\theta,\theta'}$ and $C_{t,\theta,\theta'}$ now correspond to the standard deviation and range of

$$\left\{\log\left[\frac{p(x_i^*|\theta')}{p(x_i^*|\theta)}\right] - \wp_i^*(\theta,\theta')\right]; 1\le i\le t\right\}.$$

Now let us assume the third-order derivatives at the reference point $\theta_\star$ can be bounded, say by some constant times $\max_i\|X_i\|_\infty^3$ as will be the case for the exponential family models of Section 8. Then $\hat{\sigma}_{t,\theta,\theta'}$ and $C_{t,\theta,\theta'}$ are dominated by

$$\max_i \|X_i\|_\infty^3\left(\|\theta-\theta_\star\|^3 + \|\theta'-\theta_\star\|^3\right). \tag{32}$$

But $\|\theta-\theta_\star\|$ is of order $n^{-1/2}$ if standard asymptotics (van der Vaart, 2000) yield good approximations and $\theta_\star$ is set to the maximum of the posterior. Alternatively, if one has implemented the strategy of dropping proxies regularly, then $\|\theta-\theta_\star\|$ should be of order $n^{-1/2}$ since we assume the covariance matrix of the proposal distribution is of order $1/n$. Again assuming that $\max_i\|X_i\|_\infty^3$ grows, say, like $\rho(n) = o(n^{1/3})$, we now exit the while loop when

$$\frac{1}{n} \sim \frac{\rho(n)^3}{\sqrt{t}n^{3/2}} = \frac{o(1)}{\sqrt{t}\sqrt{n}}.$$

Thus, when the target is approximately Gaussian and the chain is in the mode, the cost in likelihood evaluations per iteration of the confidence sampler with proxy is likely to be $o(n)$. The actual order of convergence depends on the rate of growth of the bounds on the third derivatives. For example, in the case of independent Gaussian data and still assuming (32), we have $t = \mathcal{O}(1)$ up to polylogarithmic factors.

8. Experiments

As a proof of concept, all experiments in this section avoid loading the dataset or proxy-related quantities into memory by building, maintaining and querying from a disk-based database using *SQLite*[2].

8.1 Logistic regression

8.1.1 A Taylor proxy for logistic regression

In logistic regression, the likelihood is defined by $\ell_i(\theta) = \phi(t_i x_i^T \theta)$, where

$$\phi(z) = -\log(1 + e^{-z})$$

and the label t_i is in $\{-1, +1\}$. We can use the Taylor expansion proxy of Section 7.2.1, using

$$g_{i,\star} = \phi'(t_i x_i^T \theta_\star) t_i x_i$$

and

$$H_{i,\star} = \phi''(t_i x_i^T \theta_\star) x_i x_i^T$$

Furthermore,

$$\frac{\partial}{\partial\theta^{(j)}\partial\theta^{(k)}\partial\theta^{(l)}}\ell_i(\theta) = t_i\phi'''(t_i x_i^T\theta)x_i^{(j)}x_i^{(k)}x_i^{(l)}$$

and

$$\left|\phi'''(z)\right| = \frac{1}{4}\left|\frac{\tanh(z/2)}{\cosh^2(z/2)}\right| \leq \frac{1}{4},$$

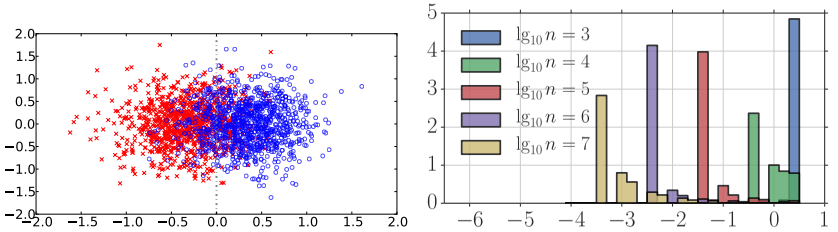
so that

$$\begin{aligned}&|\ell_i(\theta') - \ell_i(\theta) - \wp_i(\theta, \theta')| \\ &\qquad \leq \frac{1}{24}\max_{i=1}^{n}\|x_i\|^3\left\{\|\theta - \theta_\star\|^3 + \|\theta' - \theta_\star\|^3\right\}.\end{aligned}$$

8.1.2 A toy example that requires $\mathcal{O}(1)$ likelihood evaluations

In this section, we consider the simple two-dimensional logistic regression dataset in (Bardenet et al., 2014, Section 4.2.2), where the features within each class are drawn from a Gaussian. The dataset is depicted in Figure 10(a). We consider subsets of the dataset with increasing size $\log_{10} n \in \{3, 4, 5, 6, 7\}$, run a confidence MH chain for each n, started at the MAP, with $\delta = 0.1$ and a single proxy around the MAP. We report the numbers of likelihood evaluations L at each iteration in Figure 10(b). The fraction of likelihood evaluations compared to MH roughly decreases by a factor 10 when the size of the dataset is multiplied by 10: the number of likelihood evaluations is constant for n large enough. In other words, 1 000 random data points at each iteration are enough to get within $\mathcal{O}(\delta)$ of the actual posterior, the rest of the dataset appears to be superfluous. There is a *saturation* phenomenon. By relaxing the goal of sampling from π into sampling from a controlled approximation, we can break the $\mathcal{O}(n)$ barrier and in this particular example reach a cost per iteration of $\mathcal{O}(1)$.

2. `http://www.sqlite.org/`



(a) Synthetic logistic regression dataset

(b) log Fraction $\log_{10}(L/n)$ of number of likelihood evals

Figure 10: Results of 10 000 iterations of confidence MH with a single Taylor proxy, applied to a synthetic logistic regression dataset vs. n

8.1.3 The *covtype* dataset

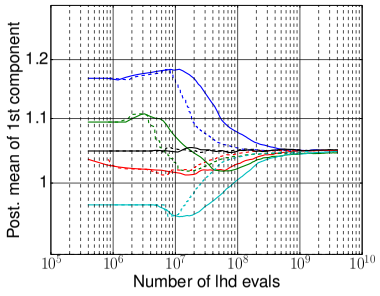
We consider the dataset *covtype.binary*[3] described in Collobert et al. (2002). The dataset consists of 581,012 points, of which we pick $n = 400,000$ as a training set, following the maximum training size in Collobert et al. (2002). The original dimension of the problem is 54, with the first 10 attributes being quantitative. To illustrate our point without requiring a more complex sampler than MH, we only consider the 10 quantitative attributes. We use the preprocessing and Cauchy prior recommended by Gelman et al. (2008).

We run 5 independent chains for 10 000 iterations, dropping proxies every 10 iterations as explained in Section 7.2.2. We obtain a Gelman-Rubin statistic of 1.01 (Robert and Casella, 2004, Section 12.3.4), which suggests the between-chain variance is low enough that we can stop sampling.

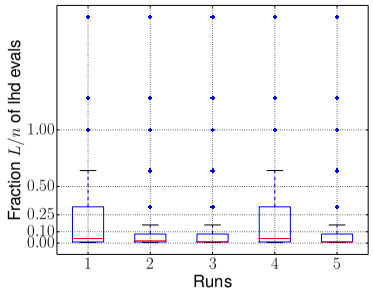
We estimate the number of likelihood evaluations L_k at MH iteration k as follows. First, note that –dropping proxies or not– on a regular iteration where the proxy is not necessarily recomputed, L_k can take values up to $2n$, unlike MH, which can store the evaluation of the likelihood at the current state of the chain from one iteration to the next, and thus only requires n likelihood evaluations per iteration. Second, at an iteration where the proxy is recomputed, the whole data has to be read anyway, so that we choose here to perform a normal MH iteration. This requires the maximum $2n$ likelihood evaluations, Assuming the cost of the likelihood evaluation is the bottleneck, we neglect here the additional cost of computing the proxy itself, and only report $L_k = 2n$ when the proxy is recomputed. Third, whenever we compute the full likelihood at a state of the chain, we store it until the chain leaves that state, similarly to any implementation of MH. Thus, at an iteration that follows a full read of the data, i.e. $L_{k-1} = 2n$, we only count the likelihood evaluations of the proposed state.

We summarize the results in Figure 11. All runs use on average 27 to 42% of n likelihood evaluations per iteration. Since we compute the proxy every $\alpha = 10$ iterations, there is a necessary $2 \times 10 = 20\%$ of n that is due to recomputing the proxy. We manually assessed

3. available at `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`



(a) Posterior mean vs. iteration number



(b) Fraction of likelihood evaluations

Figure 11: Results of 5 runs of a confidence sampler with Taylor proxies dropped every 10 iterations, applied to logistic regression on *covtype*. In Figure 11(a), a solid line corresponds to the online posterior mean of the 1st component of the chain vs. the budget of MH, while a dashed line of the same color corresponds to the budget of the confidence sampler.

the value of α, and recomputing the proxy less often increases the average number of likelihood evaluations (not shown). Thanks to these forced 20%, the rest of the iterations are considerably cheaper than n, since 50% of the iterations require less than 5% of the dataset, as shown in Figure 11(b). Relatedly, although subsampling implies a forced $2n$ likelihood evaluations to start and thus shows an initial delay in Figure 11(a), it quickly catches up and converges faster. The gains are two- or threefold, which is of limited overall practical interest, but we know from Section 7.2.3 and Figure 10(b) that increasing n will also improve the gain.

8.2 Gamma linear regression

8.2.1 A Taylor proxy for gamma regression

In gamma regression, the nonnegative response y is assumed to be gamma-distributed

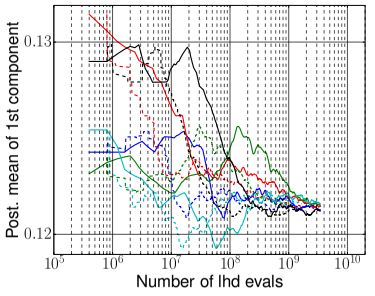
$$y \sim \Gamma\left(\kappa, \frac{e^{x^T\theta}}{\kappa}\right)$$

where $\Gamma(\kappa, s)$ is the gamma distribution with shape parameter κ and scale parameters s. Assuming κ is known, the log likelihood is thus given by

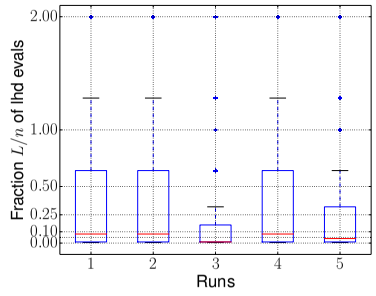
$$\log p(y|x,\theta) \propto -\kappa y e^{-\theta^T x} - \kappa\theta^T x$$

up to an additive constant, so that

$$\nabla \log p(y|x,\theta) = \kappa\left(y e^{-x^T\theta} - 1\right) x,$$



(a) Posterior mean vs. iteration number



(b) Fraction of likelihood evaluations

Figure 12: Results of 5 runs of a confidence sampler with Taylor proxies dropped every 10 iterations, applied to gamma linear regression on *covtype*. In Figure 12(a), a solid line corresponds to the online posterior mean of the 1st component of the chain vs. the budget of MH, while a dashed line of the same color corresponds to the budget of the confidence sampler.

$$\mathrm{Hess}(\log p(y|x,\theta)) = -\kappa y e^{-x^T\theta} x x^T$$

and

$$\frac{\partial}{\partial\theta^{(j)}\partial\theta^{(k)}\partial\theta^{(l)}} \log p(y|x,\theta) = \kappa y e^{-x^T\theta} x^{(j)} x^{(k)} x^{(l)}.$$

Furthermore, we can bound

$$\left|\frac{\partial}{\partial\theta^{(j)}\partial\theta^{(k)}\partial\theta^{(l)}} \log p(y|x,\theta)\right| \leq \kappa \max_{i=1}^{n} |y| \exp\left(-\min_{i=1}^{n} x_i^T\theta\right) \max_{1\leq i\leq n} \|x_i\|_\infty^3.$$

The Taylor proxies of Section 7.2.1 can thus be applied.

8.2.2 The *covtype* dataset

As an application, we consider the *covtype* dataset again and regress the nonnegative feature "horizontal distance to nearest wildfire ignition" onto the other quantitative features. We run 5 independent chains for 10 000 iterations, dropping proxies every 10 iterations as explained in Section 7.2.2. We obtain a Gelman-Rubin statistic of 1.001, which again suggests we can stop sampling. We estimate the evaluation budget as in Section 8.1.3. We summarize the results in Figure 12.

All runs use on average 33 to 54% of n likelihood evaluations per iteration, from which $2\times 10 = 20\%$ are due to recomputing the proxy every 10 iterations. Recomputing the proxy less often increases the average number of likelihood evaluations (not shown). Thanks to these forced 20% the rest of the iterations are considerably cheaper than n, since, as in Section 8.1.3, 50% of the iterations require less than 10% of the dataset. Relatedly, and

similarly to the logistic regression task in Section 8.1.3 subsampling converges two or three times faster in this example. Again, this is a proof of concept that subsampling works, and we know from Section 7.2.3 and Figure 10(b) that increasing n will also improve the gain.

9. Discussion

We have reviewed recent advances in applying MCMC to tall datasets. Divide-and-conquer approaches have yet to solve the recombination problem, i.e. how to obtain a *meaningful* distribution in a *stable* manner from the output of individual chains on a growing number of smaller datasets. Subsampling approaches face different issues, namely that of approaching the right target at a known speed, and of keeping the overall budget in terms of likelihood evaluations per iteration low.

We have proposed an alternative original subsampling approach and have shown that, under strong ergodicity assumptions on the original MH sampler, our algorithm samples from a controlled approximation of the posterior distribution of interest. While these strong assumptions are rarely satisfied in practice, our experiments suggest that our results extend to more general scenarios. Recent bounds by Rudolf and Schweizer (2015) support this, extending the results of Mitrophanov (2005) on perturbations of uniformly ergodic Markov chains to geometrically ergodic chains, and explicitly including our algorithm.

In terms of scaling, our methodology is able to lower the natural cost of $\mathcal{O}(n)$ subsamples per iteration to as low as $\mathcal{O}(1)$ in favourable scenarios, while providing MCMC estimates whose asymptotic variance is of order $\mathcal{O}(1)$. However, we have yet only observed these dramatic gains in contexts where the Bernstein-von Mises approximation is already excellent. So although our algorithm improves on other proposed subsampling approaches, the benefits of such an approach appear limited. Indeed, computing the Bernstein-von Mises approximation for regular models can be typically achieved in only a couple of passes over the data, using for example stochastic gradient to compute the maximum likelihood estimator, and the observed information matrix at this point to estimate the Hessian.

Since the first preprint of the current paper, our Taylor-based control variates have become a popular idea for tall data (Quiroz et al., 2014; Bierkens et al., 2016; Giles et al., 2016). Similarly to our methodology, the algorithms in these references allow to lower the cost per iteration to $\mathcal{O}(1)$ in favourable scenarios and provide MCMC estimates whose asymptotic variance is of order $\mathcal{O}(1)$. However, they similarly only appear to work well when the Bernstein-von Mises approximation is excellent, their performance degrading quickly otherwise (Galbraith, 2016). A similar problem arises with other techniques which implicitly use some form of control variates, such as (MacLaurin and Adams, 2014).

Hence, further work should focus on demonstrating the applicability of subsampling approaches to cases where it is either difficult to compute Bernstein-von Mises even if it is a good approximation (Chernozhukov and Hong, 2003), or – more importantly – cases where n is not large enough that Bernstein-von Mises yields a good approximation and cases where the posterior does not concentrate.

Acknowledgments

The authors thank Louis Aslett, Nando de Freitas, Pierre Jacob, François Septier, Matti Vihola, and Sebastian Vollmer for their comments and discussions on this paper and topic. RB was supported by EPSRC grant EP/I017909/1 through a 2020 science fellowship, and by ANR grants ANR-13-BS03-0006 and ANR-16-CE23-0003. AD was partially supported by EPSRC grants EP/K009850/1 and EP/K000276/1. CH wishes to acknowledge support from the EPSRC, i-like programme grant EP/K014463/1, and the Medical Research Council, Programme Leaders award MC_UP_A390_1107.

Appendix A: Proof of Proposition 1

Define

$$S_n = e^{na(\theta)} \left[1 + \sum_{k=1}^{n} \frac{1}{k!} \prod_{j=1}^{k} D_j^* \right]. \tag{33}$$

By construction and the monotone convergence theorem, $\mathbb{E}S_n \to \mathbb{E}S = e^{n\ell(\theta)}$, where

$$S = e^{na(\theta)} \left[1 + \sum_{k=1}^{\infty} \frac{1}{k!} \prod_{j=1}^{k} D_j^* \right].$$

From (Rhee and Glynn, 2015, Theorem 1), the second moment of Y is

$$\mathbb{E}Y^2 = \sum_{k=0}^{\infty} \frac{1}{\mathbb{P}(N \geq k)} \left[\mathbb{E}(S - S_{k-1})^2 - \mathbb{E}(S - S_k)^2 \right],$$

with the convention $S_{-1} = 0$ and $S_0 = e^{na(\theta)}$. We note that

$$(S - S_k)^2 = e^{2na(\theta)} \sum_{p=k+1}^{\infty} \sum_{q=k+1}^{\infty} \frac{1}{p!q!} \prod_{u=1}^{p} D_u^* \prod_{v=1}^{q} D_v^*.$$

Hence

$$\begin{aligned}
e^{-2na(\theta)} &\left[\mathbb{E}(S - S_{k-1})^2 - \mathbb{E}(S - S_k)^2 \right] \\
&= \frac{1}{k!k!} \mathbb{E} \prod_{u=1}^{k} D_u^* \prod_{v=1}^{k} D_v^* + 2 \sum_{j=k+1}^{\infty} \frac{1}{k!j!} \mathbb{E} \prod_{u=1}^{j} D_u^* \prod_{v=1}^{k} D_v^* \\
&= \frac{1}{k!k!} \left[n^2 \sigma_t(\theta)^2 + n^2 (\ell(\theta) - a(\theta))^2 \right]^k \\
&\quad + 2 \sum_{j=k+1}^{\infty} \frac{1}{k!j!} \left[n(\ell(\theta) - a(\theta)) \right]^{j-k} \left[n^2 \sigma_t(\theta)^2 + n^2 (\ell(\theta) - a(\theta))^2 \right]^k \\
&\geq \frac{\left[n^2 \sigma_t(\theta)^2 + n^2 (\ell(\theta) - a(\theta))^2 \right]^k}{k!k!}.
\end{aligned}$$

Now, since $k!k! \leq 4^{-k}(2k+1)!$, letting

$$A_n \triangleq (1+\epsilon)[n^2\sigma_t(\theta)^2 + n^2(\ell(\theta)-a(\theta))^2],$$

and by definition of N, it comes

$$\begin{aligned}
\frac{\mathrm{Var}Y}{e^{2n\ell(\theta)}} &\geq e^{-2n(\ell(\theta)-a(\theta))}\sum_{k=0}^{\infty}\frac{[2\sqrt{A_n}]^{2k}}{(2k+1)!} - 1 \\
&= \frac{e^{-2n(\ell(\theta)-a(\theta))}}{2\sqrt{A_n}}\sinh(2\sqrt{A_n}) - 1 \\
&= \frac{e^{-2n(\ell(\theta)-a(\theta))}}{4\sqrt{A_n}}\left[e^{2\sqrt{A_n}} - e^{-2\sqrt{A_n}}\right] - 1 \\
&= \frac{e^{-2n(\ell(\theta)-a(\theta))+2n\sqrt{(1+\epsilon)[\sigma_t(\theta)^2+(\ell(\theta)-a(\theta))^2]}}}{n\sqrt{(1+\epsilon)[\sigma_t(\theta)^2+(\ell(\theta)-a(\theta))^2]}} + \mathcal{O}(1).
\end{aligned}$$

Appendix B: Proof of Proposition 2

We write

$$\begin{aligned}
\mathrm{Var}_z\left[\sum_{i=1}^{n}\log p(x_i|\theta,z_i)\right] &= \sum_{i=1}^{n}\left[\mathbb{E}\log^2 p(x_i|\theta,z_i) - (\mathbb{E}\log p(x_i|\theta,z_i))^2\right] \\
&= \sum_{i=1}^{n}\left[(1-I_\theta)\log^2\left(\frac{e^{\ell_i(\theta)}-e^{b_i(\theta)}}{1-I_\theta}\right) + I_\theta\log^2\left(\frac{e^{b_i(\theta)}}{I_\theta}\right)\right] \\
&\quad - \left[(1-I_\theta)\log\left(\frac{e^{\ell_i(\theta)}-e^{b_i(\theta)}}{1-I_\theta}\right) + I_\theta\log\left(\frac{e^{b_i(\theta)}}{I_\theta}\right)\right]^2 \\
&= I_\theta(1-I_\theta)\sum_{i=1}^{n}\left[\log\left(\frac{e^{\ell_i(\theta)}-e^{b_i(\theta)}}{1-I_\theta}\right) - \log\left(\frac{e^{b_i(\theta)}}{I_\theta}\right)\right]^2 \\
&= I_\theta(1-I_\theta)\sum_{i=1}^{n}\log^2\left[\frac{I_\theta}{1-I_\theta}\left(e^{\ell_i(\theta)-b_i(\theta)}-1\right)\right].
\end{aligned}$$

References

T. M. Alkhamis, M. A. Ahmed, and V. K. Tuan. Simulated annealing for discrete optimization with estimation. *European Journal of Operational Research*, 116:530–544, 1999.

P. Alquier, N. Friel, R. Everitt, and A. Boland. Noisy Monte Carlo: Convergence of Markov chains with approximate transition kernels. *Statistics and Computing*, 26(1-2):29–47, 2016.

C. Andrieu and G. O. Roberts. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.

C. Andrieu and M. Vihola. Convergence properties of pseudo-marginal Markov chain Monte Carlo algorithms. *The Annals of Applied Probability*, 25(2):1030–1077, 2015.

C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

J.-Y. Audibert, R. Munos, and Cs. Szepesvári. Exploration–exploitation tradeoff using variance estimates in multi-armed bandits. *Theoretical Computer Science*, 410(19):1876–1902, 2009.

M. Banterle, C. Grazan, A. Lee, and C. P. Robert. Accelerating Metropolis-Hastings algorithms by delayed acceptance. *arXiv:1503.00996*, 2015.

R. Bardenet and O.-A. Maillard. Concentration inequalities for sampling without replacement. *Bernoulli*, 2015.

R. Bardenet, A. Doucet, and C. Holmes. Towards scaling up MCMC: an adaptive subsampling approach. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2014. http://jmlr.org/proceedings/papers/v32/bardenet14-supp.pdf.

M. A. Beaumont. Estimation of population growth or decline in genetically monitored populations. *Genetics*, 164:1139–1160, 2003.

M. J. Betancourt. The fundamental incompatibility of Hamiltonian Monte Carlo and data subsampling. In *Proceedings of The International Conference on Machine Learning (ICML)*, 2015.

G. Bhanot and A. D. Kennedy. Bosonic lattice gauge theory with noise. *Physics Letters B*, 157(1):70–76, 1985.

J. Bierkens, P. Fearnhead, and G.O. Roberts. The zig-zag process and super-efficient sampling for Bayesian analysis of big data. *arXiv:1607.03188*, 2016.

A. Bouchard-Côté, S. J. Vollmer, and A. Doucet. The bouncy particle sampler: A non-reversible rejection-free Markov chain Monte Carlo method. *arXiv:1510.02451*, 2015.

S. R. Bowling, M. T. Khasawneh, S. Kaewkuekool, and B. R. Cho. A logistic approximation to the cumulative normal distribution. *Journal of Industrial Engineering and Management*, 2(1):114–127, 2009.

J. Branke, S. Meisel, and C. Schmidt. Simulated annealing in the presence of noise. *Journal of Heuristics*, 14(6):627–654, 2008.

A. A. Bulgak and J. L. Sanders. Integrating a modified simulated annealing algorithm with the simulation of a manufacturing system to optimize buffer sizes in automatic assembly systems. In *Proceedings of the 20th Winter Simulation Conference*, 1988.

D. M. Ceperley and M. Dewing. The penalty method for random walks with uncertain energies. *Journal of Chemical Physics*, 110, 1999.

N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

T. Chen, E. B. Fox, and C. Guestrin. Stochastic gradient Hamiltonian Monte Carlo. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2014.

V. Chernozhukov and H. Hong. An MCMC approach to classical estimation. *Journal of Econometrics*, 115(2):293–346, 2003.

R. Collobert, S. Bengio, and Y. Bengio. A parallel mixture of SVMs for very large scale problems. *Neural Computation*, 14(5):1105–1114, 2002.

M. Cuturi and A. Doucet. Fast computation of Wasserstein barycenters. In *Proceedings of The International Conference on Machine Learning (ICML)*, 2014.

R. Douc, É. Moulines, and D. Stoffer. *Nonlinear time series*. Chapman-Hall, 2014.

A. Doucet, M. K. Pitt, G. Deligiannidis, and R. Kohn. Efficient implementation of Markov chain Monte Carlo when using an unbiased likelihood estimator. *Biometrika*, 102(2): 295–313, 2015.

S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth. Hybrid Monte Carlo. *Physics Letters B*, 195(2):216–222, 1987.

N. Galbraith. On Event-Chain Monte Carlo Methods. Master's thesis, Department of Statistics, Oxford University, 2016.

A. Gelman, A. Jakulin, M. G. Pittau, and Y.-S. Su. A weakly informative default prior distribution for logistic and other regression models. *The Annals of Applied Statistics*, pages 1360–1383, 2008.

A. Gelman, A. Vehtari, P. Jylänki, C. Robert, N. Chopin, and J. P. Cunningham. Expectation propagation as a way of life. *arXiv:1412.4869*, 2014.

M. Giles, T. Nagapetyan, L. Szpruch, S. Vollmer, and K. Zygalakis. Multilevel Monte Carlo for scalable Bayesian computations. *arXiv:1609.06144*, 2016.

P. W. Glynn and C.-H. Rhee. Exact estimation for Markov chain equilibrium expectations. *Journal of Applied Probability*, 51:377–389, 2014.

Z. Huang and A. Gelman. Sampling for Bayesian computation with large datasets. Technical report, Department of Statistics, Columbia University, 2005.

P. E. Jacob and A. H. Thiery. On non-negative unbiased estimators. *The Annals of Statistics*, 43(2):769–784, 2015.

A. Korattikara, Y. Chen, and M. Welling. Austerity in MCMC land: Cutting the Metropolis-Hastings budget. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2014.

L Lin, KF Liu, and J Sloan. A noisy Monte Carlo algorithm. *Physical Review D*, 61(7): 074505, 2000.

D. MacLaurin and R. P. Adams. Firefly Monte Carlo: Exact MCMC with subsets of data. In *Proceedings of the conference on Uncertainty in Artificial Intelligence (UAI)*, 2014.

C. H. Mak. Stochastic potential switching algorithm for Monte Carlo simulations of complex systems. *Journal of Chemical Physics*, 122(21), 2005.

J.-M. Marin, P. Pudlo, C. P. Robert, and R. Ryder. Approximate Bayesian computation methods. *Statistics and Computing*, 22(6):1167–1180, 2012.

S. Minsker, S. Srivastava, L. Lin, and D. B. Dunson. Scalable and robust Bayesian inference via the median posterior. In *Proceedings of The International Conference on Machine Learning (ICML)*, 2014.

A. Yu. Mitrophanov. Sensitivity and convergence of uniformly ergodic Markov chains. *Journal of Applied Probability*, 42(4):1003–1014, 2005.

W. Neiswanger, C. Wang, and E. Xing. Asymptotically exact, embarassingly parallel MCMC. In *Proceedings of the conference on Uncertainty in Artificial Intelligence (UAI)*, 2014.

G. K. Nicholls, C. Fox, and A. Muir-Watt. Coupled MCMC with a randomized acceptance probability. *arXiv.org:1205.6857*, 2012.

A. Pakman, D. Gilboa, D. Carlson, and L. Paninski. Stochastic bouncy particle sampler. *arXiv:1609.00770*, 2016.

E. A. J. F. Peters and G. de With. Rejection-free Monte carlo sampling for general potentials. *Physical Review E*, 85(2):026703, 2012.

N. S. Pillai and A. Smith. Ergodicity of approximate MCMC chains with applications to large data sets. *arXiv:1405.0182*, 2014.

M. Quiroz, M. Villani, and R. Kohn. Speeding up MCMC by efficient data subsampling. *arXiv:1404.4178*, 2014.

C.-H. Rhee and P. W. Glynn. Unbiased estimation with square root convergence for sde models. *Operations Research*, 63(5):1026–1043, 2015.

C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer-Verlag, New York, 2004.

G. O. Roberts and J. S. Rosenthal. Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science*, 16:351–367, 2001.

D. Rudolf and N. Schweizer. Perturbation theory for Markov chains via Wasserstein distance. *arXiv preprint arXiv:1503.04123*, 2015.

S. L. Scott, A. W. Blocker, and F. V. Bonassi. Bayes and big data: The consensus Monte Carlo algorithm. In *Proceedings of the Bayes 250 conference*, 2013.

S. Singh, M. Wick, and A. McCallum. Monte Carlo MCMC: Efficient inference by approximate sampling. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2012.

S. Srivastava, V. Cevher, Q. Tran-Dinh, and D. B. Dunson. WASP: scalable Bayes via barycenters of subset posteriors. In *Proceedings of the international conference on artificial intelligence and statistics (AISTATS)*, 2015.

H. Strathmann, D. Sejdinovic, and M. Girolami. Unbiased Bayes for big data: Paths of partial posteriors. *arXiv:1501.03326*, 2015.

Y. W. Teh, A. H. Thiery, and S. J. Vollmer. Consistency and fluctuations for stochastic gradient Langevin dynamics. *Journal of Machine Learning Research*, 17(7):1–33, 2016.

R. Trotta. Bayes in the sky: Bayesian inference and model selection in cosmology. *Contemporary Physics*, 49(2), 2006.

A. W. van der Vaart. *Asymptotic Statistics*. Cambridge University Press, 2000.

A. W. van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes*. Springer, 1996.

L. Wang and L. Zhang. Stochastic optimization using simulated annealing with hypothesis test. *Applied Mathematics and Computation*, 174:1329–1342, 2006.

X. Wang and D. B. Dunson. Parallelizing MCMC via Weierstrass sampler. *arXiv preprint arXiv:1312.4605*, 2013.

M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2011.

J. Wright. Genetics: unravelling complexity. *Nature*, 508(7494):S6–S7, 2014.

M. Xu, B. Lakshminarayanan, Y. W. Teh, J. Zhu, and B. Zhang. Distributed Bayesian posterior sampling via moment sharing. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

Journal of Machine Learning Research 18 (2017) 1-33 Submitted 5/15; Revised 2/17; Published 5/17

Explaining the Success of AdaBoost and Random Forests as Interpolating Classifiers

Abraham J. Wyner AJW@WHARTON.UPENN.EDU
Matthew Olson MAOLSON@WHARTON.UPENN.EDU
Justin Bleich JBLEICH@WHARTON.UPENN.EDU
Department of Statistics
Wharton School, University of Pennsylvania
Philadelphia, PA 19104, USA

David Mease DMEASE@APPLE.COM
Apple Inc.

Editor: Koby Crammer

Abstract

There is a large literature explaining why AdaBoost is a successful classifier. The literature on AdaBoost focuses on classifier margins and boosting's interpretation as the optimization of an exponential likelihood function. These existing explanations, however, have been pointed out to be incomplete. A random forest is another popular ensemble method for which there is substantially less explanation in the literature. We introduce a novel perspective on AdaBoost and random forests that proposes that the two algorithms work for similar reasons. While both classifiers achieve similar predictive accuracy, random forests cannot be conceived as a direct optimization procedure. Rather, random forests is a self-averaging, interpolating algorithm which creates what we denote as a "spiked-smooth" classifier, and we view AdaBoost in the same light. We conjecture that both AdaBoost and random forests succeed because of this mechanism. We provide a number of examples to support this explanation. In the process, we question the conventional wisdom that suggests that boosting algorithms for classification require regularization or early stopping and should be limited to low complexity classes of learners, such as decision stumps. We conclude that boosting should be used like random forests: with large decision trees, without regularization or early stopping.

Keywords: AdaBoost, random forests, tree-ensembles, overfitting, classification

1. Introduction

In the "boosting" approach to machine learning, a powerful ensemble of classifiers is formed by successively refitting a weak classifier to different weighted realizations of a data set. This intuitive procedure has seen a tremendous amount of success. In fact, shortly, after its introduction, in a 1996 NIPS conference, Leo Brieman crowned AdaBoost (Freund and Schapire, 1996) (the first boosting algorithm) the "best off-the-shelf classifier in the world (Friedman et al., 2000)." AdaBoost's early success was immediately followed by efforts to explain and recast it in more conventional statistical terms. The statistical view of boosting holds that AdaBoost is a stage-wise optimization of an exponential loss function (Friedman et al., 2000). This realization was especially fruitful leading to new "boosting machines" (Friedman, 2001; Ridgeway, 2006) that could perform probability estimation and regression as well as adapt to different loss functions. The statistical view, however, is not the only explanation for the success of AdaBoost. The computer science literature has found generalization error guarantees using VC bounds from PAC learning theory and margins (Guestrin, 2006). While some research has cast doubt on the ability of any one of these to fully account for the performance of AdaBoost they are generally understood to be satisfactory (Schapire, 2013).

This paper parts with traditional perspectives on AdaBoost by concentrating our analysis on the implications of the algorithm's ability to perfectly fit the training data in a wide variety of situations. Indeed, common lore in statistical learning suggests that perfectly fitting the training data must inevitably lead to "overfitting." This aversion is built into the DNA of a statistician who has been trained to believe, axiomatically, that data can always be decomposed into signal and noise. Traditionally, the "signal" is always modeled smoothly. The resulting residuals represent the "noise" or the random component in the data. The statistician's art is to walk the balance between the signal and the noise, extracting as much signal as possible without extending the fit to the noise. In this light, it is counterintuitive that any classifier can ever be successful if every training example is "interpolated" by the algorithm and thus fit without error.

The computer scientist, on the other hand, does not automatically decompose problems into signal and noise. In many classical problems, like image detection, there is no noise in the classical sense. Instead there are only complex signals. There are still residuals, but they do not represent irreducible random errors. If the task is to classify images into those with cats and without, the problem is hard not because it is noisy. There are no cats wearing dog disguises. Consequently, the computer scientist has no dogmatic aversion to interpolating training data. This was the breakthrough.

It is now well-known that interpolating classifiers can work, and work well. The AdaBoost classifier created a huge splash by being better than its established competitors (for instance, CART, neural networks, logistic regression) (Breiman, 1998) and substantively better than the technique of creating an ensemble using the bootstrap (Breiman, 1996). The statistics community was especially confounded by two properties of AdaBoost: 1) interpolation (perfect prediction in sample) was achieved after relatively few iterations, 2) generalization error continues to drop even after interpolation is achieved and maintained.

The main point of this paper is to demonstrate that AdaBoost and similar algorithms work not in spite, but because of interpolation. To bolster this claim, we will draw a constant analogy with random forests (Breiman, 2001), another interpolating classifier. The random forests algorithm, which is also an ensemble-of-trees method, is generally regarded to be among the very best commonly used classifiers (Manuel Fernández-Delgado and Amorim, 2014). Unlike AdaBoost, for which there are multiple accepted explanations, random forest's performance is much more mysterious since traditional statistical frameworks do not necessarily apply. The statistical view of boosting, for example, cannot apply to random forests since the algorithm creates decision trees at random and then averages the results—there is no stage-wise optimization. In this paper, we will put forth the argument that both algorithms are effective for the same reason. We consider AdaBoost and random forests as canonical examples of "interpolating classifiers," which we define to be a classifier's al-

©2017 Abraham J. Wyner, Matthew Olson, Justin Bleich, and David Mease. JMLR 18(48):1-33, 2017
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-240.html.

gorithmic property of fitting the training data completely without error. Each of these interpolating classifiers also exhibits a self-averaging property. We attempt to show that these two properties together make for a classifier with low generalization error. While it is easy to see that random forests has both of these mechanisms by design, it is less clear that this is true for AdaBoost.

It is worth noting that Breiman noticed the connection between random forests and AdaBoost as well, although his notion of a random forest was more general, including other types of large ensembles of randomly grown trees (Breiman, 2001). In his 2001 *Random Forests* paper, he conjectured that the weights of AdaBoost might behave like an ergodic dynamic system, converging to an invariant distribution. When run for a long time, the additional rounds of AdaBoost were equivalent to drawing trees randomly grown according to this distribution, much like a random forest. Recent work has followed up on this idea, proving that the weights assigned by AdaBoost do indeed converge to a invariant distribution [1] (Belanich and Ortiz, 2012). In this work, the authors also show that functions of these weights, such as the generalization error and margins, also converge. This work certainly complements ours, but we focus on the similarity between AdaBoost and random forests through the lens of the type of decision surfaces both classifiers produce, and ability of both algorithms to achieve zero error on the training set.

One of our key contributions will be to present a decomposition of AdaBoost as the weighted sum of interpolating classifiers. Another contribution will be to demonstrate the mechanism by which interpolation combined with averaging creates an effective classifier. It turns out that interpolation provides a kind of robustness to noise: if a classifier fits the data extremely locally, a "noise" point in one region will not affect the fit of the classifier at a nearby location. When coupled with averaging, the result is that the fit stabilizes at regions of the data where there is signal, while the influence of noise points on the fit becomes even more localized. It will be easy to see this point holds true for random forests. For AdaBoost, it is less clear, however, and a decomposition of AdaBoost and simulation results in Section 4 will demonstrate this crucial point. We will observe that the error of AdaBoost at test points near noise points will continue to decrease as AdaBoost is run for more iterations, demonstrating the localizing effect of averaging interpolating classifiers.

We will begin in Section 2 by critiquing some of the existing explanations of AdaBoost. In particular, we will discuss at length some of the shortcomings of the statistical optimization view of AdaBoost. In Section 3, we will discuss the merits of classification procedures that interpolate the training data, that is, that fit the training data set with no error. The main conclusion from this section is that interpolation, done correctly, can provide robustness to a fit in the presence of noise. This discussion will be augmented with simulations discussing the performance of random forests, AdaBoost, and other algorithms in a noisy environment. We will then derive our central observation in Section 4, namely that AdaBoost can be decomposed as a sum of classifiers, each of which fits the training data perfectly. The implication from this observation is that for the best performance, we should run AdaBoost for many iterations with deep trees. The deep trees will allow the component classifiers to interpolate the data, while a large number of iterations will lend to a bagging

1. More specifically, they consider the so called "Optimal AdaBoost" algorithm, which is assume to pick the base classifier with lowest weighted error at each round. They show that the per round average of any measurable function of the training weights converges under mild conditions.

effect. We will then demonstrate this intuition in a real data example in Section 5. Finally, we conclude with a brief discussion in Section 6.

2. Competing Explanations for the Effectiveness of Boosting

In this section we will present an overview of some of the most popular explanations for the success of boosting, with analysis of both the strengths and weaknesses of each approach. Our emphasis will focus on the margins view of boosting and the statistical view of boosting, each of which has a large literature and has led to the development of variants of boosting algorithms. For a more extensive review of the boosting literature, one is well-advised to consult Schapire and Freund (2012).

Before we begin, we will briefly review the AdaBoost algorithm not only to refresh the reader's mind, but also to establish the exact learning algorithm this paper will consider, as there are many variants of AdaBoost. To this end, the reader is invited to review Algorithm 1. In our setting, we are given N training points $(\mathbf{x}_i, y_i)$ where $\mathbf{x}_i \in \mathcal{X}$ and $y_i \in \{-1, +1\}$. On round m, where $m = 1, \ldots, M$, we fit a weak classifier $G_m(\mathbf{x})$ to a version of the data set reweighted by some weighting vector $\mathbf{w}_m$. We then calculate the weighted misclassification rate of our chosen learner, and update the weighting measure used in the next round, $\mathbf{w}_{m+1}$. The final classifier is output as the sign of a weighted linear combination of classifiers produced from each stage of the algorithm. In practice, one sometimes limits the number of rounds of boosting as a form of regularization. We will discuss this point more in the next section, and challenge its usefulness in later parts of the paper.

Algorithm 1: AdaBoost Hastie et al. (2009)

1. Initialize the observation weights $w_i = \dfrac{1}{N},\ i = 1, 2, \ldots, N.$
2. For $m = 1$ to M:
   (a) Fit a classifier $G_m(\mathbf{x})$ to the training data using weights w_i.
   (b) Compute $\text{err}_m = \dfrac{\sum_{i=1}^{N} w_i I\left(y_i \neq G_t\left(\mathbf{x}_i\right)\right)}{\sum_{i=1}^{N} w_i}$.
   (c) Compute $a_m = \log\left(\dfrac{1 - \text{err}_t}{\text{err}_t}\right)$.
   (d) Set $w_i \leftarrow w_i \cdot \exp\left(a_t \cdot I\left(y_i \neq G_t\left(\mathbf{x}_i\right)\right)\right)$
   (e) Set $f_i(\mathbf{x}) = \sum_{m=1}^{M} a_m G_m\left(\mathbf{x}\right)$
3. Output $f(\mathbf{x}) = \text{sign}\left(f_M(\mathbf{x})\right)$

2.1 Margin View of Boosting

Some of the earliest attempts to understand AdaBoost's performance predicted that its generalization error would increase with the number of iterations: as AdaBoost is run for

more rounds, it is able to fit the data increasingly well which should lead to overfitting. However, in practice we observe that running boosting for many rounds does not overfit in most cases. One of the first attempts to resolve this paradox was explored by Schapire et al. (1998), who focused on the *margins* of AdaBoost. The margins can be thought of as a measure of how confident a classifier is about how it labels each point, and one would hypothetically desire to produce a classifier with margins as large as possible. Schapire et al. (1998) proved that AdaBoost's generalization error decreases as the size of the margins increase. Indeed, in practice one observes that as AdaBoost is run for many iterations, test error decreases while the size of the empirical margins increase. In fact, recent research has demonstrated that AdaBoost can be reformulated exactly as mirror descent applied to the problem of maximizing the smallest margin in the training set under suitable separability conditions (Freund et al., 2013).

One could take these observations to suggest that a more effective algorithm might be designed to explicitly optimize margins. However, one can find evidence against this hypothesis in Breiman's *arc-gv* algorithm (Breiman, 1999). Breiman designed the *arc-gv* algorithm to maximize the minimum margin in a data set, and he found that this algorithm actually had worse generalization error than AdaBoost. Moreover, he developed generalization error bounds based on the minimum margin of a classifier which were tighter than those established by Shapire, casting doubt on the existing margin explanation. Other algorithms designed to maximize margins, such as *LP-Boost* have also been found to perform worse than AdaBoost in practice (Wang et al., 2011). Critics of these supposed counterexamples to the margin view of boosting note that AdaBoost's success likely depends on the entire distribution of margins on the data, not just the smallest margin. More recent work has improved upon the Breiman's generalization bound by taking into account other aspects of the margin that more closely reflect its distribution, adding new life to the margin explanation of AdaBoost (Gao and Zhou, 2013). While the margin explanation of AdaBoost is certainly intuitive, its role in producing low generalization error is still an area of active research.

2.2 Statistical Optimization View of Boosting

Friedman et al. (2000) take great strides to clear up the mystery of boosting to provide statisticians with a statistical view of the subject. The heart of their article is the recasting of boosting as a statistically familiar program for finding an additive model by means of a forward stage-wise approximate optimization of an exponential criterion. In short, this view places boosting firmly in classical statistical territory by clearly defining it as a procedure to search through the space of convex combinations of *weak* learners or *base* classifiers. This explanation has been widely assimilated and has reappeared in the statistical literature as well as in a plethora of computer science articles. Subsequent to the seminal publication of Friedman et al. (2000) there has been a flurry of activity dedicated to theoretical analysis of the algorithm. This was made possible by the identification of boosting as optimization, which therefore admits of a mathematically tractable representation. Research on the optimization properties of AdaBoost and the exponential loss function is still an active area of research, see Mukherjee et al. (2013), for example.

Although the statistical optimization perspective of AdaBoost is surely interesting and informative, there remain problems. First, we observe that the fact that AdaBoost minimizes an exponential loss may not alone account for its performance as a classifier. Wyner (2003) introduces a variant of AdaBoost called Beta-Boost which is very similar to AdaBoost except that by design the exponential loss function is constant throughout the iterations. Despite this, Beta-Boost was able to demonstrate similar performance to AdaBoost on simulated data sets. Furthermore, among many similar examples in the literature, Mease and Wyner (2008) present a simulation example in which the the exponential loss is monotonically increasing with the number of iterations of AdaBoost on a test set, while the generalization error decreases. In this example, the value of the exponential loss is uninformative about how well the classifier generalizes. Freund et al. (2013) also provide evidence to this end. They conduct an experiment that compares AdaBoost to two AdaBoost variants that minimize the exponential loss function at differing rates: one performs the minimization very quickly through gradient descent, while the other performs the minimization quite slowly. They find that AdaBoost performed significantly better than these two competitors, suggesting that AdaBoost's strong performance cannot be tied exclusively to its action on the exponential loss function.

We also contend that some of the mathematical theory connected with the statistical optimization view of boosting has a disconnect with the types of boosting algorithms that work in practice. The optimization theory of boosting insists that overfitting can be avoided by requiring the set of weak learners, to be just that: *weak*. Bühlmann and Yu (2003) argues that one can avoid overfitting by employing regularization with weak base learners. However, empirical evidence points to quite the opposite: boosting deep trees for many iterations tends to produce a better classifier than boosted stumps with regularization (Mease and Wyner, 2008). The use of early-stopping as a form of regularization has also been called into question (Mease and Wyner, 2008). The thrust of our paper will be to demonstrate why we *should* actually expect boosting with deep trees run for many iterations to have better generalization error. Recent work also suggests that boosting low complexity classifiers may not be able to achieve good accuracy in difficult classification tasks such as speech recognition or image recognition (Cortes et al., 2014). This paper proposes an algorithm called "DeepBoost" which encourages boosting high complexity base classifiers—such as very deep decision trees—but in a "capacity-conscious" way. One last problem with theory associated with the statistical view of boosting is that by its very nature it suggests that we should be able to extract conditional class probability estimates from the boosted fit, as the procedure is apparently maximizing a likelihood function. Mease and Wyner (2008), however, point out a number of examples where the implied conditional class estimates from the boosting fit diverge to zero and one. While boosting appears to do an excellent job as a classifier, it apparently fails to estimate probability quantiles correctly.

We can now summarize the main empirical contradictions with existing theoretical explanations of boosting, which motivates the view we present in this paper:

1. Boosting works well, perhaps best in terms of performance if not efficiency, with "strong learners" like C4.5 and CART (Niculescu-Mizil and Caruana, 2005).

2. The value of exponential loss does not always bear a clear relationship to generalization error (Mease and Wyner, 2008).

3. The optimization theory offers no explanation as to why the training error can be zero, yet the test error continues to descend (Freund and Shapire, 2000)

This paper will squarely depart from the statistical optimization view by asserting that AdaBoost may be best thought of as a (self) smoothed, interpolating classifier. We will see that unlike the statistical optimization view, this perspective suggests that for best performance once should run many iterations of AdaBoost with deep trees. This will allow us to draw a number of analogies between AdaBoost and random forests. A key component to this argument will consist of explaining the success of interpolating classifiers in noisy environments. We will pursue this line of thought in the following section.

3. Interpolating Classifiers

Algorithm 2: Random Forests Hastie et al. (2009)

1. For $b = 1$ to B:
   (a) Draw a bootstrap sample $\mathbf{X}^*$ of size N from the training data
   (b) Grow a decision tree T_b to the data $\mathbf{X}^*$ by doing the following recursively until the minimum node size n_{min} is reached:
      i. Select m of the p variables
      ii. Pick the best variable/split-point from the m variables and partition
2. Output the ensemble $\{T_b\}_b^B$

Let $\hat{C}_b(\mathbf{x}^*)$ be predicted class of tree T_b. Then $\hat{C}_{rf}^B(\mathbf{x}^*) = \text{majority vote}\{\hat{C}_b(\mathbf{x}^*)\}_1^B$.

It is a widely held belief by statisticians that if a classifier interpolates all the data, that is, it fits all the training data without error, then it cannot be consistent and should have a poor generalization error rate. In this section, we demonstrate that there are interpolating classifiers that defy this intuition: in particular, AdaBoost and random forests will serve as leading examples of such classifiers. We argue that these classifiers achieve good out of sample performance by maintaining a careful balance between the complexity required to perfectly match the training data and a general semi-smoothness property. We begin with a quick review of the random forests classifier, which will be in constant analogy with AdaBoost.

3.1 Random Forests

Random forests has gained tremendous popularity due to robust performance across a wide range of data sets. The algorithm is often capable of achieving best-in-class performance with respect to low generalization error and is not highly sensitive to choice of tuning parameters, making it the off-the-shelf tool of choice for many applications.

Algorithm 2 reviews the procedure for constructing a random forests model. Note that in many popular implementations, such as `R` implementation `randomForest` Liaw and Wiener (2002) built from Breiman's CART software, n_{min} is set to one for classification. This implies that each decision tree is designed to be grown to maximal depth and therefore

necessarily interpolates the data in its bootstrap sample (assuming as least one continuous predictor). This results in each tree being a low bias but high variance estimator. Variance is then reduced by averaging across trees, resulting in a "smoothing" of the estimated response surface. The random predictor selection within each tree further reduces variance by lowering the correlation across the trees. The final random forest classifier still fits the entire training data set perfectly, at least with very high probability. To see this is true, consider any given training point. As the number of trees increases, with probability close to one, that point will be present in the majority of the bootstrap samples used to fit the trees in the forest. Thus the point will get the correct training set label when the votes are tabulated to determine the final class label.

We wish to emphasize that despite its success, random forests is not directly optimizing any loss function across the entire ensemble; each tree is grown independently of the other trees. While each tree may optimize a criteria such as the Gini index, the full ensemble is not constructed in any optimization-driven fashion such as is the case for AdaBoost. While there has been recent theoretical work describing the predictive surface of random forests (Wager and Walther, 2015), the analysis required unnatural assumptions that are hard to justify in practice (such as the growth rate of minimum leaf size). Rather, we postulate that the success of the algorithm is due to its interpolating nature plus the self-averaging mechanism. We next consider the implications of interpolating classifiers more broadly.

3.2 Local Robustness of Interpolating Classifiers

Let us begin with a definition of interpolation:

Definition: Let X_i be vector observations of predictor variables a let Y_i be the observed class label. A classifier $f(X)$ is said to be an *interpolating* classifier if for every training set example, the classifier assigns the correct class label; that is for every i, $f(X_i) = Y_i$.

The term "interpolation" is likely jarring for some readers. In many contexts, one often thinks about interpolating a set of points with classically smooth functions, such as polynomial splines. However, strictly speaking, there are many other ways that one might interpolate a set of points—through the fit of an AdaBoost classifier, for instance! Since the notion of fitting a set of points without error is central to this paper, and since the common definition of interpolation does not preclude the kinds of fits we consider, we felt it appropriate to proceed with the term.

Many statisticians are not comfortable with classifiers that interpolate the training data: common wisdom suggests that any classifier which fits the training data perfectly must have poor generalization error. Indeed, one of the first interpolating classifiers that might come to one's mind, the one-nearest neighbor, can be shown to be inconsistent and have poor generalization error in environments with noise. Specifically, Cover and Hart (1967) have shown that the asymptotic generalization error for the one-nearest neighbor classifier is at least as large as the Bayes error rate. However, the claim that all interpolating classifiers overfit is problematic, especially in light of the demonstrated success of classifiers that perfectly fit the training data, such as random forests.

One of our key insights reverses the common intuition about classifiers: interpolation can prevent overfitting. An interpolated classifier, if sufficiently local, minimizes the influence of noise points in other parts of the data. In order to make this point conceptually clear, it is helpful to put ourselves in familiar territory with a regression example.

Suppose we are trying to predict a continuous response y based on real-valued x observations. Let us assume that the true underlying model is $y = x + \epsilon$, where ϵ is a mixture of a point mass at zero and some heavy-tailed distribution. In other words, we'll assume that most points in a given training set reflect the true linear relationship between y and x, but a few observations will be noise points. This is analogous to the types of probability models we typically consider in classification settings, such as those found in later sections of the paper. Figure 1 shows hypothetical training data: note that the only "noise" point is found at $x = 0.4$. We then consider fitting three models to this data: two interpolating functions, given by the blue and black lines, and an ordinary regression fit given by the red line. The first thing to notice is that the two interpolating fits differ only from the true target mean model $y = x$ only at the noise point $x = 0.4$. In contrast, the fit of the regression line deviates from the underlying target over the entire range of x. The one noise point corrupted the entire fit of the regression line, while the interpolating lines were able to minimize the influence of the noise point by adapting to it only very locally. Moreover, one should note that between the two interpolating fits, the blue line interpolates more locally than the black line, and thus its overall fit is even less influenced by the noise point. This simplified example is of course meant to be didactic, but we will show throughout the rest of this paper that in practice AdaBoost and random forest do indeed produce fits similar to the blue line.

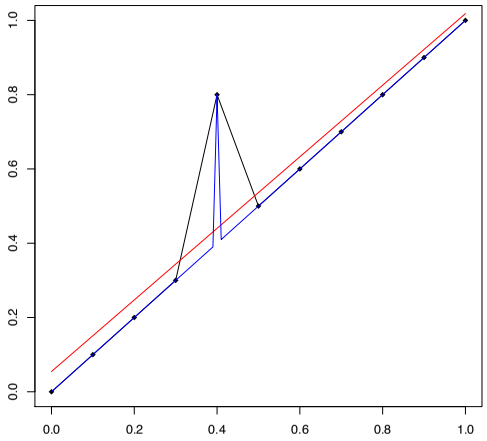


Figure 1: Three estimated regression functions with two interpolating fits (black and blue) and an ordinary least squares fit (red).

While it is conceptually clear that it is desirable to produce fits like the blue interpolating line in the previous example, one may wonder how such fits can be achieved in practice. In the classification setting, we will argue throughout this paper that this type of fit can be realized through the process of averaging interpolating classifiers. We will refer to the decision surface produced by this process as being *spiked-smooth*. The decision surface is *spiked* in the sense that it is allowed to adapt in very small regions to noise points, and it is *smooth* in the sense that it has been produced through averaging. A technical definition of a spiked-smooth decision surface would lead us too far astray. It may be helpful instead to consider the types of classifiers that do not produce a spiked-smooth decision surface, such as a logistic regression. Logistic regression separates the input space into two regions with a hyperplane, making a constant prediction on each region. This surface is not spiked-smooth because it does not allow for local variations: in large regions (namely, half-spaces), the classifier's predictions are constrained to be the same sign. Figure 2 provides a graphical illustration of this intuition.

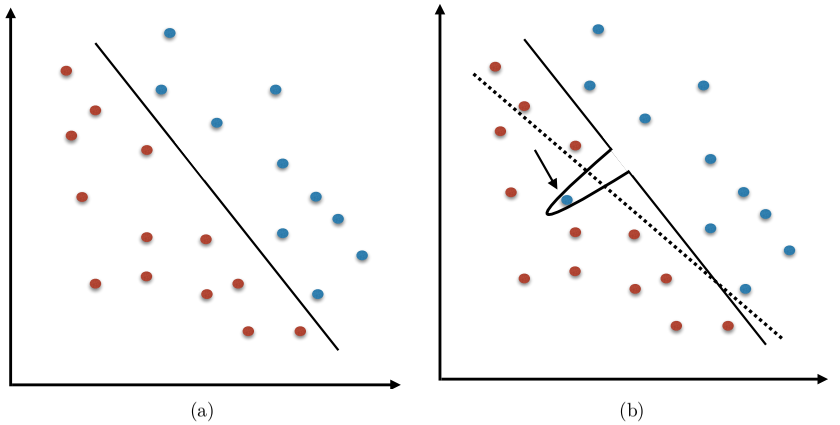


Figure 2: An illustration of the robustness to noise of interpolating classifiers. In 2a a classifier (such as a logistic regression) is fit to a set of training data, and the decision boundary produced by this classifier is shown as a solid line. In 2b, the same type of classifier is fit to the training data, except a noise point is added to the data (the blue point marked by the arrow). The one noise point shifts the entire decision boundary, which is shown as the dotted line. On the other hand, the decision boundary produced by a classifier which interpolates very locally is shown as a solid line. It is clear that this classifier is able to adapt locally to the noise point, and the overall fit does not get corrupted.

Intuitively, we expect ensembles of interpolating classifiers to generalize well because they are flexible enough to fit a complex signal, and "local" enough to prevent the undue influence of noise points. It is clear that random forests are averaged collections of interpolating classifiers, and we will show in section 4 that AdaBoost may be thought of in

the same way. Classically smooth classifiers—such as logistic regression or pruned CART trees—are forced to produce fits that are locally constant. It is harder for such classifiers to "recover" from making a mistake at a noise point since the surface of the fit will affect the fit at nearby points. Interpolating classifiers are flexible enough to make mistakes in "small regions." When averaged over many such classifiers, the influence of the noise point can be easily smoothed out. Later examples in the paper will visualize this process, and will demonstrate that the later iterations of AdaBoost have a smoothing effect which shrinks down the influence of noise points on the overall fit of the classifier.

With this discussion in mind, let us consider another conceptual example, this time in the classification setting. Suppose we have have two predictors x_1 and x_2 distributed independently and uniformly on $[0,1]^2$ and $y \in \{-1,+1\}$. Further suppose that the true conditional class probability function is

$$\mathbb{P}(\mathbf{x}) = \mathbb{P}(y=1|\mathbf{x}) = p = .75$$

for all $\mathbf{x}$. This is a pure noise model with no signal, but in general one could view this as a subspace of a more complex model in which the $\mathbb{P}(\mathbf{x})$ function is approximately constant. Since the Bayes decision rule is to classify every point as a "+1", we would desire an algorithm that will match the Bayes rule as close as possible. Again, we stress that this closeness should be judged with respect to the population or a hold-out sample. On this training data, any interpolating classifier will necessarily differ from the Bayes rule for $1-p = 25\%$ of the points on average.

Figure 3 shows a possible sample of training data from this model of size $n = 20$. The blue points represent the "+1"'s and the red points represent the "-1"'s. There are $5/20 = 25\%$ red points. The training data was sampled according to a Latin Hypercube design using the midpoints of the squares so that the points would be evenly spaced, but that is not essential.

Figure 4 shows four hypothetical classifiers that could result from fitting boosted decision tree models to the training data in Figure 3. When decision trees are used as base learners for data with continuous predictors, it is a common convention to restrict the split points of the trees to be the midpoints of the predictors in the training data. Consequently, the classifier in each small square shown in Figure 3 will necessarily be constant throughout; this is the finest resolution of the classifier resulting from boosting decision tress assuming no sub-sampling. Thus, Figure 4a represents the interpolating classifier closest to the Bayes rule. (In these plots, pink squares represent "-1"'s and light blue squares represent "+1"'s.) Note that the interpolation is in fact quite local; the estimated function varies rapidly in the small neighborhoods of the pink squares. For such a classification rule the percentage of points in a hold-out sample that would differ from the Bayes rule (in expectation over training sets) would be $(1-p)n/n^d$ where $p = \mathbb{P}(y=1|\mathbf{x})$ and d is the dimensionality of the predictor space (for our example, $d=2$). We will present evidence later that in noisy environments boosting sufficiently large trees does actually tend to find such rules as that in Figure 4a. Interestingly, since

$$\lim_{n\to\infty}(1-p)n/n^d = 0$$

such rules are in fact consistent. This illuminates the point that interpolation does not rule out consistency. By allowing the decision boundary to be "mostly" smooth, with spikes of

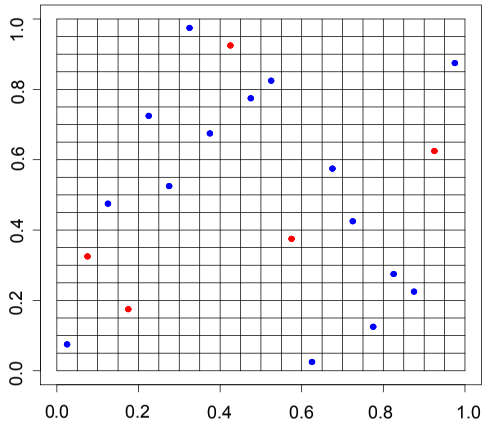
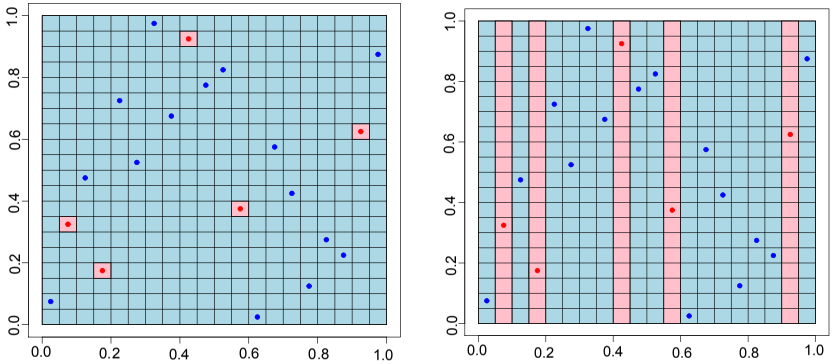


Figure 3: Training Data

vanishing measure, it is possible to obtain consistency in the limit as $n \to \infty$, even while classifying every training point correctly. This stands in direct contrast to the conclusion of others such as Bickel et al. (2006) who have observed that the "empirical optimization problem necessarily led to rules which would classify every training set observation correctly and hence not approach the Bayes rule whatever be n."

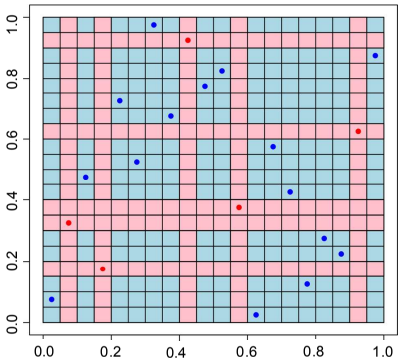
While a classifier such as that in Figure 4a would preform well and is even consistent, many possible interpolators exist, such as the others displayed in Figure 4. Figure 4b shows the (hypothetical) result of allowing the boosting algorithm to use trees involving only x_1 and not x_2. It is interesting to note that this classifier has severely overfit, even though it is a simpler model, depending on only one of the two predictors. The classifier in Figure 4c has an even worse error rate, while the classifier in Figure 4d differs from the Bayes rule with rate $((1-p)n)^2/n^2$. This final example illustrates the type of structure and error rate that occurs when stumps are used as the weak learner. In fact, Mease and Wyner (2008) show that the additive nature of stumps results in boosted classifiers that differ from the Bayes rule at a rate of at least $(1-p)^d(1-1/d)^d$ and hence is not consistent. The reason for this is that using linear combinations of stumps does not provide enough flexibility to interpolate locally around points for which the observed class differs from the Bayes rule. In contrast, boosting larger trees, such as those grown in random forests interpolating with spikes of increasingly smaller size. Some simulations demonstrating the superior performance of larger trees over stumps are given in Mease and Wyner (2008) and here in Section 4.3.

The different classification rules represented by the four plots all interpolate the training data; however, their performances on the population vary considerably due to different degrees of local interpolations of noise. In the sequel, we will show how random forests and boosted ensembles of large trees results in classifiers that are robust to noise. The classifiers behave in noisy regions as in Figure 4a. AdaBoost and random forests average

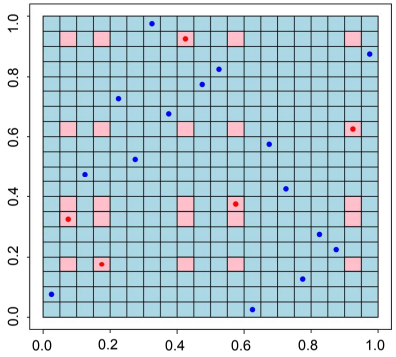


(a) Hypothetical Classifier 1

(b) Hypothetical Classifier 2



(c) Hypothetical Classifier 3



(d) Hypothetical Classifier 4

Figure 4: Four different hypothetical classifiers on a pure noise response surface where $\mathbb{P}(y = 1|\mathbf{x}) = 0.75$.

many individually overfit classifiers, similar to the one in Figure 4b. The final result is a robust classifier, that is spiked-smooth; it fits the noise but only extremely locally.

3.3 A Two-Dimensional Example with Pure Noise

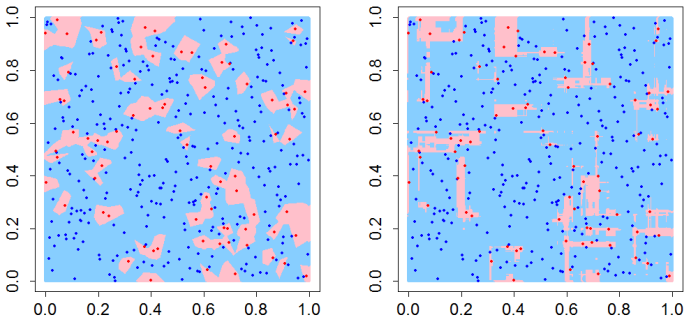
We will begin with an easy to visualize example that demonstrates how fine interpolation can provide robustness in a noisy setting. In particular, we compare the performance of AdaBoost, random forests and one-nearest neighbors, which are all interpolating classifiers. We will see see graphically that AdaBoost and random forests interpolate more locally around error points in the training data than the one-NN classifier. Consequently, AdaBoost and random forests are less affected by noise points as one-NN and have lower generalization error. We will show that the self-averaging property of AdaBoost and random forests is crucial. This property will be discussed in subsequent sections.

The implementation of AdaBoost is carried out according to the algorithm described earlier. The base learners used are trees fit by the `rpart` package (Therneau and Atkinson, 1997) in `R`. The trees are grown to a maximum depth of 8, meaning they may have at most $2^8 = 256$ terminal nodes. This will be the implementation of AdaBoost we will consider throughout the remainder of this paper.

We will consider again the "pure noise" model as described in the previous section, where the probability that y is equal to $+1$ for every $\mathbf{x}$ is some constant value $p > .5$. For the training data we will take $n = 400$ points uniformly sampled on $[0, 1]^2$ according to a Latin Hypercube using the midpoints as before. For the corresponding y values in training data we will randomly choose 80 points to be -1's so that $\mathbb{P}(y = 1|\mathbf{x}) = .8$.

Figure 5 displays the results for the following: (a) one-NN, (b) AdaBoost , and (c) random forests. Regions classified as $+1$ are colored light blue and regions classified as -1 are colored pink. The training data is displayed with blue points for $y = +1$ and red points for $y = -1$. Since the Bayes' rule would be to classify every point as $+1$, we judge the performance of the classifiers by the fraction of the unit square that matches the Bayes' rule. The nearest neighbor rule in this example classifies 79% of the region as $+1$ (we expect $p = 80\%$ on average for the one-NN) while AdaBoost performs substantially better classifying 87% of the square as $+1$ after 100 iterations (which is long after the training error equals zero). This is evidence of boosting's robustness to noise discussed in the previous section. The random forests (with 500 trees) does even better, classifying 94% of the figure as $+1$. Visually, it is obvious that the random forests and AdaBoost classifier is more spiked-smooth than one-nearest neighbors, which allows it to be less sensitive to noise points. AdaBoost and random forests do in fact overfit the noise—but only the noise. They do not allow the overfit to metastasize to modestly larger neighborhoods around the errors. It is interesting to note that there seems to be a large degree of overlap between the regions classified as -1 by both the random forests and AdaBoost; one-NN does not seem to visually follow a similar pattern.

As we will see in the Section 3.6, by increasing the sample size, number of dimensions and iterations the performance is even better. The agreement with the Bayes rule for AdaBoost and random forests converge to practically 100% despite the fact that both algorithms still interpolate the training data without error.



(a) one-NN (b) AdaBoost

(c) Random Forests

Figure 5: Performance of one-NN, AdaBoost, and random forests on a pure noise response surface with $\mathbb{P}(y = 1|\mathbf{x}) = .8$ and $n = 400$ training points.

3.4 A Visualization of Spiked-Smoothing

We have argued that local interpolation such as in Figure 4c is desirable, and we have demonstrated that AdaBoost and random forest classifiers can achieve such a fit in the previous simulation. Now, we turn to the crucial point of how these classifiers achieve such a fit. To this end, we will graphically display the process of spiked-smoothing in the case of the random forest classifier from the previous simulation. Each of the first six plots in Figure 6 shows the classification rule fit by different decision trees in the random forest. We have restricted each plot to a subset of the unit square to aide in visual ease. The bottom plot, Figure 6g shows the classifier created from a majority vote of each of the six random forest decision trees. As in the previous sections, the light blue regions indicate where a classifier returns $y = +1$, and the pink regions indicate where a classifier returns $y = -1$.

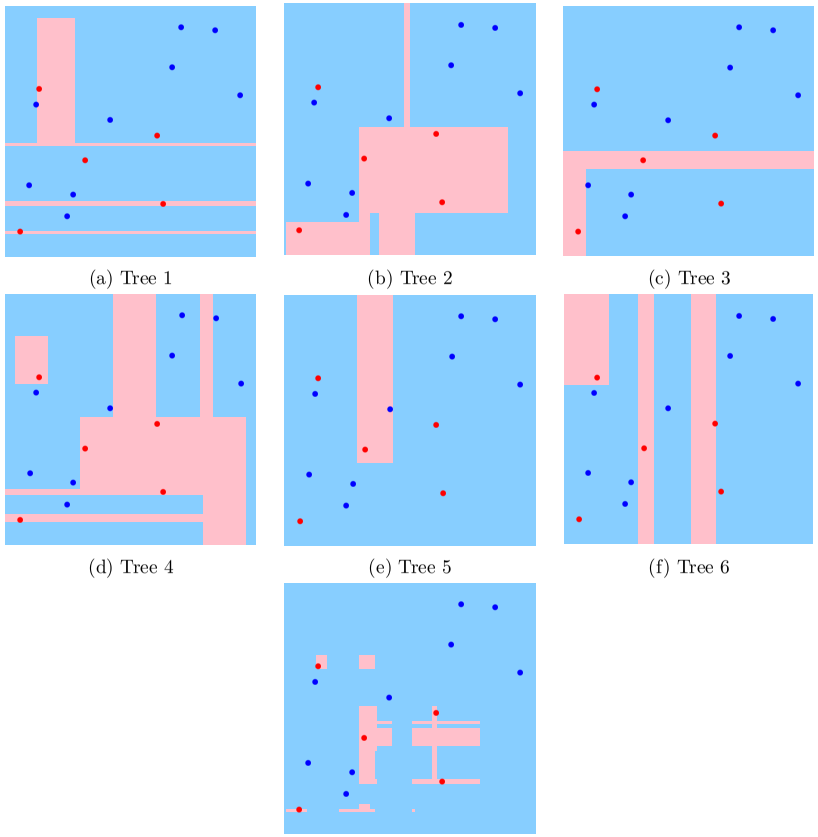
As before, we remark that the Bayes rule in this case would be to classify every point as $y = +1$, and so agreement with the Bayes rule in the plots below can be visualized as the proportion of the figure that is light blue. The first thing to notice is that each decision tree fails to reproduce the Bayes rule. Indeed, since each tree interpolates its bootstrap sample, each figure is bound to contain regions of pink, since most bootstrap samples will contain at least a few noise points. However, one will also notice that these regions of pink tend to be localized into thin strips (this is especially apparent in trees one, three, five, and six). In other words, noise points tend not to ruin the fit of the decision tree at nearby points.. The magic of spiked-smoothing is revealed in the classifier 6g created by a majority vote of the six decision trees. By itself, each decision tree is a poor classifier (evinced by relatively large regions of pink). However, when voted these regions of pink get shrunk down into smaller regions, indicating better agreement with the Bayes rule. One can easily imagine that if these "thin strips" were actually much wider, as in the case of fitting stumps, averaging would not be able to reduce the influence of these noise points enough. The end effect of averaging is to create a decision surface which is affected only very minimally by the noise points in the training set. A simulation in Section 4 will demonstrate that the additional iterations of AdaBoost serve to "shrink" the fit around noise points, much as the regions of pink in this example became more localized after averaging.

3.5 A Two-Dimensional Example with Signal

In light of the example in the previous section, one might note that certain non-interpolating algorithms, such as a pruned CART tree, would recover the Bayes error rate exactly. In this section, we consider an example where a much more complex classifier is required to recover the signal, yet the self-averaging property is still needed to prevent over-fitting to noise.

We consider $n = 1000$ training points sampled uniformly on $[0, 1]^2$ with the Latin Hypercube design. In this simulation, there is signal present. Inside of a circle of radius 0.4 centered in the square, the probability that $y = +1$ is set to 0.1, while the probability that $y = +1$ outside the circle is set to 0.9.

This simulation setting is similar to the previous one, except that the probability that $y = +1$ varies at different points over the unit square. One can see in Figure 7 that the Bayes rule in this setting is just to label every point inside the circle $y = +1$ and every point outside the circle $y = -1$, which gives a Bayes error rate of 0.1. We can then compare



(a) Tree 1 (b) Tree 2 (c) Tree 3

(d) Tree 4 (e) Tree 5 (f) Tree 6

(g) Majority Tree Vote

Figure 6: First six trees from a random forest, along with the classifier created by a majority vote over the trees.

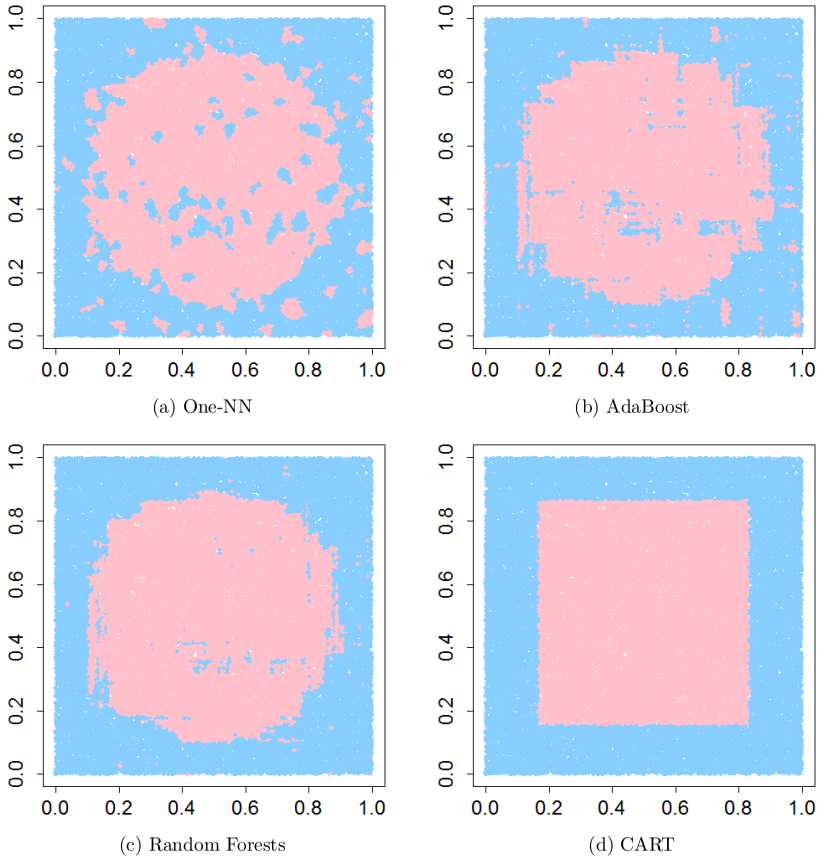
the performance of AdaBoost, random forests, and CART as in the previous section by examining how much of the circle gets classified as $y = +1$ and how much of the outer region is classified as $y = -1$. We run AdaBoost for 500 iterations, fit a random forests model with 500 trees, and build a CART tree that is pruned using cross-validation. Note that we prune the CART tree in order to show how a "classical" statistical model of limited complexity performs on the classification task.

We find that AdaBoost and random forests have an overall error rate of around 0.13, one-nearest neighbor has an overall error rate of 0.20, and CART has an error rate of 0.18. CART fails to perform well in this example because it is not allowed enough complexity to capture the circular pattern. To do so via only the splits parallel to the axes allowed by the algorithm would require a very deep tree (as allowed in random forests and AdaBoost), which pruning does not afford. Rather, a shallow tree can only recover a simple rectangular pattern due to its shallow depth. One-NN, on the other hand, again suffers from its inability to keep the interpolation localized. Outside of the circle, one can observe small "islands" of pink surrounding noise points: by failing to localize the fit, test points near these noise points get classified incorrectly. Again, one finds that random forests and AdaBoost have superior performance because they tend to finely interpolate the training data, and the process of spiked-smoothing shrinks down the influence of noise points.

3.6 A Twenty-Dimensional Example

We now repeat the simulation in Section 3.3 with a larger sample size and in 20 dimensions instead of 2. Specifically, the training data now has $n = 5000$ observations sampled according to the midpoints of a Latin Hypercube design uniformly on $[0, 1]^{20}$. We again randomly select 20% or 1000 of these points to be -1's with the remaining 4000 to be $+1$'s.

Since in 20 dimensions it is difficult to display the resulting classification rules graphically we instead examine the rules on a hold out sample of 10,000 points sampled uniformly and independently on $[0, 1]^{20}$. Figure 8 plots the proportion of points in the hold out sample classified by AdaBoost as $+1$ as a function of the number of iterations. This proportion peaks at .1433 at nine iterations but then gradually decreases to .0175 by 100 iterations and is equal to .0008 by 1,000 iterations. The fact that by 1,000 iterations only 8 of the 10,000 points in the hold out sample are classified as $+1$ means there is very little overfitting. The large number of iterations has the effect of smoothing out the classifier resulting in a rule that agrees with the Bayes rule for 99.92% of the points. Recall that AdaBoost fits the training data perfectly, and thus differs from the Bayes rule on 20% of this sample. We see clearly here that AdaBoost overfits with respect to the training data but not with respect to the population. Again, this is a result of extremely local interpolation of the points in the training data for which the observed class differs from the Bayes rule. A random forests model fit to the training data agrees with the Bayes rule at every point except for one, and hence has exceptional generalization error.



(a) One-NN (b) AdaBoost

(c) Random Forests (d) CART

Figure 7: Performance of AdaBoost, random forests, and CART on a response surface where $\mathbb{P}(y = 1|\mathbf{x}) = 0.10$ inside the circle and $\mathbb{P}(y = 1|\mathbf{x})) = 0.90$ outside of the circle. There are $n = 1000$ training points and the Bayes error is 0.10.

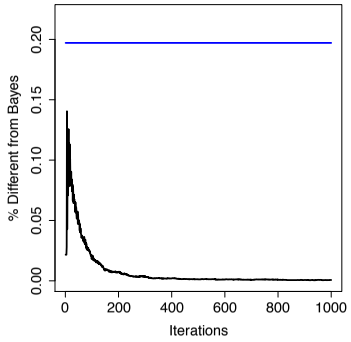


Figure 8: This plot shows the proportion of points in a test set for which the predictions made by AdaBoost and the Bayes rule differ, as a function of the number of boosting rounds (black). The blue line shows this proportion for the one nearest neighbor classifier. Note that the agreement of AdaBoost and the Bayes rule increases with the number of boosting rounds.

4. Self-Averaging Property of Boosting

4.1 Boosting is Self-Smoothing

In the previous sections, we have demonstrated simple examples where random forests and AdaBoost yield the strongest performance with respect to the Bayes rule. We have argued that these algorithms are successful classifiers due to the fact that they fit initially complex models by interpolating the training data but also exhibit smoothing properties via self-averaging that stabilizes the fit in regions with signal, while continuing to keep localized the effect of noise points on the overall fit. While this smoothing mechanism is obvious for random forests via the averaging over decision trees, it is less obvious for AdaBoost. In this section we explain why the additional iterations in boosting way beyond the point at which perfect classification of the training data (i.e interpolation) has occurred actually has the effect of smoothing out the effects of noise rather than leading to more and more overfitting. To the best of our knowledge, this is a novel perspective on the algorithm. To explain our key idea, we will recall the pure noise example from before with $p = .8$, $d = 20$ and $n = 5000$.

Recall that the classifier produced by AdaBoost corresponds to $\mathrm{I}[f_M(x) > 0]$ where

$$f_M(x) = \sum_{m=1}^{M} \alpha_m G_m(x)$$

as defined earlier. Taking $M = 1000$ which was successful in our example let us rewrite this as

$$f_{1000}(x) = \sum_{m=1}^{1000} \alpha_m G_m(x) = \sum_{j=1}^{10} \sum_{k=1}^{100} \alpha_{100(j-1)+k} G_{100(j-1)+k}(x) = \sum_{j=1}^{10} \sum_{k=1}^{100} h_k^j(x)$$

where

$$h_k^j(x) \equiv \alpha_{100(j-1)+k} G_{100(j-1)+k}(x).$$

Now define

$$h_K^j(x) \equiv \sum_{k=1}^{K} h_k^j(x)$$

and note that for every $j \in \{1, ..., 10\}$ and every $K \in \{1, ...100\}$ that $\mathrm{I}[h_K^j(x) > 0]$ is itself a classifier made by linear combinations of classification trees. The ten plots in Figure 9 display the performance on the hold-out sample for these ten classifiers corresponding to the ten different values for j as a function of K. Interestingly, each of these 10 classifiers by itself displays the characteristic of boosting: the agreement with the Bayes rule increases as more terms are added (for instance, as K is increased).

A second interesting fact about these 10 individual classifiers in the decomposition is that each one achieves perfect separation of the training data and thus each one is an interpolating classifier. This result can be expected in general, provided the total number of iterations for each classifier in the decomposition is sufficiently large. This is clear for the first classifier, since it is simply AdaBoost itself and will necessarily achieve zero training error under some standard conditions as discussed in Jiang (2002). The second classifier in the decomposition is simply AdaBoost weight carried over from the first classifier. Since re-weighting the training data does not prevent AdaBoost from obtaining zero training error, the second classifier also interpolates eventually, as does the third, and so on.

Decomposing boosting in this way offers an explanation of why the additional iterations lead to robustness and better performance in noisy environments rather than severe overfitting. In this example, AdaBoost for 100 iterations is an interpolating classifier. It makes some errors, mostly near the points in the training data for which the label differs from the Bayes rule, although these are localized. Boosting for 1000 iterations is thus a point-wise weighted average of 10 interpolating classifiers. The random errors near the points in the training data for which the label differ from the Bayes' rule cancel out in the ensemble average and become even more localized. Of course, the final classifier is still an interpolating classifier as it is an average of 10 interpolating classifiers. In this way, boosting is self-smoothing, self-averaging or self-bagging process that reduces overfitting as the number of iterations increase. The additional iterations provide averaging and smoothing—not overfitting. Empirically this is very similar to random forests and provides evidence that both algorithms, which perform well in our examples, actually do so using the same mechanism.

We further illustrate this phenomenon of increasing localization of the interpolating resulting from this averaging through the following simulation. We take the same training data as before but this time we form the hold out sample by taking a point a (Euclidean) distance of .1 from each of the 1000 points labeled as -1 in the training data in a random direction. Due to the forced (and unnatural) close proximity of the points in the hold out

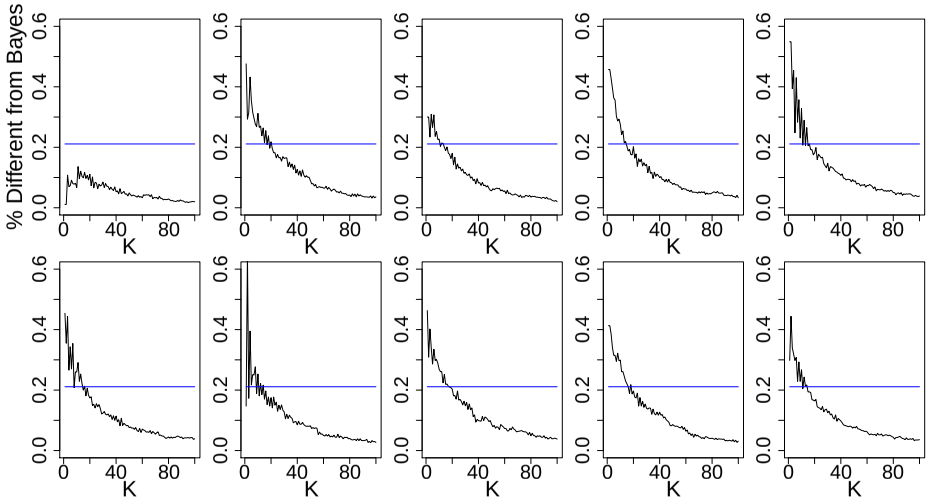


Figure 9: A decomposition of boosting

to training set deviations from the Bayes' rule (points with -1 labels), the error rate is much higher than it would be for a random sample. However, the interpolation continues to become more localized as the iterations proceed (see Figure 10) so even points that are quite close to the label errors (the -1 points) eventually become classified correctly as $+1$. Comparison to Figure 8 shows that this localization continues at a steady rate even after the error on the random hold-out sample is practically zero. In contrast, the nearest neighbor interpolator this simulation yielded 100% disagreement with the Bayes' rule.

4.2 A Five-Dimensional Example

We will now consider a second simulation to further illustrate how this self-averaging property of AdaBoost helps prevent overfitting and improves performance. In this simulation we add signal while retaining significant random noise. Let $n = 400$, $d = 5$ and sample $\mathbf{x}_i$ distributed *iid* uniform on $[0, 1]^5$. The true model from for the simulation is

$$\mathbb{P}(y = 1|\mathbf{x}) = .2 + .6\ \mathrm{I}\left[\sum_{j=1}^{2} x_j > 1\right].$$

The Bayes' error is 0.20 and the optimal Bayes' decision boundary is the diagonal of the unit square in x_1 and x_2. Even with this small sample size, AdaBoost interpolates the training data after 10 iterations. So we boost for 100 iterations which decomposes into ten

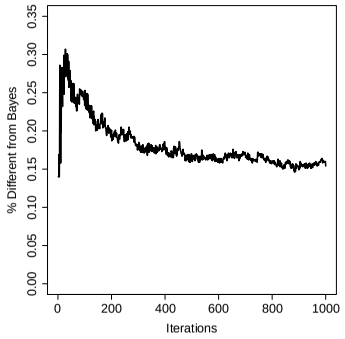


Figure 10: This plot shows the proportion of points in a test set for which the predictions made by AdaBoost and the Bayes rule differ, as a function of the number of boosting rounds (black). The test set has been chosen to contain excess points near the error points in the training set.

sets of ten (which is analogous to the 10 sets of 100 from the 20 dimensional example in the previous section).

The ten plots in the first two rows in Figure 11 show the performance of the ten classifiers corresponding to this decomposition with respect to a hold out sample of 1000 points. Each point in the figure represents a point classified differently from the Bayes rule. While each of the ten classifiers in the decomposition classifies a number of these 1000 points incorrectly especially along the Bayes boundary, exactly which points are classified incorrectly varies considerably from one classifier to the next. The final classifier (displayed in the last plot), which corresponds to AdaBoost after 100 iterations, makes fewer mistakes than each of the ten individual classifiers. Since AdaBoost is a point-wise weighted average of the 10 classifiers, the averaging over the highly variable error locations made by each classifier reduces substantially the number of errors made by the ensemble. The percentage of points classified differently from the Bayes' rule by the final classifier is 118/1000=0.118 while after the first ten there were still 162/1000=0.162 classified differently from the Bayes rule. Averaged over 200 repetitions of this simulation these numbers are .19 after the first ten iterations and .15 after 100 iterations confirming that the performance does improve by running beyond the point at which interpolation initially occurs as (a result of this self-smoothing).

In this example, we also have evidence that AdaBoost takes some measure to decorrelate the errors made by its base classifiers, as hypothesized in Amit et al. (2000) and suggested in Figure 11. To this end, we computed the correlation between $\mathbb{I}[h^1_{10}(X) = Y], \ldots, \mathbb{I}[h^{10}_{10}(X) = Y]$ over a large test set. The correlations ranged from 0.4 to 0.56, with an average value of 0.488. As a comparison, we also considered a similar calculation for the decomposition produced from a random forest with 500 trees, and from bagging 1000 depth 8 trees. As with AdaBoost, we can also decompose these classifiers into a sum of interpolating classifiers: 25

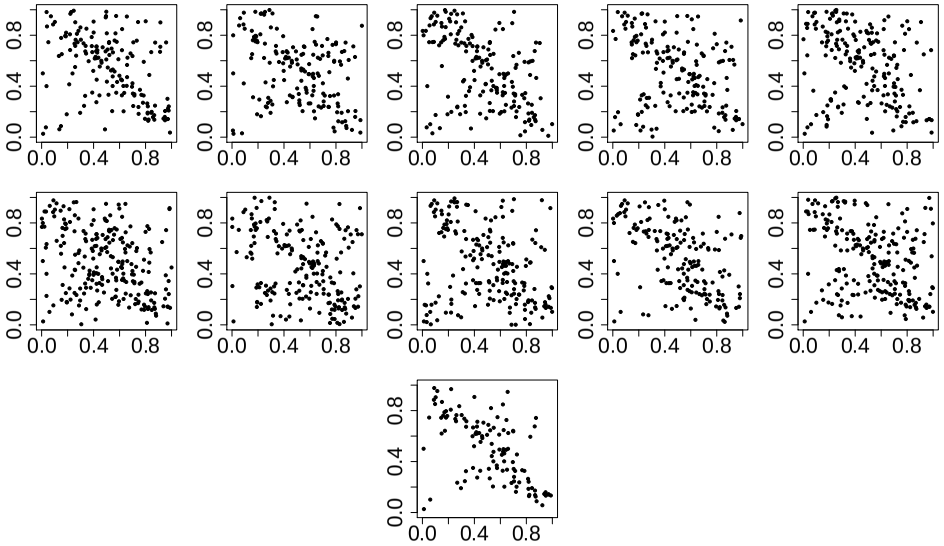


Figure 11: Errors made by each classifier in a decomposition of boosting (first two rows) and errors made by the final classifier (bottom)

sub-ensembles of 20 trees in the case of the random forest, and 10 sub-ensembles of trees in the case of bagged trees. The interpolating classifiers produced by the random forest had error correlations ranging from 0.75 to 0.87, with an average of 0.81, while the correlations for the bagged trees ranged from 0.9 to 0.94, with an average value of 0.92. In other words, the sub-ensembles produced by random forests and AdaBoost are less correlated than the bagged trees, with AdaBoost being markedly less so. When voting across constituents in its decomposition, it is clear that such a decorrelating effect would help to increase the effectiveness of the voting mechanism to cancel out error points. While we observed this phenomenon in a few data sets, we cannot state under what conditions it happens more generally. However, it is know that boosting generally outperforms bagging, and that bagged trees will be much more correlated.

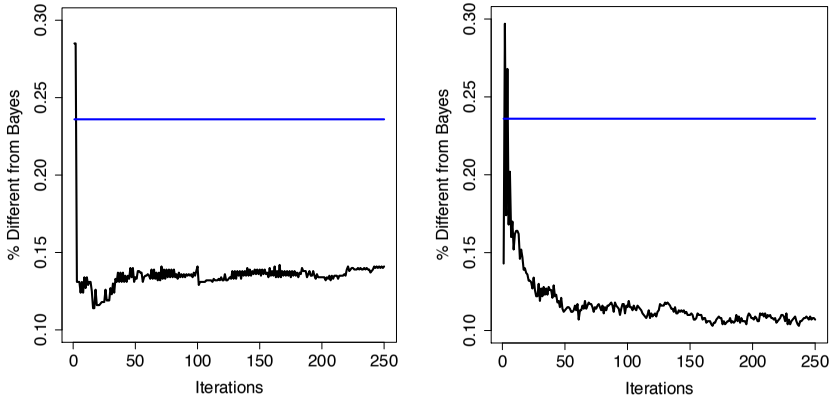
4.3 Comparison to Boosted Stumps

Throughout this paper we have considered AdaBoost with large trees of up to 2^8 terminal nodes. We have shown that AdaBoost with such large trees is a classifier which interpolates the data in such a way that it performs well out of sample for problems in which the Bayes' error rate is substantially larger than zero. In this section we will consider the performance of AdaBoost with stumps as the base learner. The statistical theory predicts that AdaBoost with stumps will overfit less than AdaBoost with larger trees, as expressed, for instance, in Jiang (2002). It is also thought that stumps should be preferable when the Bayes' decision rule is additive in the original predictors. For instance the seminal book by Hastie et al. (2009, chapter 10) advocates using trees of a depth one greater than the dominant level of interaction, which is generally quite low.

AdaBoost with stumps does not self-smooth nearly as well as with larger trees, likely because the the classifiers in the decomposition are more highly correlated, and the "rough" fits from stumps fail to interpolate the training data locally enough; the fit is not spiked-smooth around the training set error points. Consequently, AdaBoost with stumps as base learners is outperformed by AdaBoost with large trees as base learners, when the Bayes error rate is high. This is the case even when the Bayes rule is additive. This result is matched by random forests which works best with large trees. Its randomly chosen predictors at each splitting opportunity lowers the correlation among the trees and the resulting fit is more spiked-smooth.

To illustrate this with an example, we return to the five dimensional simulation from Section 4.2 which has an additive Bayes decision rule. Figure 12 displays the percentage of points that are classified differently from the Bayes rule in the hold out sample of 1000, as a function of the number of iterations. It can be seen that the stumps (left panel) do not perform as well as the 2^8 node trees (right panel). After 250 iterations, AdaBoost with stumps yields 141/1000 points in the hold out sample classified differently from the Bayes rule, compared to 116/1000 for this same data set using instead AdaBoost with 2^8 node trees. In fact, the stumps seems to suffer from overfitting when run beyond only 25 iterations, while the 2^8 trees do not have a problem with overfitting as the number of iterations are increased.

These numerical values are based only on a single run of this simulation, but the qualitative finding is reproducible over repeated runs. The result serves to illustrate that the



(a) Stump (b) 2^8 Node Trees

Figure 12: Comparison of AdaBoost with stumps (a) and 2^8 node trees (b) for the five dimensional simulation. The blue line corresponds to One-NN.

good out-of-sample performance of AdaBoost using large trees resulting from the local interpolation of noise points is not shared by AdaBoost using stumps. The idea that stumps will perform better in noisy environments because they overfit less is not supported by this simulation. While the stumps do overfit less on the training data, as evidenced by the fact that they did not give zero training error, they actually overfit worse than the larger trees out of sample. Again, we attribute this to boosting with stumps lacking the self-smoothing property and not being flexible enough to interpolate the noise locally.

5. Real Data Example

In previous sections, we explored the mechanism of spiked smoothing in simulation experiments where it was easy for us to identify noise points. Recall that with a fully specified probability model, noise points are simply sample points whose sign differs from the Bayes' rule. While simulated examples are certainly good for illustration, one may wonder whether these settings are overly simplistic. Data in the real world is typically generated by more complicated—and unknown—probability models with heteroskedastic noise components. Since boosted trees have been empirically successful in such settings, it pays to move our discussion to the context of a data set arising in the real world.

5.1 Phoneme Data

In this section, we will discuss spiked smoothing in the context of a data set arising in a speech recognition task designed to discriminate "nasal" vowels from "oral" vowels in spoken

language [2]. For a collection of 5404 examples of spoken vowels, this data set contains the amplitudes of the first 5 harmonic frequencies as collected by a cochlea spectra (hearing aid), along with a label of either nasal or oral (n=5404, p=5). The goal is then to use these harmonic frequencies to correctly identify the type of vowel. While there are many potential data sets we could have considered, this one is convenient to analyze since it consists of a relatively small number of real valued covariates. This makes it easier to talk about "neighborhoods" of points later in our discussion.

We begin by dividing the phoneme data set up into a training set consisting of 70% of the samples, and a testing set consisting of 30% of the examples. Depth eight decision trees boosted for 1000 rounds and a random forest classifier both achieve comparable test error rates of 9.0% and 9.4%, respectively [3]. Figure 13 demonstrates that boosting deep trees is preferable to shallow trees in this data set. Each frame in the figure shows the testing error in black and the training error in red as a function of the number of boosting iterations for different depth trees. One can readily observe that test error steadily decreases with the depth of tree used in AdaBoost. It is interesting to note that boosting trees of all depths are slow to overfit the data, even with boosted stumps after 1000 iterations. It is also worth noting that boosted depth 8 trees quickly interpolate the training data, as indicated by the sharply decreasing red line. This raises the often asked question: what is AdaBoost doing after it achieves a perfect fit on the training set?

Any discussion of spiked smoothing on real data is complicated by the reality that it is impossible to identify noise points without knowing the underlying probability model. As a substitute for noise points, we flipped the signs of 100 randomly chosen points in our training data set (about 2% of the data) and analyzed the fits of boosted trees and a random forest classifier around these points. After refitting the models to the perturbed data, we find the the testing errors are 10.2% and 10.0% for boosted depth eight trees and a random forest, respectively. Figure 14 illustrates that even after thousands of rounds of boosting, test error continues to decrease. The punchline is that both algorithms are able to achieve fits with comparable test set error even after flipping the sign of a large number of points in the training set.

As a comparison with another interpolating classifier, we also repeated the same experiment with a one nearest-neighbors classifier. We found the one nearest neighbor achieved a test error rate of 10.5% when fit on the original training set, and a test error rate of 12.6% when fit on the noisy training set. The increase in error rate when noise is added to the training set is larger than that of AdaBoost or a random forest, which coincides with the results form the simulated example in section 3.3. Furthermore, two sample t-tests reveal that the increase in error rate for one nearest-neighbors is significant at the 0.01 level in both cases (p-values less than $1e^{-16}$ in both cases).

We will argue that additional rounds of boosting are helping to "smooth out" and "localize" the influence of the 100 noise points that we introduced into our training set. In section 3.3, we illustrated the action of spiked smoothing in diagrams which showed AdaBoost's fit on a two dimensional plane. In this slightly higher dimensional example,

2. The original data can be found at the following address: https://www.elen.ucl.ac.be/neural-nets/Research/Projects/ELENA/databases/REAL/phoneme/phoneme.txt.
3. The error rates reported are the result from repeating the fitting procedure on 100 random train/test set splits and considering the average error on the test set.

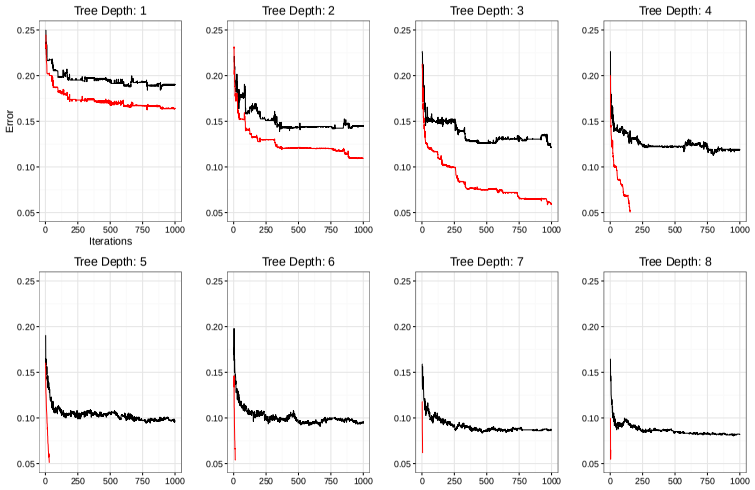


Figure 13: Plots of testing and training error as a function of number of boosting iterations for trees of different depths. The black lines show the test error rates while the red lines show training error rates. Notice that depth seven and eight boosted trees quickly fit the training data, as depicted by the rapidly decreasing red lines.

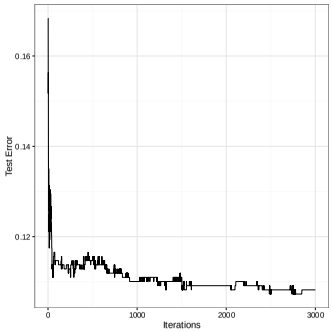
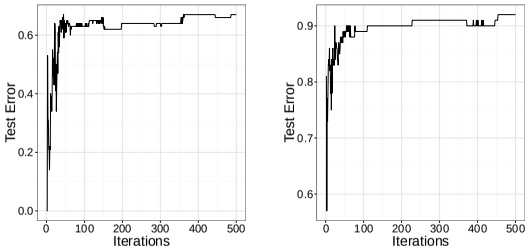


Figure 14: Plot of testing error for depth 8 boosted trees fit to a training set which has flipped labels for 100 randomly selected examples.

we instead consider small neighborhoods around each of the noise points and track the fraction of points in these neighborhoods that agree with the sign of the "correct label",

that is, before flipping the sign. [4]. Panels (a) and (b) of Figure 15 plot this fraction as a function of the number of boosting iterations for two representative "noise points." In both panels, it is clear that as the number of iterations increases, the fraction of points in each neighborhood that agrees with the original sign of the training point increases. Recall that in this case AdaBoost still fits its training data perfectly: the in-sample AdaBoost fit agrees with the sign of the flipped training point in both figures. Despite this, the algorithm still fits a majority of points in a neighborhood of each of these noise points in the correct way. The classifier is producing a spiked smooth fit, that is, it fits the data in such a way to localize the influence of noise points. One can interpret the increasing homogeneity in each neighborhood as the result of averaging. The first few iterations of deeply boosted trees produce a fit that interpolates the training data, but this fit is quite complicated in the sense that it assigns large numbers of points in the neighborhood to both '+1' and '-1'. As the number of iterations increases, this fraction increases so that the fit becomes more smooth in the classical sense.



(a) (b)

Figure 15: The fraction of points in a neighborhood that agree with the sign of the original training point before its sign was flipped. Figures (a) and (b) plot this proportion for two different, representative noise points.

5.2 Additional Data Sets

We repeated a version of the analysis conducted in Section 5.1 for five additional data sets from the UCI repository: Haberman, Wisconsin breast cancer, voting, Pima, and German credit. In our analysis we did the following: we added 5% label noise to the training data, missing values were imputed with the mean prior to model fitting, and all experiments were conducted on 50 random training-testing splits of each data set. Table 1 reports the mean increase in testing error after adding 5% label noise over the 50 random training-testing splits. For example, on the original haberman data set, AdaBoost achieved an average error rate of 34.225%, and on the noisy version of the data set achieved an average error rate of 34.354%. The mean difference of 0.13% is reported in the table. It is also apparent in each setting that AdaBoost and random forest are relatively immune to the addition of label noise. The stars in the Table 1 report the significance level when comparing the increase in error rate for AdaBoost and a random forest with that of one-nearest neighbors using a two-sample t-test. With the exception of the German credit data, the increase in error rate for one nearest neighbors was larger than that of one nearest neighbors with statistical significance.

4. We choose each neighborhood to be a rectangle centered at the point of interest, with side lengths chosen in such as way that the neighborhood contains only a small number of training points. We then chose 100 points uniformly at random in this rectangle and computed the AdaBoost classifier at each point: these points are obviously not included in the original data set.

Data set	AdaBoost	Random Forest	1-NN
Haberman	0.13^{**}	0.52^{*}	**1.55**
breast_cancer	0.20^{***}	0.39^{***}	**2.29**
voting	1.63^{**}	0.30^{***}	**2.71**
Pima	0.56^{***}	0.45^{***}	**1.75**
German	0.29	0.68	**0.68**

Table 1: The increase in average testing error after changing the sign of 5% of the training data. The stars in the table report the significance level when comparing the increase in error rate for AdaBoost and a random forest with that of one-nearest neighbors using a two-sample t-test. One star denotes significance at the 10% level, two stars denotes significance at the 5% level, and three stars denotes significance at the 1% level.

6. Concluding Remarks

AdaBoost is an undeniably successful algorithm and random forests is at least as good, if not better. But AdaBoost is as puzzling as it is successful; it broke the basic rules of statistics by iteratively fitting even noisy data sets until every training set data point was fit without error. Even more puzzling, to statisticians at least, it will continue to iterate an already perfectly fit algorithm which lowers generalization error. The statistical view of boosting understands AdaBoost to be a stage wise optimization of an exponential loss, which suggest (demands!) regularization of tree size and control on the number of iterations. In contrast, a random forest is not an optimization; it appears to work best with large trees and as many iterations as possible. It is widely believed that AdaBoost is effective because it is an optimization, while random forests works—well because it works. Breiman conjectured that "it is my belief that in its later stages AdaBoost is emulating a random forest" (Breiman, 2001). This paper sheds some light on this conjecture by providing a novel intuition supported by examples to show how AdaBoost and random forest are successful for the same reason.

A random forests model is a weighted ensemble of interpolating classifiers by construction. Although it is much less evident, we have shown that AdaBoost is also a weighted ensemble of interpolating classifiers. Viewed in this way, AdaBoost is actually a "random" forest of forests. The trees in random forests and the forests in the AdaBoost each interpolate the data without error. As the number of iterations increase the averaging of decision surface because smooths but nevertheless still interpolates. This is accomplished by whittling down the decision boundary around error points. We hope to have cast doubt on the commonly held belief that the later iterations of AdaBoost only serve to overfit the data.

Instead, we argue that these later iterations lead to an "averaging effect", which causes AdaBoost to behave like a random forest.

A central part of our discussion also focused on the merits of interpolation of the training data, when coupled with averaging. Again, we hope to dispel the commonly held belief that interpolation always leads to overfitting. We have argued instead that fitting the training data in extremely local neighborhoods actually serves to prevent overfitting in the presence of averaging. The local fits serve to prevent noise points from having undue influence over the fit in other areas. Random forests and AdaBoost both achieve this desirable level of local interpolation by fitting deep trees. It is our hope that our emphasis on the "self-averaging" and interpolating aspects of AdaBoost will lead to a broader discussion of this classifier's success that extends beyond the more traditional emphasis on margins and exponential loss minimization.

References

Yali Amit, Gilles Blanchard, and Kenneth Wilder. Multiple randomized classifiers: Mrcl. Technical report, 2000.

Joshua Belanich and Luis E Ortiz. On the convergence properties of optimal adaboost. *arXiv preprint arXiv:1212.1108*, 2012.

Peter J. Bickel, Ya'acov Ritov, and Alon Zakai. Some theory for generalized boosting algorithms. *The Journal of Machine Learning Research*, 7:705–732, 2006.

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

Leo Breiman. Arcing classifier (with discussion and a rejoinder by the author). *The Annals of Statistics*, 26(3):801–849, 1998.

Leo Breiman. Prediction games and arcing algorithms. *Neural computation*, 11(7):1493–1517, 1999.

Leo Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

Peter Bühlmann and Bin Yu. Boosting with the l 2 loss: regression and classification. *Journal of the American Statistical Association*, 98(462):324–339, 2003.

Corinna Cortes, Mehryar Mohri, and Umar Syed. Deep boosting. In *Proceedings of the Thirty-First International Conference on Machine Learning (ICML 2014)*, 2014.

Thomas M Cover and Peter E Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.

Robert M. Freund, Paul Grigas, and Rahul Mazumder. Adaboost and forward stagewise regression are first-order convex optimization methods. *arXiv preprint arXiv:1307.1192*, 2013.

Y. Freund and R.E. Shapire. Discussion of additive logistic regression: A statistical view of boosting. *Annals of Statistics*, 28:337–374, 2000.

Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *ICML*, volume 96, pages 148–156, 1996.

Jerome Friedman. Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232, 2001.

Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The Annals of Statistics*, 28(2):337–407, 2000.

Wei Gao and Zhi-Hua Zhou. On the doubt about margin explanation of boosting. *Artificial Intelligence*, 203:1–18, 2013.

Carlos Guestrin. Pac-learning, vc dimension and margin-based bounds. *Machine Learning*, 10701:15781, 2006.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*, volume 2. Springer, 2009.

Wenxin Jiang. On weak base hypotheses and their implications for boosting regression and classification. *Annals of statistics*, pages 51–73, 2002.

Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

Senén Barro Manuel Fernández-Delgado, Eva Cernadas and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *The Journal of Machine Learning Research*, 15(1):3133–3181, 2014.

David Mease and Abraham Wyner. Evidence contrary to the statistical view of boosting. *The Journal of Machine Learning Research*, 9:131–156, 2008.

Indraneel Mukherjee, Cynthia Rudin, and Robert E Schapire. The rate of convergence of adaboost. *The Journal of Machine Learning Research*, 14(1):2315–2347, 2013.

Alex Niculescu-Mizil and Rich Caruana. An empirical comparison of supervised learning algorithms using different performance metrics. Technical report, Cornell University, 2005.

Greg Ridgeway. Generalized boosted regression models. *Documentation on the R Package gbm, version 1· 5*, 7, 2006.

Robert E. Schapire. Explaining adaboost. In *Empirical Inference*, pages 37–52. Springer, 2013.

Robert E Schapire and Yoav Freund. *Boosting: Foundations and algorithms*. MIT press, 2012.

Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of statistics*, pages 1651–1686, 1998.

Terry M. Therneau and Elizabeth J. Atkinson. An introduction to recursive partitioning using the rpart routines. Technical report, Technical Report 61. URL http://www. mayo. edu/hsr/techrpt/61. pdf, 1997.

Stefan Wager and Guenther Walther. Uniform convergence of random forests via adaptive concentration. *arXiv preprint arXiv:1503.06388*, 2015.

Liwei Wang, Masashi Sugiyama, Zhaoxiang Jing, Cheng Yang, Zhi-Hua Zhou, and Jufu Feng. A refined margin analysis for boosting algorithms via equilibrium margin. *The Journal of Machine Learning Research*, 12:1835–1863, 2011.

Abraham J. Wyner. On boosting and the exponential loss. In *Proceedings of the Ninth Annual Conference on AI and Statistics Jan*, pages 3–6. Citeseer, 2003.

Journal of Machine Learning Research 18 (2017) 1-47 Submitted 12/15; Revised 11/16; Published 5/17

Clustering from General Pairwise Observations with Applications to Time-varying Graphs*

Shiau Hong Lim SHONGLIM@SG.IBM.COM
IBM Research
10 Marina Boulevard
Singapore 018983

Yudong Chen YUDONG.CHEN@CORNELL.EDU
School of Operations Research and Information Engineering
Cornell University
Ithaca, NY 14853, USA

Huan Xu HUAN.XU@ISYE.GATECH.EDU
School of Industrial and Systems Engineering
Georgia Institute of Technology
755 Ferst Drive, NW, Atlanta, GA 30332, USA

Editor: Edo Airoldi

Abstract

We present a general framework for graph clustering and bi-clustering where we are given a general observation (called a label) between each pair of nodes. This framework allows a rich encoding of various types of pairwise interactions between nodes. We propose a new tractable and robust approach to this problem based on convex optimization and maximum likelihood estimators. We analyze our algorithms under a general statistical model extending the planted partition and stochastic block models. Both sufficient and necessary conditions are provided for successful recovery of the underlying clusters. Our theoretical results subsume many existing graph clustering results for a wide range of settings, including planted partition, weighted clustering, submatrix localization and partially observed graphs. Furthermore, our results are applicable to novel settings including time-varying graphs, providing new insights to solutions of these problems. We provide empirical results on both synthetic and real data that corroborate with our theoretical findings.

Keywords: graph clustering, convex optimization, low-rank matrix, information divergence, time-varying graphs, pairwise observation, dynamic graphs

1. Introduction

In the standard formulation of graph clustering, we are given an unweighted graph and seek a partitioning of the nodes into disjoint groups such that members of the same group are more densely connected than those in different groups. Here, the presence of an edge represents certain affinity or similarity between the nodes, and the absence of an edge represents the lack thereof.

In many applications, from chemical interactions to social networks, the interactions between nodes are much richer than a simple "edge" or "non-edge". Such extra information can be used to improve the clustering quality. We may represent each type of pairwise interaction by a *label*. One simple setting of this type is weighted graphs, where instead of a 0-1 graph, we have edge weights representing the strength of the pairwise interaction. In this case the observed label between each pair is a real number. In a more general setting, the label need not be a number. For example, on social networks like Facebook, the label between two persons may be "they are friends", "they went to different schools", "they liked 21 common pages", or the concatenation of them. In such cases different labels carry different information about the underlying community structure. A standard approach that converts these pairwise interactions into a simple edge/non-edge and then applies standard clustering algorithms, often does not work well here, as much of the information may be lost. Even in the simple case of a standard weighted/unweighted graph, it may not be immediately clear how information in the graph should be used in clustering. For example, should the absence of an edge be interpreted as a *neutral* observation carrying no information, or as a *negative* observation indicating dissimilarity between the two nodes?

We emphasize that our notion of labels (types of pairwise observations) is very general. A label can take real, discrete, categorical or even mixed values—we will see several such examples in Section 4. A label can even take the form of a time series, i.e., a record of time varying interactions such as "A and B messaged each other on June 1st, 4th, 15th and 21st", or "they used to be friends, but they stop talking to each other since 2012". The labeled graph model is therefore an immediate tool for analyzing time-varying graphs.

In this paper, we present a new and principled approach for graph clustering that is directly based on general pairwise labels. We assume that between each pair of nodes i and j, one observes a label L_{ij} that takes values in a label set $\mathcal{L}$. The set $\mathcal{L}$ may be continuous, discrete or mixed, and need not have any algebraic or geometric structure. The standard graph model corresponds to a binary label set $\mathcal{L} = \{\text{edge}, \text{non-edge}\}$, and a weighted graph corresponds to $\mathcal{L} = \mathbb{R}$. Given the matrix of observed labels $L = (L_{ij}) \in \mathcal{L}^{n\times n}$ between n nodes, the goal is to partition these nodes into disjoint clusters. Our algorithmic approach is based on finding a partition that optimizes a weighted objective that is appropriately constructed from the observed labels. This formulation leads to a combinatorial optimization problem, and our algorithms use its convex relaxation.

To systematically evaluate clustering performance, we consider a generalization of the stochastic block model and the planted partition model (Holland et al., 1983; Condon and Karp, 2001). Our model assumes that the observed labels are generated based on an underlying set of ground truth clusters, where node pairs from the same cluster generate labels using a distribution μ over $\mathcal{L}$, and pairs from different clusters use a different distribution ν. The standard planted partition model corresponds to the case where μ and ν are Bernoulli distributions with $\mu(\text{edge}) = p$ and $\nu(\text{edge}) = q, p \neq q$. We provide theoretical guarantees for our algorithm under this generalized model.

By specializing to concrete examples of the distributions μ and ν, our results cover a wide range of clustering settings—with theoretical guarantees matching or stronger than existing work—including the standard stochastic block model, partially observed graphs, weighted graphs and submatrix localization. Our framework in fact allows us to handle new classes of problems that are not a priori obvious to be a special case of our model, including the clustering of time-varying graphs.

Our framework easily generalizes to the bi-clustering setting, where pairwise labels are observed between two disjoint sets of nodes, potentially from different domains, and the task is to jointly cluster these two sets of nodes given the bipartite label observations. All our algorithmic and statistical results extend to this bi-clustering setting with only minor changes.

*. Partial preliminary results are presented in the conference papers Chen et al. (2014b) and Lim et al. (2014).

©2017 Shiau Hong Lim, Yudong Chen and Huan Xu.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-659.html.

Remark 1 *Preliminary versions of some of the results here have appeared in part in Chen et al. (2014b) and Lim et al. (2014). The theory was previously stated with respect to only discrete label sets, but are now extended to general label sets, including continuous and mixed-valued labels, which appear frequently in applications (see, for example, Section 4). We also include a more detailed and systematic discussion on the bi-clustering setting and various special cases. Implementation details of efficient first-order solvers are now included. We also report a much more extensive set of empirical results on both synthetic and real data, including comparison with other methods.*

1.1 Related Work

The planted partition model/stochastic block model (Condon and Karp, 2001; Holland et al., 1983) are standard models for studying graph clustering. Variants of the models cover partially observed graphs (Oymak and Hassibi, 2011; Chen et al., 2014a), weighted graphs and the submatrix localization and bi-clustering problems (Balakrishnan et al., 2011; Kolar et al., 2011). All these models are special cases of ours. Various algorithms have been proposed and analyzed under these models, such as spectral clustering (McSherry, 2001; Chaudhuri et al., 2012; Rohe et al., 2011), convex optimization approaches (Mathieu and Schudy, 2010; Ames and Vavasis, 2011) and tensor decomposition methods (Anandkumar et al., 2014). Ours is based on convex optimization, generalizing the convexified maximum likelihood approach in Chen et al. (2014c).

Time-varying graphs arise in a broad range of applications, and the problem of clustering such graphs has been studied in various context (see Fortunato, 2010; Sun et al., 2007; Chakrabarti et al., 2006; Kawadia and Sreenivasan, 2012; Nguyen et al., 2011, and the references therein). The stochastic block model has also been extended in a number of ways to accommodate time-varying graphs. For example, Han et al. (2015) consider multiple, independent graphs, where edge distributions can differ in each graph and each cluster membership pair. In the work of Xu and Hero (2014); Matias and Miele (2016), the cluster membership of an individual node is allowed to change with time. Another extension is in allowing mixed cluster membership in time-varying networks, as done by Fu et al. (2009). These extensions allow additional flexibilities, but the existing solutions lack the kind of theoretical performance guarantees afforded by our approach. Also note that dealing with time-varying graph is only *one* of the applications of our general theory on clustering based on pairwise labels.

Most related to our setting is the *labelled stochastic block model* proposed by Heimlicher et al. (2012) and Lelarge et al. (2013). A main difference in their model is that they assume each pairwise observation is from a two-step process: first an edge/non-edge is observed; if it is an edge then a label is associated with it. In our model all observations are in the form of labels—in particular, an edge or no-edge is also a label—which covers their setting as a special case. Our model is therefore more general and natural, and as a result our theory covers a broad class of subproblems including time-varying graphs. Moreover, their analysis is mainly restricted to the two-cluster setting with edge probabilities on the order of $\Theta(1/n)$, while we allow for an arbitrary number of clusters and a wide range of edge/label distributions. In addition, we consider the setting where the distributions of the labels are not precisely known. Algorithmically, they use belief propagation (Heimlicher et al., 2012) and spectral methods (Lelarge et al., 2013).

Appearing after the conference versions of this paper, the work by Jog and Loh (2015) studies a form of labelled stochastic block model and thus is also related to ours. Restricting to the homogeneous case with equal-size clusters, they derive recovery conditions in terms of the Renyi divergence. Their results are based on the maximum likelihood decoder, which is not computationally feasible. Another recent work by Chen et al. (2015) also studies the statistical limits of information recovery from pairwise observations. Their model is however quite different from ours, and the results are again based on the intractable maximum likelihood decoder.

2. Problem Setup and Algorithms

We assume n nodes are partitioned into r disjoint clusters of size at least K. The clusters are unknown and considered as the ground truth. For each pair of nodes (i, j), a label $L_{ij} \in \mathcal{L}$ is observed, where $\mathcal{L}$ is the set of all possible values of the label. These labels are generated independently across node pairs according to some distributions μ and ν on $\mathcal{L}$.[1] In particular, if nodes i and j are in the same cluster, the observed label L_{ij} follows the distribution μ, otherwise L_{ij} follows ν. The goal is to recover the ground truth clusters given the pairwise labels, which is represented as a matrix $L = (L_{ij}) \in \mathcal{L}^{n\times n}$. We encode the true clusters as an $n \times n$ *cluster matrix* Y^*, where $Y^*_{ij} = 1$ if nodes i and j belong to the same cluster and $Y^*_{ij} = 0$ otherwise, with the convention that $Y^*_{ii} = 1$ for all i. The problem is therefore to find Y^* given L.

We take an optimization approach to this problem. To motivate our algorithm, first consider the case of clustering a weighted graph, where $\mathcal{L} = \mathbb{R}$ and all labels are real numbers. Suppose that positive weights indicate affinity between node pairs while negative weights indicate dissimilarity. A natural approach is to partition the nodes in a way that maximizes the total weight inside the clusters (this is equivalent to *correlation clustering* by Bansal et al. 2004). Mathematically, this formulation is to find a clustering, represented by a cluster matrix $Y \in \{0, 1\}^{n\times n}$, such that the sum $\sum_{i,j} L_{ij}Y_{ij}$ is maximized. In the setting of general labels, we pick a *weight function* $w : \mathcal{L} \to \mathbb{R}$, which assigns a number $W_{ij} = w(L_{ij})$ to the label L_{ij} observed at each pair (i, j). We then solve the following max-weight problem:

$$\begin{aligned} \max_{Y} \quad & \langle W, Y\rangle \\ \text{s.t.} \quad & Y \text{ is an } n \times n \text{ cluster matrix,} \end{aligned} \tag{1}$$

where $\langle W, Y\rangle := \sum_{i,j} W_{ij}Y_{ij}$ is the trace inner product. Note that once the weight function is specified, this formulation effectively converts the problem of clustering from labels into a weighted clustering problem.

The optimization program (1) is non-convex due to the combinatorial constraint. Our algorithm is based on a convex relaxation of (1), using the fact that any cluster matrix is a symmetric positive semidefinite, block-diagonal 0-1 matrix. Relaxing the constraint in (1) leads to the following convex optimization problem:

$$\begin{aligned} \max_{Y} \quad & \langle W, Y\rangle \\ \text{s.t.} \quad & Y \in \mathcal{S}^n_+, \\ & 0 \le Y_{ij} \le 1, \forall (i, j), \end{aligned} \tag{2}$$

where $\mathcal{S}^n_+$ denotes the set of $n \times n$ symmetric positive semidefinite matrices. We say that the program (2) recovers the true clusters if it has a unique optimal solution equal to the true cluster matrix Y^*.

1. More precisely, we assume that there is a σ-algebra $\mathcal{F}$ such that $(\mathcal{L}, \mathcal{F})$ is a measurable space, endowed with probability measures μ and ν.

One has the freedom of choosing the weight function w. Here, the likelihood ratio between label distributions μ and ν will play an important role. We assume that the measures μ and ν are absolutely continuous with each other, as well as with some base measure λ on $\mathcal{L}$. With a slight abuse of notation, the likelihoods of a label $l \in \mathcal{L}$ with respect to μ and ν are given by

$$\mu(l) = \frac{\mathrm{d}\mu}{\mathrm{d}\lambda}(l) \quad \text{and} \quad \nu(l) = \frac{\mathrm{d}\nu}{\mathrm{d}\lambda}(l),$$

where $\frac{\mathrm{d}\mu}{\mathrm{d}\lambda}$ and $\frac{\mathrm{d}\nu}{\mathrm{d}\lambda}$ are the respective Radon-Nikodym derivatives. In the sequel, unless specified otherwise, we will always assume that λ is the counting measure if $\mathcal{L}$ is a discrete label set, in which case we simply have $\mu(l) = \Pr(L_{ij} = l)$. For a continuous set $\mathcal{L}$, we use the Lebesgue base measure and hence $\mu(l)$ is the usual density function of μ. Later we will encounter more general, mixed-valued label sets $\mathcal{L}$.

Intuitively, a weight function w should assign $w(L_{ij}) > 0$ to a label L_{ij} with $\mu(L_{ij}) > \nu(L_{ij})$, so the program (2) is encouraged to place nodes i and j in the same cluster, the more likely possibility; similarly we should have $w(L_{ij}) < 0$ if $\mu(L_{ij}) < \nu(L_{ij})$. In other words, a good weight function should reflect the information in μ and ν. Our theoretical results in Section 3 characterize the performance of the program (2) for *any* given weight function w. Building on this general result, we further derive robust and optimal choices of w. Since μ and ν are often unknown, Section 3 also includes results when w is chosen based on distributions that are different from the true distributions.

For cluster recovery to be possible, the observed labels must contain sufficient information to distinguish different clusters. If $\mu = \nu$, the observed labels for intra- and inter-cluster pairs will have the same distribution, in which case the data L is distributed independently of the underlying clusters and recovery is impossible. In general, the problem becomes harder if μ and ν are more similar to each other. We quantify this relation precisely in Section 3.

2.1 Alternative Formulations and the Bi-clustering Problem

Program (2) uses the fact that Y^* is symmetric positive semidefinite. The following program, which is based on a more relaxed constraint using the nuclear norm $\|\cdot\|_*$,[2] also works and has essentially the same theoretical guarantees:

$$\begin{aligned} \max_{Y} \quad & \langle W, Y \rangle, \\ \text{s.t.} \quad & \|Y\|_* \le \|Y^*\|_* \\ & 0 \le Y_{ij} \le 1, \forall (i,j). \end{aligned} \tag{3}$$

Intuitively, this relaxed problem uses the fact that a cluster matrix always satisfies $\|Y^*\|_* = n$, which is much smaller than a general $n \times n$ binary matrix.

The formulation (3) has the advantage that it applies directly to a bi-clustering setting. In this setting, there are two disjoint sets of nodes $\mathcal{N}_1$ and $\mathcal{N}_2$, where $|\mathcal{N}_1| = n_1$ and $|\mathcal{N}_2| = n_2$. Each of the sets $\mathcal{N}_1$ and $\mathcal{N}_2$ are partitioned into r clusters $\{C_k, k \in [r]\}$ and $\{C'_k, k \in [r]\}$, respectively, where the clusters C_k and C'_k are associated with each other and called a bi-cluster. Similarly to the clustering case, the true cluster matrix $Y^* \in \{0,1\}^{n_1 \times n_2}$ is defined as $Y^*_{ij} = 1$ if and only if $(i,j) \in C_k \times C'_k$ for some $k \in [r]$. Labels are generated from the distribution μ for each pair (i,j) with $Y^*_{ij} = 1$, and from ν otherwise. Here Y^* is not necessarily a square or positive semidefinite matrix. The program (3) still applies, with the understanding that the matrices W and Y have size $n_1 \times n_2$ instead of $n \times n$.

While we always have $\|Y^*\|_* = n$ in the clustering case, this is not the case in general for bi-clustering. The value of $\|Y^*\|$ used in the constraint of (3) is usually unknown in practice, in which case we can instead solve an equivalent formulation in terms of a Lagrange multiplier:

$$\begin{aligned} \max_{Y} \quad & \langle W, Y \rangle - \eta \|Y\|_* \\ \text{s.t.} \quad & 0 \le Y_{ij} \le 1, \forall (i,j). \end{aligned} \tag{4}$$

The formulations (2)–(4) are semidefinite programs that can be solved in polynomial time by various methods. In Section 5 we describe efficient first order solvers and discuss other implementation details. Note that by convex duality, if the constrained formulation (3) succeeds in recovering the true cluster matrix Y^*, then there exists a multiplier η for which the Lagrangian formulation (4) also succeeds.[3] Therefore, all our theoretical results for the formulations (2) and (3) in Section 3 also hold for the formulation (4) with a suitable η.

3. Theoretical Results

In this section, we provide theoretical analysis for the performance of the convex formulations (2) and (3) under the statistical model described in Section 2. We give sufficient and necessary conditions for cluster recovery, and discuss choices of the weight function. All our results apply to both clustering and bi-clustering settings. In the bi-clustering case, we let $n = \max\{n_1, n_2\}$, and recall that (C_k, C'_k) is the k-th bi-cluster for $k \in [r]$. The minimum cluster size is $K = \min_{k \in [r]} \min\left\{|C_k|, |C'_k|\right\}$. These notations are consistent with the clustering setting, for which n is the total number of nodes and K the minimum cluster size.

Our main result is a general theorem that gives sufficient conditions for the programs (2) and (3) to recover the true cluster matrix Y^*. These conditions are stated in terms of the minimum cluster size K, the label distributions μ and ν, as well as any given weight function $w(\cdot)$ through the quantities

$$\mathbb{E}_\mu w := \int_{\mathcal{L}} w(l)\,\mathrm{d}\mu \quad \text{and} \quad \mathrm{Var}_\mu w := \int_{\mathcal{L}} [w(l) - \mathbb{E}_\mu w]^2\,\mathrm{d}\mu;$$

$\mathbb{E}_\nu w$ and $\mathrm{Var}_\nu w$ are defined similarly. We assume in the sequel that all the relevant integrals and expectations are well-defined. With these notations, we now state our general theorem.

Theorem 2 (Main) *Suppose b is any number that satisfies $|w(l)| \le b$ almost everywhere (a.e.) over $\mathcal{L}$ with respect to μ and ν. There exists a universal constant $c > 0$ such that if*

$$-\mathbb{E}_\nu w \ge c \frac{b \log n + \sqrt{K \log n}\sqrt{\mathrm{Var}_\nu w}}{K}, \tag{5}$$

$$\mathbb{E}_\mu w \ge c \frac{b \log n + \sqrt{n \log n}\sqrt{\max(\mathrm{Var}_\mu w, \mathrm{Var}_\nu w)}}{K}, \tag{6}$$

2. The nuclear norm of a matrix is defined as the sum of its singular values. A cluster matrix is positive semidefinite, so its nuclear norm is equal to its trace and also called the trace norm.

3. Both (3) and (4) are strictly feasible, so strong duality holds by Slater's condition.

then Y^* is the unique solution to the programs (2) *and* (3) *with probability at least* $1 - n^{-10}$.[4]

We prove this claim in Section 7. Theorem 2 is in fact a special case of the more general Theorem 9 (Section 3.5), which does not require the boundedness assumption $w(l) \le b$.

Theorem 2 is valid for any given weight function w. Below we discuss how to choose w optimally, and then address the case where w deviates from the optimal choice.

3.1 Optimal Weights

A candidate for a good weight function w can be derived from the maximum likelihood estimator (MLE) of Y^*. Given the observed label matrix L, the log-likelihood of the true cluster matrix taking the value Y is

$$\begin{aligned}\log p\big(L \,|\, Y^* = Y\big) &= \sum_{i,j} \log \left[\mu(L_{ij})^{Y_{ij}} \nu(L_{ij})^{1-Y_{ij}}\right] \\ &= \sum_{i,j} Y_{ij} \underbrace{\log \frac{\mu(L_{ij})}{\nu(L_{ij})}}_{W_{ij}} + \underbrace{\sum_{i,j} \log \nu(L_{ij})}_{c} \\ &= \langle W, Y\rangle + c.\end{aligned}$$

The MLE, which maximizes the above expression over Y, therefore corresponds to using the log likelihood ratio as the weight function:

$$w(l) \leftarrow w^{\text{MLE}}(l) := \log \frac{\mu(l)}{\nu(l)}.$$

Specializing Theorem 2 to the weight function w^{MLE}, we obtain the following theorem that characterizes the performance of using the MLE weights in the convex relaxations. Here $D(\cdot \,\|\, \cdot)$ denotes the KL divergence between two distributions.

Theorem 3 (MLE) *Suppose that $w = w^{MLE}$ is used as the weight function, and b and ζ are any numbers that satisfy with $D(\nu\|\mu) \le \zeta D(\mu\|\nu)$ and $\left|\log \frac{\mu(l)}{\nu(l)}\right| \le b, \forall l \in \mathcal{L}$. There exists a universal constant $c > 0$ such that if*

$$D(\nu\|\mu) \ge c(b+2)\frac{\log n}{K}, \tag{7}$$

$$D(\mu\|\nu) \ge c(\zeta+1)(b+2)\left(\frac{n \log n}{K^2}\right), \tag{8}$$

then with probability at least $1 - n^{-10}$, Y^* is the unique solution to the programs (2) *and* (3)*. Moreover, it always holds that $D(\nu\|\mu) \le (2b+3)D(\mu\|\nu)$, so we can take $\zeta = 2b + 3$.*

We prove this claim in Appendix A. The two conditions (7) and (8) are not symmetric due to the fact that there are more cross-cluster pairs than in-cluster pairs in general. The quantity ζ accounts for the asymmetry between $D(\nu\|\mu)$ and $D(\mu\|\nu)$.

4. In all subsequent results, we use an arbitrary choice of exponent in the failure probability n^{-10}. It is easily seen from Theorem 9 that the constant c scales linearly with the exponent.

Theorem 3 has the intuitive interpretation that the in/cross-cluster label distributions μ and ν should be sufficiently different, measured by their KL divergence, for the underlying clusters to be recovered. Using a classical result in information theory (Topsoe, 2000), we may replace the KL divergences with a quantity called the triangle discrimination that is often easier to work with. This is summarized in the following corollary.

Corollary 4 (MLE 2) *Suppose w^{MLE} is used, and b, ζ are defined as in Theorem 3. There exists a universal constant c such that Y^* is the unique solution to the programs* (2) *and (3) with probability at least $1 - n^{-10}$ if*

$$\int_{\mathcal{L}} \frac{(\mu(l) - \nu(l))^2}{\mu(l) + \nu(l)} \, \mathrm{d}\lambda \ge c(\zeta+1)(b+2)\left(\frac{n \log n}{K^2}\right). \tag{9}$$

One may take $\zeta = 2b + 3$.

We prove this claim in Appendix A. Note that the left hand side of (9) is the *triangle discrimination* between μ and ν, which lower bounds of the KL-divergence (cf. Lemma 23.) It can be seen that the constant c if Corollary 4 may be trivially chosen such that it only differs from the c in Theorem 3 by a factor of 2. In general, we do not assume any special relationship between universal constants.

The MLE weight function w^{MLE} turns out to be near-optimal, at least in the two-cluster case, in the sense that no other weight function (in fact, no other algorithm) has significantly better statistical performance. This is shown by establishing a necessary condition for *any* algorithm to recover the true clustering Y^*. Here, an algorithm is a measurable function $\hat{Y}$ that maps the data L to a clustering represented by a cluster matrix.

Theorem 5 (Converse) *The following holds for some universal constants $c, c' > 0$. Suppose $K = \frac{n}{2}$, and the quantity b defined in Theorem 3 satisfies $b \le c'$. If*

$$\int_{\mathcal{L}} \frac{(\mu(l) - \nu(l))^2}{\mu(l) + \nu(l)} \, \mathrm{d}\lambda \le \frac{c \log n}{n}, \tag{10}$$

then $\inf_{\hat{Y}} \sup_{Y^*} \mathbb{P}\big(\hat{Y}(L) \ne Y^*\big) \ge \frac{1}{2}$, where the supremum is over all possible cluster matrices.

We prove this claim in Appendix B.

Under the assumption of Theorem 5, the conditions (9) and (10) match up to a constant factor, showing that program (3) with the MLE weights w^{MLE} is statistically order-wise optimal.

3.2 Monotonicity

We sometimes do not know the exact label distributions μ and ν in computing the MLE weights w^{MLE}. Instead, we may construct the weights using the log likelihood ratios of some "incorrect" distributions $\bar{\mu}$ and $\bar{\nu}$. Our algorithms have a nice *monotonicity* property: as long as the divergence between the true μ and ν is larger than that between $\bar{\mu}$ and $\bar{\nu}$ (hence an "easier" problem), then the algorithm still has the same (if not better) probability of success, even though the wrong weights are used.

More precisely, we say that the pair (μ, ν) is *more divergent* than $(\bar{\mu}, \bar{\nu})$ if, for each $l \in \mathcal{L}$, we have

$$\text{either} \quad \frac{\mu(l)}{\nu(l)} \ge \frac{\mu(l)}{\bar{\nu}(l)} \ge \frac{\bar{\mu}(l)}{\bar{\nu}(l)} \ge 1 \quad \text{or} \quad \frac{\nu(l)}{\mu(l)} \ge \frac{\nu(l)}{\bar{\mu}(l)} \ge \frac{\bar{\nu}(l)}{\bar{\mu}(l)} \ge 1.$$

Theorem 6 (Monotonicity) *Suppose that we use the weight function* $w(l) = \log \frac{\bar{\mu}(l)}{\bar{\nu}(l)}, \forall l \in \mathcal{L}$, *while the actual label distributions are* μ *and* ν. *If the conditions in Theorem 3 or Corollary 4 hold with* μ, ν *replaced by* $\bar{\mu}, \bar{\nu}$, *and* (μ, ν) *is more divergent than* $(\bar{\mu}, \bar{\nu})$, *then with probability at least* $1 - n^{-10}$, Y^* *is the unique solution to programs* (2) *and* (3).

We prove this claim in Appendix C.

Theorem 6 suggests that one may choose the weight function by using the log likelihood ratios of a *conservative* estimate (i.e., a less divergent one) of the true label distribution pair.

3.3 Using Inaccurate Weights

We now consider a more general way of choosing the weight function w, which need not be conservative, but is only required to be not too far from the true log-likelihood ratios w^{MLE}. Let

$$\varepsilon(l) := w(l) - w^{\text{MLE}}(l) = w(l) - \log \frac{\mu(l)}{\nu(l)}$$

be the weighting error for each label $l \in \mathcal{L}$. Then the quantities $\Delta_\mu := \int_{\mathcal{L}} \varepsilon(l)\, \mathrm{d}\mu$ and $\Delta_\nu := \int_{\mathcal{L}} \varepsilon(l)\, \mathrm{d}\nu$ represent the average errors with respect to μ and ν. Note that Δ_μ and Δ_ν can be positive or negative. The theorem below characterizes the performance of using such an inaccurate weight function w.

Theorem 7 (Inaccurate Weights) *Let* b *and* ζ *be defined as in Theorem 3. Suppose that the weight function* w *satisfies*

$$|w(l)| \le \alpha \left| \log \frac{\mu(l)}{\nu(l)} \right|, \ \forall l \in \mathcal{L}, \quad |\Delta_\mu| \le \gamma D(\mu\|\nu), \quad \textit{and} \quad |\Delta_\nu| \le \gamma D(\nu\|\mu)$$

for some numbers $\alpha > 0$ *and* $\gamma < 1$. *Then* Y^* *is unique solution to the programs* (2) *and* (3) *with probability at least* $1 - n^{-10}$ *provided that*

$$D(\nu\|\mu) \ge c \frac{\alpha^2}{(1-\gamma)^2}(b+2)\frac{\log n}{K} \quad \textit{and} \quad D(\mu\|\nu) \ge c \frac{\alpha^2}{(1-\gamma)^2}(\zeta+1)(b+2)\left(\frac{n \log n}{K^2}\right).$$

We prove this claim in Appendix D.

Therefore, as long as the errors Δ_μ and Δ_ν in w are not too large, the condition for recovery is order-wise similar to that of the MLE weight given in Theorem 3. The numbers α and γ measure the level of inaccuracy in the weight function w with respect to the ideal choice w^{MLE}. The last two conditions in Theorem 7 thus quantify the relation between the inaccuracy in w and the statistical price to pay for using such a weight.

3.4 Linear Weights

We next consider a weight function that is an alternative to the MLE weights, and may be preferable in certain scenarios. It is based on a linear approximation to the MLE weight function, hence referred to as the *linear weights*:

$$w^{\text{LIN}}(l) := \frac{\mu(l) - \nu(l)}{\mu(l) + \nu(l)}.$$

It is straightforward to show that

$$\mathbb{E}_\mu w^{\text{LIN}} = -\mathbb{E}_\nu w^{\text{LIN}} = \frac{\text{Var}_\mu w^{\text{LIN}} + \text{Var}_\nu w^{\text{LIN}}}{2} = \frac{1}{2}\int_{\mathcal{L}} \frac{(\mu(l) - \nu(l))^2}{\mu(l) + \nu(l)}\, \mathrm{d}\lambda.$$

Using this observation, we immediately obtain the following corollary for w^{LIN} from Theorem 2.

Corollary 8 (Linear) *Suppose the weight function* w^{LIN} *is used. There exists a universal constant* c *such that* Y^* *is the unique solution to the program* (3) *with probability at least* $1 - n^{-10}$ *if*

$$\int_{l \in \mathcal{L}} \frac{(\mu(l) - \nu(l))^2}{\mu(l) + \nu(l)}\, \mathrm{d}\lambda \ge c \frac{n \log n}{K^2}. \tag{11}$$

Note the the left hand side of (11) coincides with the triangle discrimination term that has appeared in Corollary 4 as a lower bound of the KL-divergence. Comparing Corollary 8 above to the converse results in Theorem 5, we see that, at least for the case of two equal-size clusters, the linear weight function w^{LIN} provides order-wise optimal recovery guarantees.

The linear weights $w^{\text{LIN}}(l)$ are always between -1 and 1 and thus well bounded. Empirically, it is observed that w^{LIN} performs slightly worse than the MLE weight function w^{MLE} in general. However, in certain cases w^{LIN} can actually outperform w^{MLE}. In particular, comparing Corollaries 4 and 8 suggests that this most likely happens when the quantity ζ is large, i.e., $D(\nu\|\mu) \gg D(\mu\|\nu)$. We demonstrate this phenomenon in our empirical results in Section 6.2, where ζ is large in the case of dense graphs.

3.5 Unbounded Weights

The general result in Theorem 2 is valid for any weight function that is uniformly bounded, i.e., $|w(l)| \le b$ a.e. on $\mathcal{L}$. (The theorem becomes vacuous if $b \to \infty$.) We now give a more general result, which allows some of the weights to have arbitrarily large magnitudes. Unbounded weights arise when some of the pairwise observations are highly certain, or given as hard constraints. For example, the label between nodes i and j may have the form l_{same} = "these nodes are known to be in the same cluster", in which case assigning an unbounded weight $w(l_{\text{same}}) \to \infty$ in the program (1) forces the nodes i and j to be clustered together. Similarly, for two nodes that are known to be in different clusters, a large negative weight would be desirable.

We identify the set of labels associated with unbounded weights as

$$\mathcal{L}_\infty := \{l \in \mathcal{L} : |w(l)| > b_\infty\},$$

where b_∞ should be thought of as a very large positive number. For all $l \notin \mathcal{L}_\infty$, we assume that $|w(l)| \le b$ for some $b < b_\infty$. We further assume that the weight function w is $\mathcal{L}_\infty$-*consistent* in the sense that

$$\mu(l) = 0, \ \ \forall l \in \mathcal{L}_\infty : w(l) < 0 \qquad \text{and} \qquad \nu(l) = 0, \ \ \forall l \in \mathcal{L}_\infty : w(l) > 0.$$

This condition ensures that whenever a node pair is assigned with unbounded weight, the sign of the weight is consistent with the true cluster relation between the pair, that is, with probability one,

$$W_{ij}(2Y^*_{ij} - 1) > 0, \quad \forall (i,j) : L_{ij} \in \mathcal{L}_\infty. \tag{12}$$

Denote by $s_\mu := \int_{\mathcal{L}_\infty} \mathrm{d}\mu$ the total probability of the labels in $\mathcal{L}_\infty$ under the distribution μ, and

$$\tilde{w}_b(l) := \begin{cases} 0 & \text{if } l \in \mathcal{L}_\infty, \\ w(l) & \text{otherwise,} \end{cases}$$

the weight function restricted to labels with bounded weights. Finally, in the bi-clustering setting, we need an additional parameter ξ that measures bi-cluster skewness:

$$\xi := \max_{k\in[r]} \left\{ \sqrt{\frac{|C_k|}{|C'_k|}}, \sqrt{\frac{|C'_k|}{|C_k|}} \right\}.$$

In the standard clustering case we simply have $\xi = 1$.

Under the above setting, we have the following result valid for unbounded weights with arbitrarily large b_∞.

Theorem 9 *Suppose that the weight function w is $\mathcal{L}_\infty$-consistent, and $\beta > 0$ is any fixed number. There exists a universal constant $c > 0$ such that the following is true. If*

$$\mathbb{E}_\mu \tilde{w}_b \geq 0, \tag{13}$$

$$-\mathbb{E}_\nu \tilde{w}_b > c\,\frac{b\beta\log n + \sqrt{K\beta\log n}\sqrt{\mathrm{Var}_\nu \tilde{w}_b}}{K}, \tag{14}$$

and at least one of the following two inequalities holds:

$$\mathbb{E}_\mu \tilde{w}_b > c\,\frac{b\beta\log n + \sqrt{n\beta\log n}\sqrt{\max(\mathrm{Var}_\mu \tilde{w}_b, \mathrm{Var}_\nu \tilde{w}_b)}}{K}, \tag{15}$$

$$s_\mu + \frac{\mathbb{E}_\mu \tilde{w}_b}{b_\infty} > c\,\max\left\{ \max\left\{1, \frac{b\xi}{b_\infty}\right\}\frac{\beta\log n}{K}, \frac{n\max(\mathrm{Var}_\mu \tilde{w}_b, \mathrm{Var}_\nu \tilde{w}_b)}{Kb_\infty^2} \right\}, \tag{16}$$

then with probability at least $1 - n^{-\beta}$ the solution to the programs (2) *and* (3) *is unique and equals Y^*.*

We prove this claim in Section 7.

Theorem 9 is particularly useful when we take $b_\infty \to \infty$ while keeping b, n and K finite. In this case the condition (16) simplifies to

$$s_\mu > c\frac{\beta\log n}{K}, \tag{17}$$

a lower bound on the probability of observing a label from $\mathcal{L}_\infty$ between two nodes in the same cluster. The conditions (17) together with (14) are sufficient for successful recovery of Y^*. Significantly, these conditions only depend *logarithmically* on n, which is in contrast to the dependence on $\sqrt{n}$ in Theorem 2. This improvement demonstrates the benefit of assigning unbounded weights to labels in $\mathcal{L}_\infty$ (i.e., label with high certainty).

Also note that Theorem 2 is a special case of Theorem 9, as we show in Section 7.

4. Consequences and Applications

We now apply the general results in the last section to different concrete cases. In sections 4.1 and 4.2, we consider two clustering settings with sub-Gaussian weights or non-uniform edge probabilities, both of which generalize the standard stochastic block model. In these settings two immediate corollaries of our main theorems recover, and in fact improve upon, existing results. In sections 4.3 and 4.4, we turn to the more complicated setting of clustering time-varying graphs and derive several new results.

Throughout this section we use c, c_0 etc. to denote universal positive constants.

4.1 Clustering a Sub-Gaussian Matrix with Partial Observations

Analogous to the planted partition and stochastic block models for unweighted graphs, the *submatrix localization* problem concerns clustering a weighted graph whose adjacency matrix has sub-Gaussian entries. This problem is often used as an idealized model for the bi-clustering of drugs and proteins, species and gene sequences, customers and products, and other forms of object-feature pairs (see Kolar et al., 2011, and references therein).

Here we consider a generalization of this problem, where some of the entries are unobserved. This setting arises when the relations between some node pairs are costly to determine, so it is unknown whether or not they are connected by an edge, nor is the weight of the edge (Shamir and Tishby, 2011). For simplicity, we state our results assuming that the adjacency matrix and its submatrices are symmetric, and that the entries of the matrix are Gaussian. Our theoretical results apply to the general sub-Gaussian case without change.

Specifically, we observe a matrix $L \in (\mathbb{R}\cup\{?\})^{n\times n}$, where $L_{ij} =?$ with probability $1-s$, and otherwise L_{ij} follows the Gaussian distribution $\mathcal{N}(u_{ij}, 1)$.[5] Here $L_{ij} =?$ denotes that the relation between the i-th and j-th nodes is unobserved. The means $\{u_{ij}\}$ of the Gaussians satisfy the following: within L there are r submatrices of size at least $K\times K$ with disjoint row and column support; we have $u_{ij} = \bar{u}$ if the pair (i,j) is inside one of the submatrices, and $u_{ij} = \underline{u}$ if outside, where $\bar{u} > \underline{u} \geq 0$. The goal is to locate these submatrices with elevated means given the large matrix L. We may think of this problem as finding clusters in a weighted graph with the partially observed adjacency matrix L.

This problem is a special case of our general framework with a mixed-valued label set $\mathcal{L} = \mathbb{R}\cup\{?\}$. The base measure λ is chosen to be such that it equals the Lebesgue measure on $\mathbb{R}$ and 1 on $\{?\}$. Then the density functions of μ and ν are

$$\mu(l) = \begin{cases} 1-s, & \text{if } l =?, \\ s\cdot\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(l-\bar{u})^2}{2}\right), & \text{if } l\in\mathbb{R}, \end{cases} \quad \text{and} \quad \nu(l) = \begin{cases} 1-s, & \text{if } l =?, \\ s\cdot\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{(l-\underline{u})^2}{2}\right), & \text{if } l\in\mathbb{R}. \end{cases}$$

The MLE weight function therefore has the form

$$w^{\mathrm{MLE}}(l) \propto \begin{cases} 0, & \text{if } l =?, \\ l - (\bar{u}+\underline{u})/2, & \text{if } l\in\mathbb{R}. \end{cases}$$

5. Equivalently, L_{ij} is generated according to $\mathcal{N}(u_{ij}, 1)$, and then is replaced by ? with probability $1-s$.

This submatrix localization problem is most interesting when $\bar{u} - \underline{u} \leq c_0\sqrt{\log n}$ for some universal constant $c_0 > 0$,[6] which we assume to hold in the sequel. Observe that $D(\mu\|\nu) = D(\nu\|\mu) = \frac{1}{4}s(\bar{u} - \underline{u})^2$. In Appendix E we apply the general result in Theorem 2 to derive the following recovery guarantee.

Corollary 10 (Gaussian Graphs) *Under the above setting, if*

$$s\,(\bar{u} - \underline{u})^2 \geq c\frac{n\log^3 n}{K^2}, \tag{18}$$

then with probability at least $1 - 2n^{-10}$, Y^* *is the unique solution to the programs* (2) *and* (3) *with the weight function* $w = w^{MLE}$.

In the full observation case $s = 1$, Corollary 10 recovers the results in Ames (2014); Balakrishnan et al. (2011); Kolar et al. (2011) up to log factors. Our results are more general as we allow for partial observation, which is not considered in previous work. Corollary 10 allows the observation probability to be as small as $s = \Omega(\frac{n}{K^2})$, or the signal strength to be $\bar{u} - \underline{u} = \Omega(\frac{\sqrt{n}}{K})$ (ignoring log factors). In general we see a quadratic tradeoff between these two quantities: with four times more observations, the signal strength can be 50% smaller.

4.2 Planted Partition with Non-Uniform Edge Probabilities

Recall that in the standard planted partition model, the observed unweighted graph is generated by independently connecting each node pair with probability p if they are in the same cluster, or with probability q if they are in different clusters, where $p, q \in [0, 1]$ are two fixed numbers. Here we consider a more general setting, where the in/cross-cluster edge probabilities can be different across node pairs. This setting has a range of applications as discussed at the end of this sub-section.

Concretely, suppose that each node pair (i, j) is associated with two numbers P_{ij} and Q_{ij}. Independently for each (i, j), the values of P_{ij} and Q_{ij} are generated randomly according to some distribution ψ on $[0, 1] \times [0, 1]$, which is assumed to have a density function with respect to some reference measure λ_0. Conditioned on P_{ij} and Q_{ij}, the nodes i and j are connected by an edge with probability P_{ij} if they are in the same cluster, or with probability Q_{ij} if they are in different clusters. For each (i, j), one knows the values of P_{ij} and Q_{ij}, but not which of them is the probability that generates the edge. Assuming that there are r ground-truth clusters of size at least K, the goal is to find these clusters given the graph adjacency matrix $A \in \{0, 1\}^{n\times n}$ and the edge probabilities (P_{ij}) and $(Q_{ij}) \in [0, 1]^{n\times n}$.

This model can be cast as a special case of our labeled framework, in which each pairwise label is a triplet of the form $L_{ij} = (A_{ij}, P_{ij}, Q_{ij}) \in \mathcal{L} = \{0, 1\} \times [0, 1] \times [0, 1]$. Note that the labels take a combination of discrete and real values. If we choose the base measure λ to be the product of the counting measure on $\{0, 1\}$ and λ_0, then the density functions of μ and ν are given as follows: for each $l = (a, p, q) \in \{0, 1\} \times [0, 1] \times [0, 1]$,

$$\mu(l) = \begin{cases} p\psi(p, q), & \text{if } a = 1, \\ (1 - p)\psi(p, q), & \text{if } a = 0, \end{cases} \quad \text{and} \quad \nu(l) = \begin{cases} q\psi(p, q), & \text{if } a = 1, \\ (1 - q)\psi(p, q), & \text{if } a = 0. \end{cases}$$

6. Otherwise, if $\bar{u} - \underline{u} > c_0\sqrt{\log n}$ for some sufficiently large constant $c_0 > 0$, then with high probability the smallest entry inside the submatrices will dominate the largest entry outside, so simple element-wise thresholding finds the submatrices (Balakrishnan et al., 2011; Kolar et al., 2011).

The MLE weight function therefore has the form

$$w^{\text{MLE}}(l) = w^{\text{MLE}}\big((a, p, q)\big) = a\log\frac{p}{q} + (1 - a)\log\frac{1 - p}{1 - q}. \tag{19}$$

Applying Corollary 4 for MLE weights, we immediately obtain the following recovery guarantee:

Corollary 11 (Non-uniform Edge Probabilities, I) *Under the above setting, suppose that there exist* ζ *and* b *such that* $D(\nu\|\mu) \leq \zeta D(\mu\|\nu)$ *and* $|\log\frac{\mu(l)}{\nu(l)}| \leq b$ *almost everywhere. If*

$$\mathbb{E}_{(P,Q)\sim\psi}\left[\frac{(P - Q)^2}{(P + Q)(2 - P - Q)}\right] \geq c\,(\zeta + 1)(b + 2)\frac{n\log n}{K^2}.$$

then with probability at least $1 - n^{-10}$, Y^* *is the unique solution to the programs* (2) *and* (3) *with the MLE weight function in* (19). *One may take* $\zeta = 2b + 3$.

Here the notation $(P, Q) \sim \psi$ means that (P, Q) is a pair of numbers sampled from the distribution ψ.

We can simplify the above condition considerably by considering a conservative weight function (cf. Section 3.2): on the RHS of (19) we replace p and q with $\bar{p} = \frac{3}{4}p + \frac{1}{4}q$ and $\bar{q} = \frac{1}{4}p + \frac{3}{4}q$, respectively, which leads to the weight function

$$\bar{w}^{\text{MLE}}(l) = \bar{w}^{\text{MLE}}\big((a, p, q)\big) = a\log\frac{3p + q}{p + 3q} + (1 - a)\log\frac{4 - 3p - q}{4 - p - 3q}. \tag{20}$$

These weights are clearly bounded by a constant. Applying Theorem 6 for monotonicity together with Corollary 4 for MLE weights, we obtain the following recovery guarantee:

Corollary 12 (Non-uniform Edge Probabilities, II) *Under the above setting, suppose that* $P \geq Q$ *almost surely with respect to* ψ. *If*

$$\mathbb{E}_{(P,Q)\sim\psi}\left[\frac{(P - Q)^2}{P(1 - Q)}\right] \geq c\,\frac{n\log n}{K^2},$$

then with probability at least $1 - n^{-10}$, Y^* *is the unique solution to the programs* (2) *and* (3) *with the conservative MLE weight function* $\bar{w}^{MLE}$ *in* (20).

As a passing observation, we note the $\log n$ factor in the above two corollaries can be removed by a slightly more careful analysis; see Remark 20.

Below we discuss two applications of Corollary 12, which leads to generalization and improvement over existing results.

4.2.1 CLUSTERING PARTIALLY OBSERVED UNWEIGHTED GRAPHS

In Section 4.1 we discussed clustering Gaussian weighted graphs with partial observations. Here we consider such a similar setting for unweighted graphs under the planted partition model.[7]

7. The planted partition model satisfies the sub-Gaussian assumption, so the results in Section 4.1 in fact apply to this setting. Here we take into account the variance information to derive tighter bounds.

Specifically, suppose that the edge probabilities $(P_{ij}, Q_{ij}) \sim \psi$ is distributed as $(P_{ij}, Q_{ij}) = (p, q)$ with some probability s, and $P_{ij} = Q_{ij} = \frac{1}{2}$ with probability $1 - s$, where $p > q$ are two fixed numbers. This setting extends the standard planted partition model to partial observation: with probability $1 - s$, the connection A_{ij} between a pair (i, j) is known to be purely random and uninformative about the cluster membership of i and j—having such a purely random observation is equivalent to saying that one does not observe whether or not i and j are connected by an edge.[8] This setting is sometimes called the planted partition model with partial or *censored* edge observation. Note that the conservative MLE weight function (20) assigns zero weight to unobserved/uninformative pairs, as should be expected.

By calculating the left hand side of the condition in Corollary 12, we immediately obtain the following guarantee.

Corollary 13 *Under the above planted partition model with partial observation, if*

$$\frac{s(p-q)^2}{p(1-q)} \geq c\,\frac{n \log n}{K^2},$$

then with probability at least $1 - n^{-10}$*,* Y^* *is the unique solution to the programs* (2) *and* (3) *with the conservative MLE weight function* $\bar{w}^{MLE}$ *in* (20)*.*

In the full observation setting $s = 1$, the above result matches the best existing bounds for standard planted partition (e.g., in Chen et al., 2014c; Anandkumar et al., 2014), up to a $\log n$ factor that can be removed (see the remark after Corollary 12). In the partial observation setting $s < 1$, the work by Vinayak et al. (2014); Chen et al. (2014a) gives a similar bound under the additional assumption $p > 0.5 > q$, which is not required by Corollary 13. For general p and q, the best existing bounds can be found in Oymak and Hassibi (2011); Chen et al. (2014c), where unobserved entries in the adjacency matrix A are replaced with 0 and recovery is guaranteed under the condition $\frac{s(p-q)^2}{p(1-sq)} \gtrsim \frac{n \log n}{K^2}$.[9] Our result is tighter when p and q are close to 1. Finally, we note that our result is non-asymptotic and valid under any scaling of the parameters n, K, r, p, q, s.

4.2.2 PLANTED PARTITION WITH NON-UNIFORM UNCERTAINTY

We next consider an application of Corollary 12 to a further generalization of the partial observation setting above. In the above setting, if an entry A_{ij} of the adjacency matrix is unobserved or purely random, the cluster relation between the nodes i and j is completely uncertain (based on only A_{ij}). Here we assume that each A_{ij} is associated with a different, continuous level of uncertainty. For some node pairs, the observation A_{ij} of the existence of an edge (or respectively, the absence of an edge) may be generated by accurate measurements, and therefore a strong indicator that the nodes i and j should be assigned to the same (or respectively, different) clusters.

For example, in *crowd-clustering* a number of users are asked whether or not they think a pair of nodes (e.g., movies or images) are similar, and the final graph is constructed by aggregating the users' answers by say majority voting (Gomes et al., 2011; Yi et al., 2012). The uncertainty levels are naturally non-uniform across pairs: a pair receiving a large number of unanimous votes

8. This model can be rephrased in the following equivalent form: given a partially observed adjacency matrix $A \in \{0, 1, ?\}^{n \times n}$, one replaces each unobserved entry ? in A with an independent Bernoulli random variable.
9. We use $\gtrsim$ when the inequality is up to a constant factor.

is associated with high confidence, whereas those with a few votes or divergent votes have low confidence, and a pair receiving no votes is completely uncertain.

In such a setting, each pairwise observation A_{ij} should be treated differently according to its level of uncertainty. Often, one has prior knowledge on the uncertainty levels, for example when the graph is built from a known process as in the crowd-clustering example. Intuitively, using such knowledge improves clustering performance. It is, however, less clear how this knowledge can be used and how much improvement it can provide. Below we use Corollary 12 to obtain a quantitative answer.

For simplicity, we focus on a special case of the setting in Corollary 12: we assume that the distribution ψ of edge probabilities is symmetric, in the sense that we always have $Q_{ij} \equiv 1 - P_{ij}$, where $Q_{ij} \in [0, \frac{1}{2}]$.[10] In this case, the quantity Q_{ij} can be interpreted as the probability of an erroneous observation—namely, a no-edge between two nodes in the same cluster (a false negative), or an edge between two nodes in different clusters (a false positive). Therefore, Q_{ij} measures the level of uncertainty in the observation A_{ij} at (i, j). In particular, $Q_{ij} = 0$ means that A_{ij} is a noiseless observation of the cluster relation between i and j, whereas $Q_{ij} = \frac{1}{2}$ corresponds to a completely uncertain observation with no information.

For each observation $A_{ij} \in \{0, 1\}$ with an uncertainty level $Q_{ij} \in [0, \frac{1}{2}]$, the conservative MLE weight function (20) assigns the weight

$$W_{ij} = (2A_{ij} - 1) \log \frac{3 - 2Q_{ij}}{1 + 2Q_{ij}}.$$

Note that the magnitude $|W_{ij}|$ of the weight is small for a high uncertainty level Q_{ij}; in particular, one has $W_{ij} = 0$ for a completely uncertain observation with $Q_{ij} = \frac{1}{2}$.

Using these weight, Corollary 12 guarantees that the programs (2) and (3) recover the clusters with high probability provided that

$$\mathbb{E}\left[\left(\frac{1}{2} - Q\right)^2\right] \gtrsim \frac{n \log n}{K^2},$$

where the expectation is with respect to $(P, Q) \sim \psi$. We can compare this result with the presumably sub-optimal unweighted approach, which ignores the non-uniformity of the uncertainty level and uses uniform weights $W_{ij} = 2A_{ij} - 1 \in \{-1, 1\}$. By Theorem 2 this unweighted approach succeeds if

$$\left(\frac{1}{2} - \mathbb{E}Q\right)^2 \gtrsim \frac{n \log n}{K^2}.$$

The difference between the left hand sides of the last two conditions is $\mathbb{E}[Q^2] - (\mathbb{E}Q)^2 = \mathrm{Var}[Q]$, the variance of the uncertainty level.[11] This is therefore evidence that the weighted approach is indeed better than the unweighted one, and the gain is large precisely when the uncertainty level is very non-uniform with a high variance. We refer the readers to Chen et al. (2014b) for empirical verification of this gain as well as further discussion and applications.

Of course our results are not limited to the symmetric setting $Q_{ij} \equiv 1 - P_{ij}$. Corollary 12 is applicable under a general distribution ψ of P_{ij} and Q_{ij}, in which case a larger value of $|P_{ij} - Q_{ij}|$ corresponds to a lower level of uncertainty. We omit discussion of this general setting due to space limit.

10. That is, the density function $\psi(p, q)$ is supported on the line segment $\{(p, q) \in \mathbb{R}^2 : q = 1 - p, 0 \leq q \leq \frac{1}{2}\}$.
11. Asymptotically the difference between the right hand sides can be made arbitrarily small, regardless of the contants. Furthermore, empirically the constants involved are very similar, depending on how the weights are bounded.

4.3 Clustering Time-varying Multiple-snapshot Graphs

Standard graph clustering concerns the partition of a single, static graph. We now consider a setting where the graph is time-varying. Specifically, for each time interval $t = 1, 2, \ldots, T$, one observes a snapshot of the (label) graph $L^{(t)} \in \mathcal{L}^{n\times n}$. In this subsection, we assume that each snapshot is generated by the distributions μ and ν independently of other snapshots. In the next subsection we consider the more general setting of Markov snapshots.

We can map this problem into our labeled framework, by considering the whole time sequence of $\bar{L}_{ij} := (L_{ij}^{(1)}, \ldots, L_{ij}^{(T)})$ as a single label observed at the pair (i, j). In this case the label set is the set of all possible sequences, i.e., $\bar{\mathcal{L}} = \mathcal{L}^T$, and the label distributions are (with a slight abuse of notation) given by the products $\mu(\bar{L}_{ij}) = \prod_{t=1}^{T} \mu(L_{ij}^{(t)})$ and $\nu(\bar{L}_{ij}) = \prod_{t=1}^{T} \nu(L_{ij}^{(t)})$. The MLE weight (normalized by T) is therefore the average log-likelihood ratio:

$$w^{\text{MLE}}(\bar{L}_{ij}) = \frac{1}{T} \log \frac{\mu(\bar{L}_{ij})}{\nu(\bar{L}_{ij})} = \frac{1}{T} \sum_{t=1}^{T} \log \frac{\mu(L_{ij}^{(t)})}{\nu(L_{ij}^{(t)})}.$$

Since the weight $w^{\text{MLE}}(\bar{L}_{ij})$ is the average of T independent random variables, its variance scales as $\frac{1}{T}$. Applying Theorem 2 and following almost identical arguments as in the proof of Theorem 3, we obtain the following guarantee for clustering a time-varying graph with independent snapshots.

Corollary 14 (Independent Snapshots) *Suppose that* $|\log \frac{\mu(l)}{\nu(l)}| \le b, \forall l \in \mathcal{L}$ *and* $D(\nu\|\mu) \le \zeta D(\mu\|\nu)$. *If*

$$D(\nu\|\mu) \ge c(b+2)\frac{\log n}{K} \qquad \text{and} \tag{21}$$

$$D(\mu\|\nu) \ge c(b+2)\max\Big\{\frac{\log n}{K}, (\zeta+1)\frac{n\log n}{TK^2}\Big\}, \tag{22}$$

then with probability at least $1 - n^{-10}$, Y^* *is the unique solution to the programs* (2) *and* (3) *with the MLE weights given above.*

Note that setting $T = 1$ above recovers the result in Theorem 3 for clustering a single label graph.

The second term on the right hand side of (22) usually dominates. In this case, Corollary 14 says that the clustering problem becomes easier if T is larger (i.e., more snapshots of the graph are observed), and the relation between T and cluster recovery is quantified precisely. As a concrete example, suppose that each snapshot is generated by the standard planted partition model with edge probabilities p and q, where $q = p/2$. By Corollary 14 we need $p \gtrsim \frac{n\log n}{TK^2}$ to guarantee cluster recovery. Therefore, if four times more snapshots are available, then one can recover clusters whose sizes K are 50% smaller. Similarly, we see a tradeoff between the number of snapshots T and the graph sparsity p.

4.4 Markov Sequence of Snapshots

We now consider a more general and useful setting for time-varying graphs, where the graph snapshots are not assumed to be independent but instead form a Markov chain.

For simplicity we assume that the Markov chain is time-invariant and has a finite state space and a unique stationary distribution that is also the initial distribution. Therefore, the observations

$L_{ij}^{(t)}, t = 1, 2, \ldots$ at each pair (i, j) are generated by first drawing a label $L_{ij}^{(1)}$ from the stationary distribution μ_0 (or ν_0, according to the cluster membership of i and j) over the finite set $\mathcal{L}$ at $t = 1$, then applying a one-step transition to obtain the label at each subsequent t. In particular, given the previously observed label l, let the intra-cluster and inter-cluster conditional distributions of the next observation be $\mu(\cdot|l)$ and $\nu(\cdot|l)$. We assume that the Markov chains $\{L_{ij}^{(1)}, L_{ij}^{(2)}, \ldots\}$ with respect to both μ and ν are geometrically ergodic, in the sense that for each integer $\tau \ge 1$ and pair $L_{ij}^{(1)}, L_{ij}^{(\tau+1)}$,

$$\Big|\Pr_\mu\big(L_{ij}^{(\tau+1)}|L_{ij}^{(1)}\big) - \mu_0\big(L_{ij}^{(\tau+1)}\big)\Big| \le \kappa\phi^\tau \quad \text{and} \quad \Big|\Pr_\nu\big(L_{ij}^{(\tau+1)}|L_{ij}^{(1)}\big) - \nu_0\big(L_{ij}^{(\tau+1)}\big)\Big| \le \kappa\phi^\tau \tag{23}$$

for some constants $\kappa \ge 1$ and $0 < \phi < 1$ that only depend on μ and ν. Let $D_l(\mu\|\nu)$ denote the KL-divergence between $\mu(\cdot|l)$ and $\nu(\cdot|l)$; $D_l(\nu\|\mu)$ is similarly defined. Let

$$\mathbb{E}_{\mu_0} D_l(\mu\|\nu) := \textstyle\sum_{l\in\mathcal{L}} \mu_0(l) D_l(\mu\|\nu)$$

and similarly for $\mathbb{E}_{\nu_0} D_l(\nu\|\mu)$. As in the previous subsection, we use the average log-likelihood ratio $w^{\text{MLE}}(\bar{L}_{ij}) = \frac{1}{T}\log\frac{\mu(\bar{L}_{ij})}{\nu(\bar{L}_{ij})}$ as the weight, where $\mu(\bar{L}_{ij})$ is the joint distribution of the sequence $\bar{L}_{ij} = \big(L_{ij}^{(1)}, \ldots, L_{ij}^{(T)}\big)$ under the above Markov chain; similarly for $\nu(\bar{L}_{ij})$. Finally, define the quantity

$$\Phi := \frac{\kappa}{(1-\phi)\min_{l\in\mathcal{L}}\{\mu_0(l), \nu_0(l)\}}.$$

With these notations, we have the following corollary of Theorem 2.

Corollary 15 (Markov Snapshots) *Under the above setting, suppose that for each label pair* $(l, l') \in \mathcal{L}\times\mathcal{L}$, *we have* $\Big|\log\frac{\mu_0(l)}{\nu_0(l)}\Big| \le b$, $\Big|\log\frac{\mu(l'|l)}{\nu(l'|l)}\Big| \le b$, $D(\nu_0\|\mu_0) \le \zeta D(\mu_0\|\nu_0)$ *and* $\mathbb{E}_{\nu_0} D_l(\nu\|\mu) \le \zeta\mathbb{E}_{\mu_0} D_l(\mu\|\nu)$ *for some numbers* b *and* ζ. *If*

$$\frac{1}{T}D(\nu_0\|\mu_0) + \Big(1-\frac{1}{T}\Big)\mathbb{E}_{\nu_0} D_l(\nu\|\mu) \ge c(b+2)\frac{\log n}{K}, \tag{24}$$

$$\frac{1}{T}D(\mu_0\|\nu_0) + \Big(1-\frac{1}{T}\Big)\mathbb{E}_{\mu_0} D_l(\mu\|\nu) \ge c(b+2)\max\Big\{\frac{\log n}{K}, (\zeta+1)\Phi\frac{n\log n}{TK^2}\Big\}, \tag{25}$$

then with probability at least $1 - n^{-10}$, Y^* *is the unique solution to the programs* (2) *and* (3) *with MLE weight function* w^{MLE}.

See Appendices F for the proof of this claim, and G for additional discussion of the assumptions.

As an illuminating example, consider the case where $\mu_0 \approx \nu_0$, i.e., the marginal distributions for individual snapshots are very close or even identical in and across clusters. This means that the information about cluster membership is not contained in the labels themselves in individual snapshots, but instead in the *change* of labels between snapshots. This point is made evident in the left hand sides of (24) and (25), as first terms therein are approximately zero. In this case, *it is necessary to use the temporal information* in order to perform clustering. Such information would be lost if one disregards the ordering of the snapshots (for example, by averaging the snapshots) and then applies a single-snapshot clustering algorithm. This example therefore highlights a crucial difference between clustering time-varying graphs and static graphs.

5. Implementation

The convex programs (2) and (4) used in our clustering approach are semidefinite programs and can be solved efficiently by the Alternating Direction Method of Multipliers (ADMM) (Boyd et al., 2011). We provide the pseudocode for a complete implementation of the programs (2) and (4) in Algorithms 1 and 2 below.

Algorithm 1 ADMM solver for Program (2)

Input: Weight matrix $W \in \mathbb{R}^{n\times n}$ symmetric, convergence threshold $\epsilon > 0$
Output: Y

1. $\rho \leftarrow 1, k \leftarrow 0$
2. $Y^k \leftarrow 0, Q^k \leftarrow 0, (Y^k, Q^k \in \mathbb{R}^{n\times n})$
3. $X^{k+1} \leftarrow U \max\{\Lambda, 0\}U^\top$, where $U\Lambda U^\top$ is an eigen-decomposition of $(Y^k - Q^k + \frac{1}{\rho}W)$.
4. $Y^{k+1} \leftarrow \min\left\{\max\left\{X^{k+1} + Q^k, 0\right\}, 1\right\}$
5. $Q^{k+1} \leftarrow Q^k + X^{k+1} - Y^{k+1}$
6. If $\|X^{k+1} - Y^{k+1}\|_F \le \epsilon \max\{\|X^{k+1}\|_F, \|Y^{k+1}\|_F\}$ and $\|Y^{k+1} - Y^k\|_F \le \epsilon\|Q^{k+1}\|_F$ then stop and output $Y = Y^{k+1}$.
7. (Optional) Update ρ and Q^{k+1}
8. $k \leftarrow k + 1$, go to step 3.

The inputs and outputs of Algorithms 1 and 2 are the same terms used in the programs (2) and (4), respectively. We find that in practice, using the tuning parameter $\eta = \sqrt{2n}$ for the program (4) works well. The criterion for convergence is specified by the threshold $\epsilon > 0$, and using $\epsilon = 10^{-4}$ provides a good tradeoff between the convergence time and the quality of the solution. All our experiment results in Section 6 are based on these choices of η and ϵ.

In both Algorithms 1 and 2, an optional Step 7 for updating ρ can be used to potentially improve the speed of convergence. Boyd et al. (2011) suggest one such updating rule, which takes an additional parameter τ and aims to balance the primal and dual residuals. In particular, if $\|X^{k+1} - Y^{k+1}\|_F > \tau\rho\|Y^{k+1} - Y^k\|_F$, then set $\rho \leftarrow 2\rho$ and $Q^{k+1} \leftarrow Q^{k+1}/2$. On the other hand, if $\tau\|X^{k+1} - Y^{k+1}\|_F < \rho\|Y^{k+1} - Y^k\|_F$, then set $\rho \leftarrow \rho/2$ and $Q^{k+1} \leftarrow 2Q^{k+1}$. Typically $\tau = 10$ is a stable choice, which we use in all our experiments. For further details of this updating rule we refer the reader to Boyd et al. (2011).

In practice, due to finite precision and numerical errors, the output matrix Y will not in general have entries that are exactly 0 or 1, even if the true cluster matrix Y^* is in fact the unique optimal solution. If this is the case, a simple rounding of Y will give the correct solution, from which the clusters can then be obtained by sorting the rows. If Y can not be rounded into a cluster matrix, we use a simple k-means algorithm to the rows of Y to extract a desired number of clusters.

Algorithm 2 ADMM solver for Program (4)

Input: Weight matrix $W \in \mathbb{R}^{n_1\times n_2}$, tuning parameter $\eta > 0$, convergence threshold $\epsilon > 0$
Output: Y

1. $\rho \leftarrow 1, k \leftarrow 0$
2. $Y^k \leftarrow 0, Q^k \leftarrow 0, (Y^k, Q^k \in \mathbb{R}^{n_1\times n_2})$
3. $X^{k+1} \leftarrow U \max\{\Sigma - \frac{\eta}{\rho}, 0\}V^\top$, where $U\Sigma V^\top$ is an SVD of $(Y^k - Q^k)$.
4. $Y^{k+1} \leftarrow \min\left\{\max\left\{X^{k+1} + Q^k + \frac{1}{\rho}W, 0\right\}, 1\right\}$
5. $Q^{k+1} \leftarrow Q^k + X^{k+1} - Y^{k+1}$
6. If $\|X^{k+1} - Y^{k+1}\|_F \le \epsilon \max\{\|X^{k+1}\|_F, \|Y^{k+1}\|_F\}$ and $\|Y^{k+1} - Y^k\|_F \le \epsilon\|Q^{k+1}\|_F$ then stop and output $Y = Y^{k+1}$.
7. (Optional) Update ρ and Q^{k+1}
8. $k \leftarrow k + 1$, go to step 3.

6. Empirical results

In this section we report empirical results by applying Algorithm 1 to a variety of both synthetic and real data sets.[12] In our experiments, unless specified otherwise, we report the "full recovery rate" based on 100 repeated trials, i.e., the fraction of trials where the output of Algorithm 1 (after rounding to the nearest 0 and 1) equals Y^* exactly. Error bars show 95% confidence intervals.

6.1 Clustering with General Labels

We first evaluate graph clustering performance on a generic graph model with 5 labels. We use $n = 200$ with 4 equal-size clusters. In each experiment, two distributions μ and ν are randomly chosen from a uniform prior over all distributions as the in-cluster and the cross-cluster label distributions. Then, 100 random graphs are generated using this (μ, ν) pair, and each clustering outcome is checked against the corresponding ground truth. This is repeated 500 times to get a large variety of pairs. In other words, 500 random (μ, ν) pairs are generated, each tested on 100 random graphs. According to Theorem 3, the KL-divergence between μ and ν is the key deciding factor for the successful recovery of the underlying true clusters. The results are shown in Figure 1. In the left panel of Figure 1, we use the sum $D(\mu\|\nu) + D(\nu\|\mu)$ as the predictor (the horizontal axis is cut off at 2 since beyond this range all recovery rates are essentially 1). In the right panel, we use $\min\{D(\mu\|\nu), D(\nu\|\mu)\}$ as the predictor. The results indeed support the theoretical prediction as given by conditions (7) and (8) of Theorem 3.

12. Although we only report results from Algorithm 1 and for standard clustering, we note that similar results are obtained with Algorithm 2 and in the bi-clustering case as well.

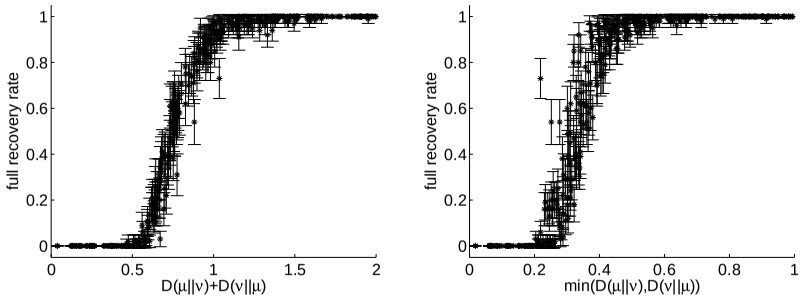


Figure 1: Clustering performance under different label distributions

6.2 Clustering Sparse/Dense Graphs with Partial Observations

In the next experiment, we test the planted partition model with partial observations. The model is as described in Section 4.2. Each pair is observed with probability s. For each observed in-cluster pairs, an edge is generated with probability p while for each observed cross-cluster pairs, an edge is generated with probability q. We consider both sparse graphs (p and q close to 0) and dense graphs (p and q close to 1). For the sparse case, we fix $q = 0.02$ and vary p, whereas for the dense case, we fix $p = 0.98$ and vary q.

Figures 2 and 3 show the results for graphs with $n = 200$, and Figures 4 and 5 show the results for $n = 1000$, with 4 equal-size clusters in both cases. For $n = 200$, we set the observation probability to $s = 0.8$; for $n = 1000$, we use $s = 0.5$ since the problem is significantly easier.

For comparison, we include results for the MLE weights (w^{MLE}), the linear weights (w^{LIN}), and the uniform weights. An imputation scheme labeled "MLE (no partial)", where all unobserved entries are treated as "no edge", is also included.

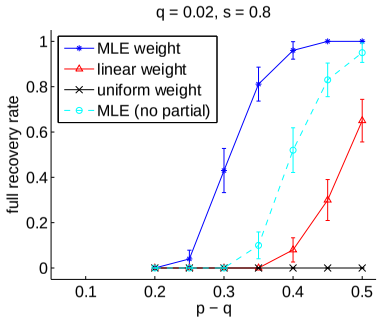


Figure 2: Clustering sparse graph ($n = 200$)

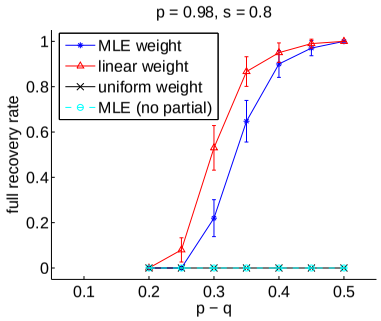


Figure 3: Clustering dense graph ($n = 200$)

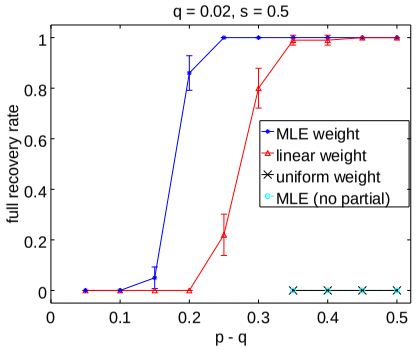


Figure 4: Clustering sparse graph ($n = 1000$)

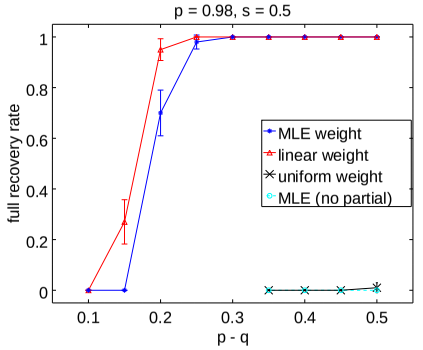


Figure 5: Clustering dense graph ($n = 1000$)

Corollary 11 predicts more success as the ratio $\frac{s(p-q)^2}{p(1-q)}$ gets larger. All else being the same, label distributions with small ζ (corresponding to sparse graphs in the planted partition setting, where $\frac{D(\nu\|\mu)}{D(\mu\|\nu)}$ is small) are easier to solve. Note that these predictions are with respect to the MLE weights. Both predictions are consistent with the empirical results given in Figures 2–5. The results also indicate that the MLE weights outperform the other weights in the sparse settings. On the other hand, in the dense case, we observe that the linear weights outperform the MLE weights by a small margin. This empirical observation is consistent with the prediction given in Section 3.4.

To give an idea of the computation time involved, Figures 6 and 7 plot the average CPU time needed to solve program (2) with Algorithm 1 on a typical quad-core desktop machine in Matlab. A commonly observed trend is that the number of iterations needed to converge is usually small when the problem is either too "hard" ($p - q$ small) or too "easy" ($p - q$ large). Note that we did not attempt to optimize the algorithm in terms of speed. Improvement in the computational aspect is certainly an interesting direction to explore.

We next examine the effect of varying cluster size K on the performance, with the total number of nodes held fixed. Figures 8 and 9 show clustering results with various values of K for $n = 400$. We choose a particular value of (p, q, s) that shows an interesting region. As expected, the success rates improve when K grows. The results remain qualitatively similar for other values of p, q and s.

6.2.1 COMPARISON WITH SPECTRAL METHODS

Lelarge et al. (2013) proposed a labeled stochastic block model, which is a special case of our model (cf. Section 1.1). In particular, their setup is restricted to only the two-cluster case, with balanced cluster sizes (i.i.d. uniform cluster membership). They proposed a spectral method specially tailored for this case. In this section, we compare our approach with theirs on the exact same setup that they use. In this setup, each pairwise observation can be an edge/non-edge. Each edge can take one of two possible labels. This is equivalent to a 3-label case in our setup. Figure 10 shows the results in terms of the "overlap" (as used by Lelarge et al. (2013)), which is a measure of how closely the resulting clusters match the ground truth. In Fig. 10, "convex" refers to results

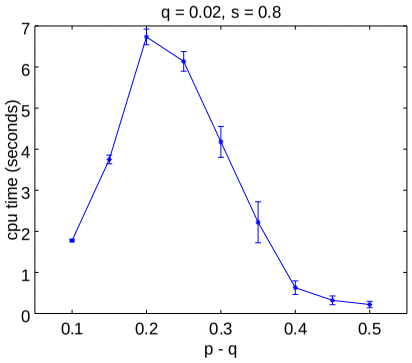


Figure 6: CPU time (sparse graph, $n = 200$)

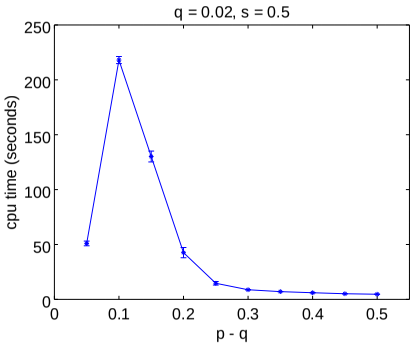


Figure 7: CPU time (sparse graph, $n = 1000$)

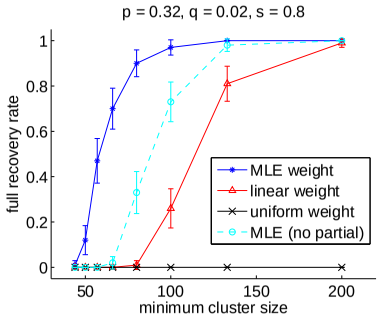


Figure 8: Clustering sparse graph ($n = 400$)

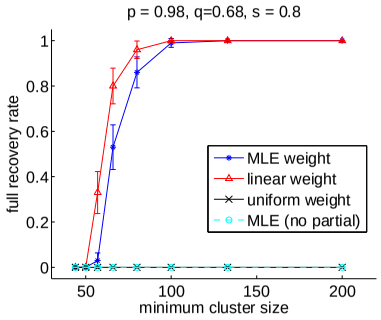


Figure 9: Clustering dense graph ($n = 400$)

based on Algorithm 1, where the final clustering is obtained by running K-means on the normalized rows of the output matrix. "Spectral" refers to results based on the proposed spectral method by Lelarge et al. (2013). We can observe that both approaches produce comparable results. Although computationally more costly,[13] our approach can handle a much wider range of problem setups, and as shown in Section 6.5.1, can significantly outperform the spectral method in many cases.

13. We note that computationally the cost is typically dominated by the SVD operations, especially for large n. For spectral clustering, only 1 SVD needs to be performed, whereas the ADMM solver performs 1 SVD per iteration. The number of iterations may range from just a few to several hundreds, depending on the problem.

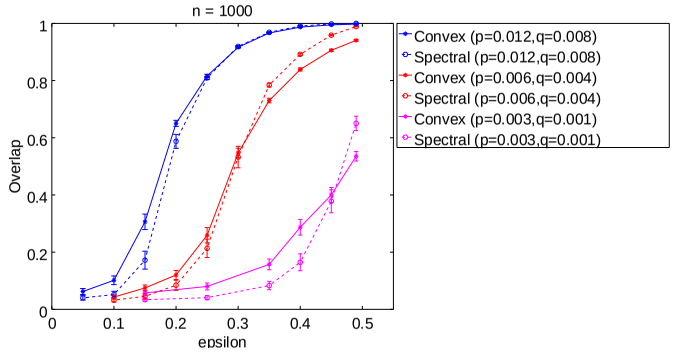


Figure 10: Comparison with spectral method. p and q are within-cluster and between-cluster edge probabilities, respectively. ϵ denotes the difference in within/between-cluster label distributions, where 0 is the smallest and 0.5 is the largest.

6.3 Gaussian Graphs and Inaccurate Weights

In this experiment, we evaluate real-valued labels drawn from Gaussian distributions. We also evaluate the effect of using weights that deviate from the MLE weights. Figure 11 shows clustering results on graphs with $n = 200$ and 8 equal-size clusters. The cross-cluster label distribution has mean 0 and variance 1. The in-cluster label distribution has the same variance but with elevated mean of $\mu = 2$ (blue) and $\mu = 1.5$ (red). Obviously, the case of $\mu = 2$ is an easier problem. In each case, we run Algorithm 1 using MLE weights that correspond to an in-cluster distribution with mean $\mu + \Delta$ instead of μ. Figure 11 shows that clustering performance drops when the deviation gets sufficiently large. Observe that negative Δ (conservative weights) performs better—consistent with Theorem 6. Figure 12 shows results based on the same settings, except that random noise is added to the (true) MLE weight, where the noise is uniformly distributed within the range $[-\Delta, \Delta]$. We observe a gradual drop of performance with respect to Δ, as predicted by Theorem 7.

6.4 Clustering with Highly Certain Observations

We empirically test the prediction of Theorem 9 regarding highly certain observations. For partially observed graphs with uncertain observations, the sufficient condition for recovery given by Corollary 13 requires that the fraction of observed entries, s, grows linearly with n (ignoring logarithmic term) if the other parameters p, q and K are held fixed. On the other hand, if all observed entries are highly certain, Theorem 9 predicts that s needs to grow only logarithmically with respect to n.

We run two experiments, each with $n = 600, 1200, 1800, 2400$. In all cases, the cluster size K is fixed at 150, and the labels are either "edge", "no edge" or "unobserved". In the first experiment we consider uncertain observations, where each observed entry is equal to the corresponding entry of Y^* with probability 0.7, and otherwise it is flipped. In the second experiment with highly certain

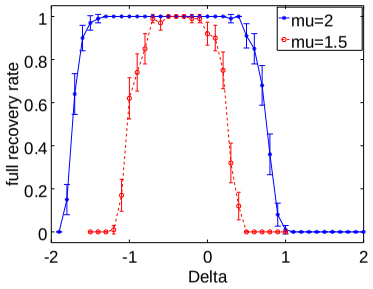


Figure 11: Systematic weight deviation

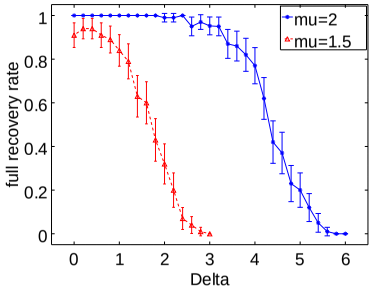


Figure 12: Random weight deviation

observations, all observed entries are equal to Y^* with probability 1. In both experiments, we search for the smallest fraction of observed entries needed to achieve a 90% full-recovery rate.

The results are shown in Figure 13. The plots indeed suggest a linear dependency on n for the uncertain observations, but a sub-linear dependency on n for the highly certain case.

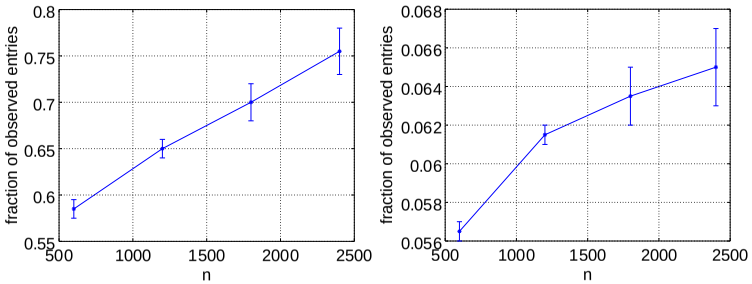


Figure 13: Fraction of observed entries needed for 90% full-recovery rate. Left panel: uncertain observations. Right panel: highly-certain observations.

6.5 Clustering Time-varying Graphs

We next investigate clustering performance on time-varying graphs. Figure 14 shows results based on multiple independent snapshots of partially observed graphs. Each graph is generated according to the planted partition model with partial observation (20% observed) as described in Section 4.2.1, with uniform error rate $q = 1 - p$ for all node pairs. We tested a wide range of error rates ($q \in [0.006, 0.325]$), and the horizontal axis tracks the corresponding KL divergence between μ and ν. We used $n = 200$ with 8 equal-size clusters. As predicted by Corollary 14, the clustering performance improves as the number of snapshots T grows.

Figure 15 shows results for Markov label sequences. Here, we test a simple model with two labels "interaction" and "no-interaction" (equivalent to "edge" and "no-edge"). For within-cluster pairs, the probability of interaction is greater (i.e., equal to $0.5 + \epsilon$) in the next time-step if there is no interaction in the current time-step, and vice versa. For inter-cluster pairs, the occurrence of interaction/no-interaction is completely random (i.e., $\epsilon = 0$) and therefore independent across snapshots. In this setting, both the marginal and stationary distributions for μ and ν are identical in every time step, and therefore at least two consecutive time steps are needed for informative clustering (cf. Section 4.4). The figure shows results for 200 nodes partitioned into 8 equal-size clusters, with the horizontal axis tracking the average KL-divergence between the conditional distributions (by varying ϵ). As predicted by Corollary 15, the performance improves as the number of snapshots T increases.

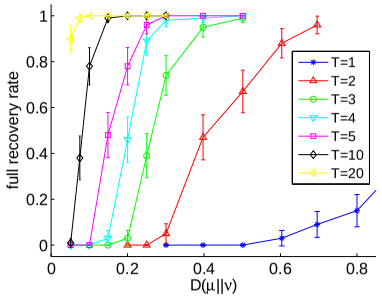


Figure 14: Independent snapshots

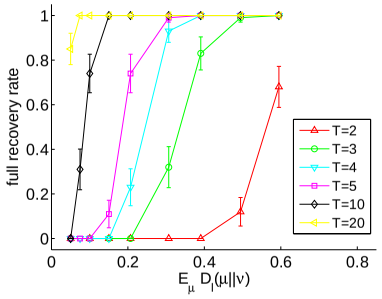


Figure 15: Markov snapshots

6.5.1 COMPARISON WITH EVOLUTIONARY SPECTRAL CLUSTERING

In this section, we compare our approach with an existing approach in clustering time-varying graphs based on evolutionary spectral clustering, as proposed by Chi et al. (2009). In particular, to handle time-varying graphs, one can assign a weight to each time step and compute a weighted average of the normalized graph Laplacian in each time step, then perform standard spectral clustering using this averaged graph Laplacian. We use the same settings as in Figure 14, except that we use the fully-observed case, to make sure that the comparison is fair.[14] For our approach, the final cluster assignments are obtained via running K-means on the normalized rows of the output matrix of the convex program. For spectral clustering, the final cluster assignments are based on K-means on the normalized rows of the top-K left singular vectors of the averaged graph Laplacian. For reference, we also include the results based on simply running K-means on the normalized rows of the averaged graph Laplacian. Figure 16 shows the results for $T = 1$, $T = 2$ and $T = 3$ in terms

14. Our method has extra advantage in the partially observed case, since it is generally unclear what weight to use for spectral clustering in this case. We tested a few imputation scheme for spectral clustering but the results are qualitatively the same.

of full clusters recovery rate. Figure 17 shows the pairwise error rate, which is the fraction of all node pairs that have been wrongly classified as either belonging to the same or different clusters.

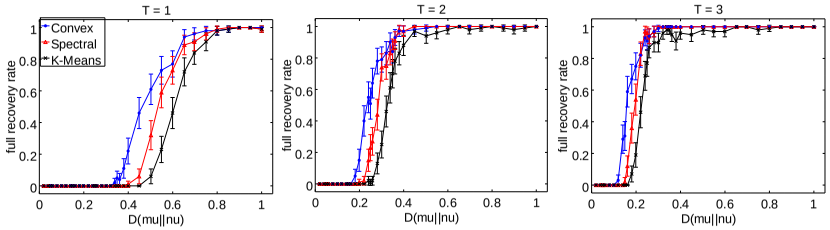


Figure 16: Comparison with evolutionary spectral clustering (full recovery rate)

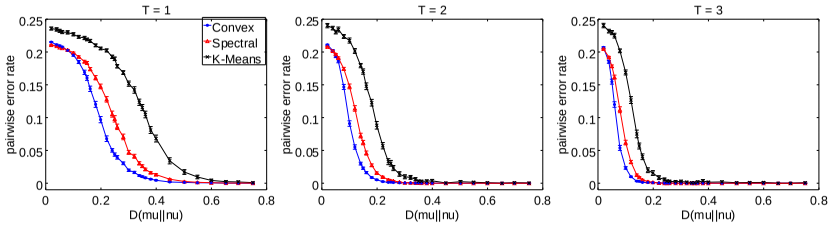


Figure 17: Comparison with evolutionary spectral clustering (pairwise error rate)

As expected, the overall performance improves as more snapshots are included. Among the three approaches, ours performs the best in terms of clustering error rates, but at the expense of higher computational cost relative to the other two.

6.6 Real-world Data Set

We consider three real-world data sets for which reliable ground truth is available. All three sets involve interactions among different social groups, with different clustering structures and different temporal granularity.

The *Reality Mining* data set (Eagle and Pentland, 2006) contains individuals from two main groups, the MIT Media Lab and the Sloan Business School, which we use as the ground-truth clusters. The data set records when two individuals interact, i.e., become proximal of each other or make a phone call, over a 9-month period. We choose a window of 14 weeks (the Fall semester) during which most individuals have non-empty interaction data. This sub-dataset consists of $n = 85$ individuals with 25 of them from Sloan and 60 from the Media Lab. We represent the data as a time-varying graph with 14 snapshots (one per week), each with two observations $\mathcal{L}_t = \{$"interact", "no-interact"$\}$: "interact" if a pair of individuals interact within the week, and "no-interact" otherwise. Note that with T snapshorts, the full label set $\mathcal{L} = \mathcal{L}_1 \times \ldots \times \mathcal{L}_T$ consists of all possible binary sequences of length T.

We compare three models on this data set: the Markov snapshot model in Section 4.4, the independent snapshot model in Section 4.3, and a single snapshot approach (i.e., our approach with $T = 1$) applied to a static graph generated by taking the union of all snapshots (i.e., two individuals are connected if they interact during any of the 14 weeks). In each trial, the in/cross-cluster label distributions are estimated from a fraction of randomly selected pairwise interaction data. In particular, each parameter (i.e., the interaction probability and the transition kernel) is estimated using the corresponding empirical frequency in the selected data, regularized by adding 1 to each count. We use the MLE weights in all instances. To ensure that a valid clustering is always obtained, we run K-means on the normalized rows of the output matrix of Algorithm 1.

Figure 18 shows the results with respect to varying number of snapshots (left panel) and varying number of training pairs used for parameter estimation (right panel). For comparison, we also added result based on evolutionary spectral clustering (Chi et al., 2009). The vertical axis shows the fraction of pairs whose cluster relationships are correctly identified. The accuracy generally improves when more snapshots are used. However, the improvement with respect to T is lower than expected, most likely due to the fact that the snapshots are not independent. The union model is expected to improve with T for *sparse* graphs—due to increase in the KL-divergence between the in/cross-cluster distributions. Overall, the Markov model managed to achieve higher accuracy than both the union and independent model, though the best accuracy is achieved by the evolutionary spectral clustering approach. Further inspection of the data suggests that the snapshots are non-stationary in the sense that the earlier snapshots have rather different label distributions than the later snapshots.

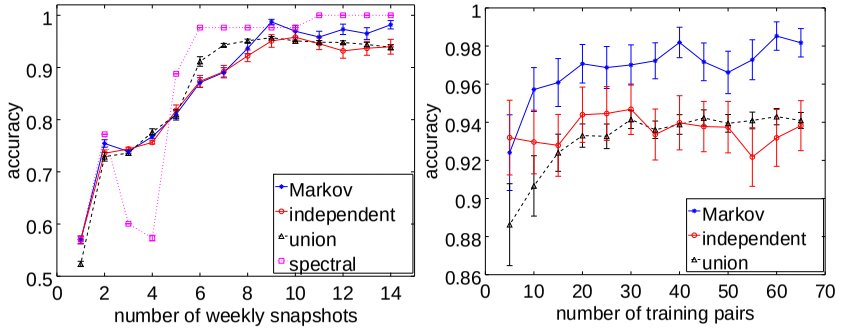


Figure 18: Reality Mining data set. Left: Varying number of snapshots (50 training pairs). Right: Varying number of training pairs (14 snapshots).

The next two data sets, *Workplace* and *Primary-school*, are from Genois et al. (2015) and Stehl et al. (2011), respectively. The Workplace data involves human contacts among 92 employees in an office building, where the ground truth clusters correspond to 5 different departments. We split this data set (over 10 days) into 10 daily snapshots, each with a binary $\{$"interact", "no-interact"$\}$ observation set. The Primary-school data involves contacts among 232 children in a primary school

(teachers are omitted), where the ground truth clusters correspond to 10 different classes. We split this data set (over 2 days) into 18 hourly snapshots, with the same binary observation per snapshot.

Figures 19 and 20 show the empirical results using the Markov snapshot, independent snapshot and single snapshot models described above, as well as evolutionary spectral clustering. All 3 models achieve comparable clustering accuracy in the Workplace data set, while the union model performs significantly worse in the Primary-school data set. The evolutionary spectral clustering approach performs worse overall—with large variations across varying number of snapshots. We believe that this is due to its sensitivity to the distribution of the top eigenvectors of the graph Laplacian.

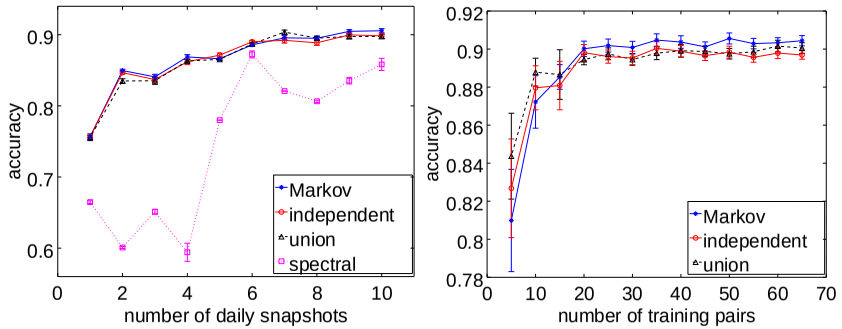


Figure 19: Workplace data set. Left: Varying number of snapshots (40 training pairs). Right: Varying number of training pairs (10 snapshots).

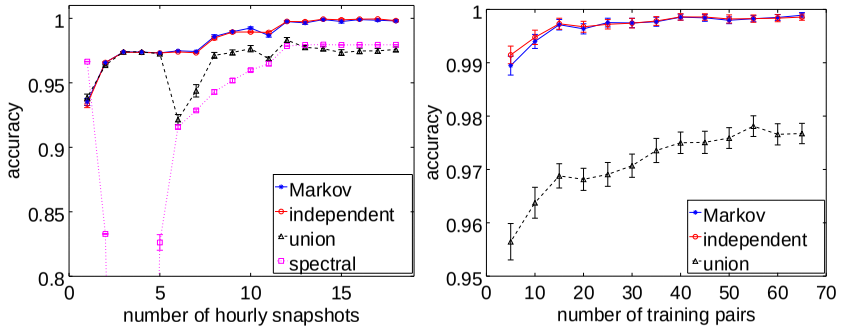


Figure 20: Primary-school data set. Left: Varying number of snapshots (50 training pairs). Right: Varying number of training pairs (18 snapshots).

7. Proof of Theorems 2 and 9

In this section, we prove the general results in the two main theorems. Observe that Theorem 2 is a special case of Theorem 9 with $\beta = 10$: if the weight function w is bounded, then there exists some b_∞ such that the set $\mathcal{L}_\infty$ is empty, in which case we have $\tilde{w}_b \equiv w$ and the conditions (14) and (15) in Theorem 9 reduce to (5) and (6) in Theorem 2. Therefore, it suffices to prove Theorem 9.

7.1 Notations

We need some additional notation to account for the bi-clustering setting. Recall that $|\mathcal{N}_1| = n_1$ and $\mathcal{N}_2 = n_2$, and set $n = \max\{n_1, n_2\}$. For a node $i \in \mathcal{N}_1$, let $\mathcal{C}_i$ be the cluster that contains node i and $K_i = |\mathcal{C}_i|$. Similarly, $\mathcal{C}'_j$ is the cluster that contains a node j in $\mathcal{N}_2$, and $K'_j = |\mathcal{C}'_j|$. Each cluster $\mathcal{C}_i$ in $\mathcal{N}_1$ is associated with exactly one cluster $\mathcal{C}'_j$ in $\mathcal{N}_2$, which is denoted by $\mathcal{C}_i \sim \mathcal{C}'_j$. These notations are consistent with those for clusteirng, with the understanding that $\mathcal{N}_1 = \mathcal{N}_2$, $n_1 = n_2 = n$, $\mathcal{C}_i = \mathcal{C}'_i$ and $K_i = K'_i$.

Let $\Omega = \{(i,j) : L_{ij} \in \mathcal{L}_\infty\}$ and $R = \{(i,j) : Y^*_{ij} = 1\}$. Let $\mathcal{P}_\Omega$ be the projection operator on matrices such that

$$(\mathcal{P}_\Omega Z)_{ij} = \begin{cases} Z_{ij} & \text{if } (i,j) \in \Omega, \\ 0 & \text{otherwise.} \end{cases}$$

The projections $\mathcal{P}_R$, $\mathcal{P}_{\Omega \cap R}$ etc are defined similarly.

Let $U\Sigma V^\top$ be the rank-r SVD of Y^*, where r is the number of groud-truth clusters and the rank of Y^*. Note that

$$(UU^\top)_{ij} = \begin{cases} \frac{1}{K_i} & \text{if } \mathcal{C}_i = \mathcal{C}_j, \\ 0 & \text{otherwise,} \end{cases} \tag{26}$$

$$(VV^\top)_{ij} = \begin{cases} \frac{1}{K'_i} & \text{if } \mathcal{C}'_i = \mathcal{C}'_j, \\ 0 & \text{otherwise,} \end{cases} \tag{27}$$

and

$$(UV^\top)_{ij} = \begin{cases} \frac{1}{\sqrt{K_i K'_j}} & \text{if } \mathcal{C}_i \sim \mathcal{C}'_j, \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

Define the projection operator

$$\mathcal{P}_T Z := UU^\top Z + ZVV^\top - UU^\top ZVV^\top \tag{29}$$

and its complementary projection

$$\mathcal{P}_{T^\perp} Z := Z - \mathcal{P}_T Z.$$

Denote by $\|\cdot\|$ the spectral norm (largest singular value) of a matrix. For any matrix X with $\|X\| \le \eta_0$, the matrix $UV^\top + \eta_0^{-1}\mathcal{P}_{T^\perp}X$ is a subgradient of the nuclear norm $\|\cdot\|_*$ at Y^* (Recht et al., 2010). It follows that for any feasible solution Y to the program (3), we have

$$0 \ge \|Y\|_* - \|Y^*\|_* \ge \langle UV^\top + \eta_0^{-1}\mathcal{P}_{T^\perp}X, Y - Y^* \rangle,$$

which implies

$$\langle X, Y^* - Y \rangle \ge \langle \mathcal{P}_T X - \eta_0 UV^\top, Y^* - Y \rangle. \tag{30}$$

7.2 Preliminary Lemmas

The proof of Theorem 9 builds on the following four lemmas. The first lemma gives a closed form expression for the projected matrix $\mathcal{P}_T Z$.

Lemma 16 *For any matrix Z and each index (i,j), we have*

$$(\mathcal{P}_T Z)_{ij} = \frac{1}{K_i}\sum_{k\in\mathcal{C}_i} Z_{kj} + \frac{1}{K'_j}\sum_{l\in\mathcal{C}'_j} Z_{il} - \frac{1}{K_i K'_j}\sum_{k\in\mathcal{C}_i}\sum_{l\in\mathcal{C}'_j} Z_{kl}. \tag{31}$$

Proof The lemma is immediate by the definition (29) of the projection $\mathcal{P}_T$ and the expressions (26) and (27) for the matrices $UU^\top$ and $VV^\top$. ■

Recall that W is the weight matrix used in program (3). The second lemma controls the spectral norm and the element magnitudes of the matrix $\mathcal{P}_\Omega UV^\top$.

Lemma 17 *Define*

$$\eta_1 := c_1 \frac{\beta\log n + \sqrt{K s_\mu(1-s_\mu)\beta\log n}}{K}.$$

With probability at least $1-n^{-\beta}$ the followings hold:

$$\|\mathcal{P}_\Omega UV^\top - \mathbb{E}[\mathcal{P}_\Omega UV^\top]\| \le \eta_1 \tag{32}$$

and for all i,j

$$|(\mathcal{P}_T(\mathcal{P}_\Omega UV^\top - \mathbb{E}[\mathcal{P}_\Omega UV^\top]))_{ij}| \le \begin{cases} \frac{\eta_1}{\sqrt{K_i K'_j}} & \text{if } \mathcal{C}_i \sim \mathcal{C}'_j \\ 0 & \text{otherwise} \end{cases} \tag{33}$$

Proof For the first inequality (32), consider $\mathcal{P}_\Omega UV^\top - \mathbb{E}[\mathcal{P}_\Omega UV^\top]$ as the sum of independent, zero-mean random matrices:

$$\mathcal{P}_\Omega UV^\top - \mathbb{E}[\mathcal{P}_\Omega UV^\top] = \sum_{(i,j),\mathcal{C}_i\sim\mathcal{C}'_j} X_{i,j},$$

where, noting the expression (28), we define

$$X_{i,j} := \mathcal{P}_\Omega\left(\frac{1}{\sqrt{K_i K'_j}}(e_i e_j^\top)\right) - \mathbb{E}\mathcal{P}_\Omega\left(\frac{1}{\sqrt{K_i K'_j}}(e_i e_j^\top)\right)$$

with e_i denoting the i-th standard basis vector. Note that

$$\|X_{i,j}\| \le \frac{1}{K}, \quad \forall i,j,$$

and

$$\left\|\sum_{(i,j),\mathcal{C}_i\sim\mathcal{C}'_j} \mathbb{E}X_{i,j}X_{i,j}^\top\right\| = \left\|\sum_{(i,j),\mathcal{C}_i\sim\mathcal{C}'_j} \frac{s_\mu(1-s_\mu)}{K_i K'_j}(e_i e_i^\top)\right\| \le \frac{s_\mu(1-s_\mu)}{K}.$$

Similarly we have $\left\|\sum_{(i,j),\mathcal{C}_i\sim\mathcal{C}'_j} \mathbb{E}X_{i,j}^\top X_{i,j}\right\| \le \frac{s_\mu(1-s_\mu)}{K}$. By applying the matrix Bernstein inequality (Tropp, 2012), we obtain the desired inequality (32).

Turning to the inequality (33), let $Z := \mathcal{P}_\Omega UV^\top - \mathbb{E}[\mathcal{P}_\Omega UV^\top]$. Note that each entry of Z is an independent zero-mean random variable. From the expression (31), let $\hat{z}$ be the first term of its RHS, which is an average of K_i independent zero-mean random variables with $|Z_{kj}| \le \frac{1}{\sqrt{K_i K'_j}}$ and

$$\mathrm{Var}(Z_{kj}) = \frac{s_\mu(1-s_\mu)}{K_i K'_j}.$$

By the standard Bernstein's inequality, we obtain that $|\hat{z}| \le \frac{\eta_1}{\sqrt{K_i K'_j}}$ with probability at least $1-n^{-\beta-2}$. The same reasoning can be used to arrive at the same bound for the second and the third RHS term of expression (31). Applying a union bound over all (i,j) we obtain the desired inequality (33). ■

The next lemma is an analogue of Lemma 17 and controls the matrix $\mathcal{P}_{\Omega^c\cap R}W$.

Lemma 18 *Define*

$$\eta_2 := c_2\left(b\beta\log n + \sqrt{n\mathrm{Var}_\mu(\tilde{w}_b)\beta\log n}\right),$$
$$\eta_3 := c_3\left(b\xi\beta\log n + \sqrt{n\mathrm{Var}_\mu(\tilde{w}_b)\beta\log n}\right).$$

With probability at least $1-n^{-\beta}$ the followings hold:

$$\|\mathcal{P}_{\Omega^c\cap R}W - \mathbb{E}[\mathcal{P}_{\Omega^c\cap R}W]\| \le \eta_2 \tag{34}$$

and for all i,j

$$|(\mathcal{P}_T(\mathcal{P}_{\Omega^c\cap R}W - \mathbb{E}[\mathcal{P}_{\Omega^c\cap R}W]))_{ij}| \le \begin{cases} \min\left\{\frac{\eta_2}{K}, \frac{\eta_3}{\sqrt{K_i K'_j}}\right\} & \text{if } \mathcal{C}_i\sim\mathcal{C}'_j, \\ 0 & \text{otherwise.} \end{cases} \tag{35}$$

Proof This claim follows the same arguments as in the proof of Lemma 17. ■

The last lemma is again analogous to Lemma 17 and controls the matrix $\mathcal{P}_{\Omega^c\cap R^c}W$.

Lemma 19 *Define*

$$\eta_4 := c_4\left(b\beta\log n + \sqrt{n\mathrm{Var}_\nu(\tilde{w}_b)\beta\log n}\right),$$
$$\eta_5 := c_5\frac{b\beta\log n + \sqrt{K\mathrm{Var}_\nu(\tilde{w}_b)\beta\log n}}{K}.$$

With probability at least $1-n^{-\beta}$ the followings hold:

$$\|\mathcal{P}_{\Omega^c\cap R^c}W - \mathbb{E}[\mathcal{P}_{\Omega^c\cap R^c}W]\| \le \eta_4 \tag{36}$$

and for all i,j

$$|(\mathcal{P}_T(\mathcal{P}_{\Omega^c\cap R^c}W - \mathbb{E}[\mathcal{P}_{\Omega^c\cap R^c}W]))_{ij}| \le \begin{cases} 0 & \text{if } \mathcal{C}_i\sim\mathcal{C}'_j, \\ \eta_5 & \text{otherwise,} \end{cases} \tag{37}$$

Proof This claim follows the same arguments as in the proof of Lemma 17. ■

Remark 20 *The $\sqrt{\log n}$ factors in the expressions of η_2, η_3 and η_4 can be removed via a more careful analysis, for example by using the results in Bandeira and van Handel (2016). We do not delve into the details here.*

7.3 Proof of Theorem 9

We are now ready to complete the proof of Theorem 9. We will show that with probability at least $1 - n^{-\beta}$ the following inequality holds for all $Y \neq Y^*$ feasible to the program (3):

$$\langle W, Y^* - Y \rangle > 0,$$

which implies that Y^* is the unique solution of program (3). Note that the feasible set of the program (2) is a subset of the feasible set of the program (3) (but always contains Y^*). This means that whenever Y^* is the unique solution of the program (3), it is also the unique solution of the program (2).

To proceed, we decompose $\langle W, Y^* - Y \rangle$ as follows:

$$\begin{aligned}\langle W, Y^* - Y \rangle =& \langle \mathcal{P}_{\Omega \cap R} W, Y^* - Y \rangle + \langle \mathcal{P}_{\Omega \cap R^c} W, Y^* - Y \rangle + \langle \mathcal{P}_{\Omega^c \cap R} W, Y^* - Y \rangle \\ &+ \langle \mathcal{P}_{\Omega^c \cap R^c} W, Y^* - Y \rangle.\end{aligned} \tag{38}$$

We analyze separately the four terms on the RHS above.

For the first RHS term, note that by the assumption (12), all entries of the matrix $\mathcal{P}_{\Omega \cap R} W$ are positive with probability one. We therefore have

$$\begin{aligned}&\langle \mathcal{P}_{\Omega \cap R} W, Y^* - Y \rangle \\ &\geq b_\infty K \langle \mathcal{P}_\Omega UV^\top, Y^* - Y \rangle \\ &= b_\infty K \left(\langle \mathbb{E}[\mathcal{P}_\Omega UV^\top], Y^* - Y \rangle + \langle \mathcal{P}_\Omega UV^\top - \mathbb{E}[\mathcal{P}_\Omega UV^\top], Y^* - Y \rangle \right) \\ &= b_\infty K \left(\langle s_\mu UV^\top, Y^* - Y \rangle + \langle \mathcal{P}_\Omega UV^\top - \mathbb{E}[\mathcal{P}_\Omega UV^\top], Y^* - Y \rangle \right) \\ &\overset{(a)}{\geq} b_\infty K \left(\langle s_\mu UV^\top, Y^* - Y \rangle + \langle \mathcal{P}_T(\mathcal{P}_\Omega UV^\top - \mathbb{E}[\mathcal{P}_\Omega UV^\top]) - \eta_1 UV^\top, Y^* - Y \rangle \right) \\ &\overset{(b)}{\geq} b_\infty K \left(\langle s_\mu UV^\top, Y^* - Y \rangle + \langle -2\eta_1 UV^\top, Y^* - Y \rangle \right) \\ &= b_\infty K (s_\mu - 2\eta_1) \langle UV^\top, Y^* - Y \rangle,\end{aligned}$$

where we apply the bounds (30) and (32) in the step (a), and the bound (33) in the step (b). Again by assumption (12) we have $\langle \mathcal{P}_{\Omega \cap R} W, Y^* - Y \rangle \geq 0$. We conclude that

$$\langle \mathcal{P}_{\Omega \cap R} W, Y^* - Y \rangle \geq \max\{0, b_\infty K (s_\mu - 2\eta_1)\} \cdot \langle UV^\top, Y^* - Y \rangle. \tag{39}$$

For the second RHS term in (38), it follows immediately from the assumption (12) that

$$\langle \mathcal{P}_{\Omega \cap R^c} W, Y^* - Y \rangle \geq 0. \tag{40}$$

Turning to the third RHS term in (38), we have

$$\begin{aligned}&\langle \mathcal{P}_{\Omega^c \cap R} W, Y^* - Y \rangle \\ =& \langle \mathbb{E}[\mathcal{P}_{\Omega^c \cap R} W], Y^* - Y \rangle + \langle \mathcal{P}_{\Omega^c \cap R} W - \mathbb{E}[\mathcal{P}_{\Omega^c \cap R} W], Y^* - Y \rangle \\ =& \langle (\mathbb{E}_\mu \tilde{w}_b) Y^*, Y^* - Y \rangle + \langle \mathcal{P}_{\Omega^c \cap R} W - \mathbb{E}[\mathcal{P}_{\Omega^c \cap R} W], Y^* - Y \rangle \\ \overset{(a)}{\geq}& \langle (\mathbb{E}_\mu \tilde{w}_b) Y^*, Y^* - Y \rangle + \langle \mathcal{P}_T(\mathcal{P}_{\Omega^c \cap R} W - \mathbb{E}[\mathcal{P}_{\Omega^c \cap R} W]) - \eta_2 UV^\top, Y^* - Y \rangle \\ \overset{(b)}{\geq}& \max\left\{ \left(\mathbb{E}_\mu \tilde{w}_b - \frac{2\eta_2}{K} \right) \langle Y^*, Y^* - Y \rangle, (K \mathbb{E}_\mu \tilde{w}_b - 2\eta_3) \langle UV^\top, Y^* - Y \rangle \right\},\end{aligned} \tag{41}$$

where we use the bounds (30) and (34) in the step (a), and the bound (35) in the step (b); also in (b) we use the inequality $\langle (\mathbb{E}_\mu \tilde{w}_b) Y^*, Y^* - Y \rangle \geq \langle (K \mathbb{E}_\mu \tilde{w}_b) UV^\top, Y^* - Y \rangle$, which follows from the assumption (13).

Finally, letting J denote the all one matrix, we can bound the last RHS term in (38) as

$$\begin{aligned}&\langle \mathcal{P}_{\Omega^c \cap R^c} W, Y^* - Y \rangle \\ &= \langle \mathbb{E}[\mathcal{P}_{\Omega^c \cap R^c} W], Y^* - Y \rangle + \langle \mathcal{P}_{\Omega^c \cap R^c} W - \mathbb{E}[\mathcal{P}_{\Omega^c \cap R^c} W], Y^* - Y \rangle \\ &= (-\mathbb{E}_\nu \tilde{w}_b) \langle J - Y^*, Y \rangle + \langle \mathcal{P}_{\Omega^c \cap R^c} W - \mathbb{E}[\mathcal{P}_{\Omega^c \cap R^c} W], Y^* - Y \rangle \\ &\overset{(a)}{\geq} (-\mathbb{E}_\nu \tilde{w}_b) \langle J - Y^*, Y \rangle + \langle \mathcal{P}_T(\mathcal{P}_{\Omega^c \cap R^c} W - \mathbb{E}[\mathcal{P}_{\Omega^c \cap R^c} W]) - \eta_4 UV^\top, Y^* - Y \rangle \\ &\overset{(b)}{\geq} (-\mathbb{E}_\nu \tilde{w}_b - \eta_5) \langle J - Y^*, Y \rangle - \langle \eta_4 UV^\top, Y^* - Y \rangle,\end{aligned} \tag{42}$$

where we use the bounds (30) and (36) in the step (a), and the bound (37) in the step (b).

Applying the above inequalities (39), (40), (41) and (42) to the equation (38), we obtain that

$$\begin{aligned}\langle W, Y^* - Y \rangle \geq& \max\left\{ \left(\mathbb{E}_\mu \tilde{w}_b - \frac{2\eta_2 + \eta_4}{K} \right) \langle Y^*, Y^* - Y \rangle, \right. \\ &\left. (b_\infty K (s_\mu - 2\eta_1) + K \mathbb{E}_\mu \tilde{w}_b - 2\eta_3 - \eta_4) \langle UV^\top, Y^* - Y \rangle \right\} \\ &+ (-\mathbb{E}_\nu \tilde{w}_b - \eta_5) \langle J - Y^*, Y \rangle \\ =& \max\left\{ \left(\mathbb{E}_\mu \tilde{w}_b - \frac{2\eta_2 + \eta_4}{K} \right) \langle Y^*, Y^* - Y \rangle, \right. \\ &\left. b_\infty K \left(s_\mu + \frac{\mathbb{E}_\mu \tilde{w}_b}{b_\infty} - \left(2\eta_1 + \frac{2\eta_3 + \eta_4}{b_\infty K} \right) \right) \langle UV^\top, Y^* - Y \rangle \right\} \\ &+ (-\mathbb{E}_\nu \tilde{w}_b - \eta_5) \langle J - Y^*, Y \rangle.\end{aligned} \tag{43}$$

We see that the condition (14) in the theorem statement ensures that $-\mathbb{E}_\nu \tilde{w}_b - \eta_5 > 0$. On the other hand, the condition (15) ensures that

$$\mathbb{E}_\mu \tilde{w}_b - \frac{2\eta_2 + \eta_4}{K} > 0,$$

whereas the condition (16) ensures that

$$s_\mu + \frac{\mathbb{E}_\mu \tilde{w}_b}{b_\infty} - \left(2\eta_1 + \frac{2\eta_3 + \eta_4}{b_\infty K} \right) > 0.$$

Either of last two inequalities is sufficient to guarantee that the right hand side of the equation (43) is strictly positive. This completes the proof of Theorem 9.

8. Conclusion

In this paper we presented a general framework for graph clustering by assuming that all pairwise observations are in the form of labels. The algorithm involves solving a tractable convex optimization problem with an appropriately weighted objection function based on the observed labels.

A key contribution of our theoretical results is in showing that the MLE weights are order-wise optimal under a generalized version of the stochastic block model, thus providing a principled way of assigning weights to the observed graph. Our main results also identify the relevant parameters that are crucial to the successful recovery of the underlying clusters. These include the minimum cluster size, the distance between the label distributions, as well as properties of the observations such as the number of snapshots in a time-varying graph.

This framework recovers as special cases a broad range of existing results in graph clustering, and in fact provides substantial improvement for many of them. Important features such as partial observability and non-uniform uncertainties can be readily analyzed using our results, which provide new insights in many applications. Moreover, our framework is powerful enough to yield new results on novel settings such as the clustering of time-varying graphs.

An interesting future direction is to extend the proposed approach to problems with more complex structures such as overlapping clusters. Another important direction is to develop scalable solution applicable to very large data sets, both in terms of storage and computational complexity.

Acknowledgments

S.H. Lim and H. Xu were supported by the Ministry of Education of Singapore through AcRF Tier Two grant R-265-000-443-112. H. Xu was also supported by the School of Industrial and System Engineering of Georgia Institute of Technology. Y. Chen was supported by the National Science Foundation CRII award 1657420 and the School of Operations Research and Information Engineering at Cornell University.

Appendix A. Proofs of Theorem 3 and Corollary 4

In this section, we prove Theorem 3 and Corollary 4 for using MLE weights.

A.1 Proof of Theorem 3

Throughout this subsection, w always means the MLE weight function w^{MLE}. The proof is based on several lemmas. The first lemma, proved in Section A.1.1 to follow, bounds the log-likelihood ratios.

Lemma 21 *Suppose that* $\left|\log \frac{\mu(l)}{\nu(l)}\right| \le b, \forall l \in \mathcal{L}$. *Then, for any* $l \in \mathcal{L}$,

$$\left|\log \frac{\mu(l)}{\nu(l)}\right| \le (b+2)\left|\frac{\mu(l)-\nu(l)}{\mu(l)+\nu(l)}\right|.$$

The second lemma, proved in Section A.1.2 to follow, controls the variance terms.

Lemma 22 *Suppose that* $|\log \frac{\mu(l)}{\nu(l)}| \le b, \forall l \in \mathcal{L}$ *and* $D(\nu\|\mu) \le \zeta D(\mu\|\nu)$, *then*

$$\mathrm{Var}_\nu w \le 3(b+2)D(\nu\|\mu) \tag{44}$$

and

$$\max(\mathrm{Var}_\mu w, \mathrm{Var}_\nu w) \le (\zeta+1)(b+2)D(\mu\|\nu). \tag{45}$$

The last lemma is a classical result in information theory. The lemma bounds the KL divergence $D(\mu\|\nu)$ and $D(\nu\|\mu)$ in terms of the triangle discrimination between μ and ν.

Lemma 23 (Topsoe 2000) *The following holds for any distributions* μ *and* ν, *assuming that* μ *and* ν *are absolutely continuous with each other and with respect to a base measure* λ:

$$\min\{D(\mu\|\nu), D(\nu\|\mu)\} \ge \frac{1}{2}\int_{\mathcal{L}} \frac{(\mu(l)-\nu(l))^2}{\mu(l)+\nu(l)}\,\mathrm{d}\lambda.$$

We are now ready to prove Theorem 3. To this end, we verify that the conditions (5) and (6) in Theorem 2 are satisfied under the assumption of Theorem 3. Note that

$$-\mathbb{E}_\nu w = -\int_{\mathcal{L}} \log\frac{\mu(l)}{\nu(l)}\,\mathrm{d}\nu = D(\nu\|\mu).$$

Combining the assumption (7) in Theorem 3 with the previous bound (44), the first condition (5) in Theorem 2 is satisfied as follows:

$$\begin{aligned}\frac{b\log n}{K} + \frac{\sqrt{K\log n}\sqrt{\mathrm{Var}_\nu w}}{K} &\le \frac{(b+2)\log n}{K} + \sqrt{\frac{\log n}{K}}\sqrt{3(b+2)D(\nu\|\mu)} \\ &\le cD(\nu\|\mu) + c'\sqrt{D(\nu\|\mu)}\sqrt{D(\nu\|\mu)} \\ &\le c''D(\nu\|\mu) = -c''\mathbb{E}_\nu w.\end{aligned}$$

Turning to the condition (6), we note that $\mathbb{E}_\mu w = D(\mu\|\nu)$. Combining the assumption (8) in Theorem 3 with the previous bound (45) in a similar manner establishes the condition (6).

Finally, the last sentence in the statement of Theorem 3 is a special case of the following more general result, which is useful later in the proof of Theorem 5.

Lemma 24 *Suppose that* $\left|\log \frac{\mu(l)}{\nu(l)}\right| \le b, \forall l \in \mathcal{L}$. *Then we have*

$$D(\nu\|\mu) + D(\mu\|\nu) \le (b+2)\int_{\mathcal{L}} \frac{(\nu(l)-\mu(l))^2}{\nu(l)+\mu(l)}\,\mathrm{d}\lambda$$

and

$$\frac{1}{2b+3} \le \frac{D(\nu\|\mu)}{D(\mu\|\nu)} \le 2b+3.$$

Proof We have

$$\begin{aligned}\frac{D(\nu\|\mu)}{D(\mu\|\nu)}+1 &= \frac{D(\nu\|\mu)+D(\mu\|\nu)}{D(\mu\|\nu)}\\ &= \frac{\int_{\mathcal{L}}(\nu(l)-\mu(l))\log\frac{\nu(l)}{\mu(l)}\,\mathrm{d}\lambda}{D(\mu\|\nu)}\\ &\overset{(a)}{\leq} \frac{(b+2)\int_{\mathcal{L}}\frac{(\nu(l)-\mu(l))^2}{\nu(l)+\mu(l)}\,\mathrm{d}\lambda}{D(\mu\|\nu)},\end{aligned}$$

where we use Lemma 21 in the inequality (a). This proves the first equation in the lemma. Bounding the right side of (a) using Lemma 23, we prove the upper bound in the second equation of the lemma. The lower bound in the second equation follows from switching the roles of μ and ν. ■

A.1.1 PROOF OF LEMMA 21

Consider the function $g(p)=\log\frac{1-p}{p}$ for $p\in[\frac{1}{e^b+1},0.5]$. By the convexity of $g(p)$ in this range we can linearly upper-bound it and show that $\log\frac{1-p}{p}\leq b\left(\frac{e^b+1}{e^b-1}\right)(1-2p)$. Taking $p=\frac{\nu}{\mu+\nu}$, we then have

$$\left|\log\frac{\mu}{\nu}\right| = \left|\log\frac{1-p}{p}\right| \leq b\left(\frac{e^b+1}{e^b-1}\right)|1-2p| = b\left(\frac{e^b+1}{e^b-1}\right)\left|\frac{\mu-\nu}{\mu+\nu}\right| \leq (b+2)\left|\frac{\mu-\nu}{\mu+\nu}\right|.$$

A.1.2 PROOF OF LEMMA 22

We need a version of the Padé approximation for logarithms:

$$\log\frac{1}{x}\geq\frac{(1-x)(5+x)}{2+4x},\quad\forall x>0,$$

which follows from that fact that the function $g(x)=\log\frac{1}{x}-\frac{(1-x)(5+x)}{2+4x}$ has a unique minimum $g(1)=0$. Using this inequality, we obtain that

$$\begin{aligned}3\int_{\mathcal{L}}\log\frac{\nu(l)}{\mu(l)}\,\mathrm{d}\nu-\int_{\mathcal{L}}\left|\frac{\mu(l)-\nu(l)}{\mu(l)+\nu(l)}\right|\left|\log\frac{\mu(l)}{\nu(l)}\right|\,\mathrm{d}\nu &= 2\int_{\mathcal{L}}\frac{2\mu(l)+\nu(l)}{\mu(l)+\nu(l)}\log\frac{\nu(l)}{\mu(l)}\,\mathrm{d}\nu\\ &\geq 2\int_{\mathcal{L}}\frac{2\mu(l)+\nu(l)}{\mu(l)+\nu(l)}\frac{(1-\frac{\mu(l)}{\nu(l)})(5+\frac{\mu(l)}{\nu(l)})}{2+4\frac{\mu(l)}{\nu(l)}}\,\mathrm{d}\nu\\ &= \int_{\mathcal{L}}\frac{(\nu(l)-\mu(l))(5\nu(l)+\mu(l))}{\mu(l)+\nu(l)}\,\mathrm{d}\lambda\\ &= -4+8\int_{\mathcal{L}}\frac{\mu(l)^2}{\mu(l)+\nu(l)}\,\mathrm{d}\lambda\\ &\overset{(a)}{\geq} 0,\end{aligned}$$

where the last inequality (a) follows from the fact that $\int_{\mathcal{L}}\frac{\mu(l)^2}{\mu(l)+\nu(l)}\,\mathrm{d}\lambda\geq\frac{1}{2}$. The first inequality (44) in the lemma then follows from

$$\begin{aligned}\mathrm{Var}_\nu w &\leq \mathbb{E}_\nu w^2\\ &= \int_{\mathcal{L}}\left(\log\frac{\mu(l)}{\nu(l)}\right)^2\,\mathrm{d}\nu\\ &\overset{(a)}{\leq} (b+2)\int_{\mathcal{L}}\left|\frac{\mu(l)-\nu(l)}{\mu(l)+\nu(l)}\right|\left|\log\frac{\mu(l)}{\nu(l)}\right|\,\mathrm{d}\nu\\ &\overset{(b)}{\leq} 3(b+2)\int_{\mathcal{L}}\log\frac{\nu(l)}{\mu(l)}\,\mathrm{d}\nu\\ &= 3(b+2)D(\nu\|\mu),\end{aligned}$$

where the step (a) follows from Lemma 21 and the step (b) follows from the inequality above.

For the second inequality (45), again by using Lemma 21 we obtain that

$$\begin{aligned}\max(\mathrm{Var}_\mu w,\mathrm{Var}_\nu w) &\leq \mathbb{E}_\mu w^2+\mathbb{E}_\nu w^2\\ &= \int_{\mathcal{L}}(\mu(l)+\nu(l))\left(\log\frac{\mu(l)}{\nu(l)}\right)^2\,\mathrm{d}\lambda\\ &\leq (b+2)\int_{\mathcal{L}}(\mu(l)+\nu(l))\left|\frac{\mu(l)-\nu(l)}{\mu(l)+\nu(l)}\right|\left|\log\frac{\mu(l)}{\nu(l)}\right|\,\mathrm{d}\lambda\\ &= (b+2)\big(D(\mu\|\nu)+D(\nu\|\mu)\big)\\ &\leq (\zeta+1)(b+2)D(\mu\|\nu).\end{aligned}$$

A.2 Proof of Corollary 4

The corollary follows immediately from Theorem 3, by lower bounding the left hand sides of the equations (7) and (8) using Lemma 23.

Appendix B. Proof of Theorem 5

In this section, we prove the converse result in Theorem 5. We use a standard technique of converting a statistical estimation problem to multiple hypothesis testing—in particular, we shall apply Theorem 2.5 of Tsybakov (2009). We will consider the standard clustering case where Y^* is symmetric. Set $M=n$ and let $\theta_0\in\{0,1\}^{n\times n}$ be a fixed cluster matrix corresponding to two equal sized clusters. For $k=1,\ldots,\frac{M}{2}$, let θ_k be the cluster matrix of a new clustering by swapping the 1st member of cluster 1 with the k-th member of cluster 2. Likewise, for $k=\frac{M}{2}+1,\ldots,M$, θ_k is obtained by swapping the 2nd member of cluster 1 with the k-th member of cluster 2. Let $L^0,L^1,\ldots,L^M$ be the random label matrices generated by the corresponding clustering.

Since the label of each pair (i,j) is generated independently, we have:

$$\begin{aligned} D(L^j \| L^0) &= \sum_{i<j} D(L^k_{ij} \| L^0_{ij}) \\ &\overset{(a)}{=} (n-2) D(\mu\|\nu) + (n-2) D(\nu\|\mu) \\ &\overset{(b)}{\leq} (n-2)(b+2) \int_{\mathcal{L}} \frac{(\nu(l)-\mu(l))^2}{\nu(l)+\mu(l)} \,\mathrm{d}\lambda \\ &\overset{(c)}{\leq} (c'+2) c \log n; \end{aligned}$$

here in step (a) we use the fact that due to the membership swap, exactly $n-2$ intra-cluster pairs in θ_0 become inter-cluster pairs in θ_j and vise-versa, step (b) follows from the first equation in Lemma 24, and step (c) holds due to the assumption (10) of the theorem and that b is bounded by a universal constant.

The result then follows from taking c sufficiently small and applying Theorem 2.5 of Tsybakov (2009).

Appendix C. Proof of Theorem 6

In this section we prove the monotonicity property in Theorem 6. Let $E = \{l \in \mathcal{L} : \bar{\mu}(l) \geq \bar{\nu}(l)\}$ and $E^c = \mathcal{L} \setminus E$. Since (μ,ν) is strictly more divergent than $(\bar{\mu},\bar{\nu})$, we have that for all $l \in E$, $\mu(l) \geq \bar{\mu}(l) \geq \bar{\nu}(l) \geq \nu(l)$ and for all $l \in E^c$, $\nu(l) \geq \bar{\nu}(l) > \bar{\mu}(l) \geq \mu(l)$.

Suppose that the label matrix L is generated using the following two-stage procedure:

1. First, generate a matrix $\bar{L}$ from $(\bar{\mu},\bar{\nu})$. Set $L \leftarrow \bar{L}$.

2. Second:

   - For each (i,j) where $Y^*_{ij} = 0$, if $L_{ij} \in E$, then with probability $1 - \frac{\nu(L_{ij})}{\bar{\nu}(L_{ij})}$, set $L_{ij} \leftarrow l$ where l is drawn from the set E^c with distribution $\frac{\nu(l)-\bar{\nu}(l)}{\int_{l' \in E^c} \nu(l')-\bar{\nu}(l') \,\mathrm{d}\lambda(l')}$. Let Ω_- be the set of all such entries, i.e. where L_{ij} has switched from E to E^c.
   - For each (i,j) where $Y^*_{ij} = 1$, if $L_{ij} \in E^c$, then with probability $1 - \frac{\mu(L_{ij})}{\bar{\mu}(L_{ij})}$, set $L_{ij} \leftarrow l$ where l is drawn from the set E with distribution $\frac{\mu(l)-\bar{\mu}(l)}{\int_{l' \in E} \mu(l')-\bar{\mu}(l') \,\mathrm{d}\lambda(l')}$. Let Ω_+ be the set of all such entries, i.e. where L_{ij} has switched from E^c to E.

It is straightforward to verify that the resulting distribution of L is identical to that generated by the pair (μ,ν).

Consider the program (3) with $\bar{L}$ as input, and let $\bar{W}$ be the corresponding MLE weights. Since the pair $(\bar{\mu},\bar{\nu})$ satisfies the condition of Theorem 3, we have that with probability at least $1-n^{-10}$, the matrix Y^* is the unique optimal solution and hence satisfies $\langle \bar{W}, Y^* \rangle > \langle \bar{W}, Y \rangle$ for any feasible solution $Y \neq Y^*$. Now, consider the program (3) with L as the input, using the corresponding MLE

weights W based on $(\bar{\mu},\bar{\nu})$. We have that for any feasible $Y \neq Y^*$,

$$\begin{aligned} \langle W, Y^* \rangle - \langle \bar{W}, Y^* \rangle &= \sum_{(i,j)\in\Omega_+} (W_{ij} - \bar{W}_{ij}) Y^*_{ij} \\ &\geq \sum_{(i,j)\in\Omega_+} (W_{ij} - \bar{W}_{ij}) Y_{ij} \\ &\geq \sum_{(i,j)\in\Omega_-} (W_{ij} - \bar{W}_{ij}) Y_{ij} + \sum_{(i,j)\in\Omega_+} (W_{ij} - \bar{W}_{ij}) Y_{ij} \\ &= \langle W, Y \rangle - \langle \bar{W}, Y \rangle, \end{aligned}$$

which implies that

$$\langle W, Y^* \rangle - \langle W, Y \rangle \geq \langle \bar{W}, Y^* \rangle - \langle \bar{W}, Y \rangle > 0.$$

Therefor, Y^* is still the unique optimal solution.

Appendix D. Proof of Theorem 7

In this section, we prove Theorem 7 for using inaccurate weights. Our strategy is to apply Theorem 2. To avoid confusion we use b' and c' to denote the constants b and c in Theorem 2.

To show that the condition (5) in Theorem 2 is satisfied, we upper bound its right hand side as follows:

$$\begin{aligned} c' \frac{b' \log n + \sqrt{K \log n} \sqrt{\mathrm{Var}_\nu w}}{K} &\overset{(a)}{\leq} \frac{c' b' (1-\gamma)^2}{c\alpha^2 (b+2)} D(\nu\|\mu) + \sqrt{3c'^2 \alpha^2 (b+2) D(\nu\|\mu) \frac{\log n}{K}} \\ &\leq \frac{c'(1-\gamma)^2}{c\alpha} D(\nu\|\mu) + \sqrt{\frac{3c'^2(1-\gamma)^2}{c} D(\nu\|\mu)^2} \\ &\overset{(b)}{\leq} (1-\gamma) D(\nu\|\mu) \\ &\overset{(c)}{\leq} D(\nu\|\mu) - \Delta_\nu \\ &= -\mathbb{E}_\nu w, \end{aligned}$$

where in step (a) we use $\mathrm{Var}_\nu w \leq \alpha^2 \mathrm{Var}_\nu w^{\mathrm{MLE}}$ due to the condition $|w| \leq \alpha |\log \frac{\mu}{\nu}|$ and apply the bound (44), step (b) holds by choosing an appropriately large c, and in step (c) we use the assumption $|\Delta_\nu| \leq \gamma D(\nu\|\mu)$ in the statement Theorem 7.

We can use similar arguments, using the bound (45) instead of (44), to prove that the condition (6) in Theorem 2 is satisfied. Theorem 7 then follows from applying Theorem 2.

Appendix E. Proof of Corollary 10

In this section, we prove Corollary 10 for clustering Gaussian graphs. Ideally we would like apply Theorem 3 as we are using the MLE weight. A minor technical difficulty is that the boundedness condition $|w^{\mathrm{MLE}}(L_{ij})| \leq b$ is not satisfied as the Gaussian entries of L are unbounded. To overcome this we use a standard truncation argument.

Without loss of generality assume that $\underline{u} = 0$. Define a truncated version $\tilde{w}$ for the weight function w^{MLE} by

$$\tilde{w}(l) = \begin{cases} 0, & \text{if } |l| > c'\sqrt{\log n} \text{ or } l = ?, \\ l - (\bar{u} + \underline{u})/2, & \text{if } |l| < c'\sqrt{\log n}, \end{cases}$$

where $c' > 4c_0$ is a universal constant to be chosen later. With the goal of applying Theorem 2 to this weight function, we verify that the conditions of the theorem are satisfied. By the assumption $\bar{u} \le \underline{u} + c_0\sqrt{\log n} = c_0\sqrt{\log n}$, the weight function $\tilde{w}$ satisfies $\tilde{w}(l) \le b = (c' + c_0)\sqrt{\log n}, \forall l \in \mathcal{L}$, so the boundedness condition holds. We next verify the condition (6). Letting $\phi(x) = (2\pi)^{-1/2}e^{-x^2/2}$ be the density function of the standard normal, we note that

$$\begin{aligned} \mathbb{E}_\mu \tilde{w} = \int_{l:|l|<c'\sqrt{\log n}} w^{\text{MLE}}(l)\,\mathrm{d}\mu &= \int_{\mathcal{L}} w^{\text{MLE}}(l)\,\mathrm{d}\mu - \int_{l:|l|>c'\sqrt{\log n} \text{ or } l=?} w^{\text{MLE}}(l)\,\mathrm{d}\mu \\ &= \frac{s\bar{u}}{2} - s\int_{|l|>c'\sqrt{\log n}} (l - \bar{u}/2)\phi(l-\bar{u})\,\mathrm{d}l \\ &\ge \frac{s\bar{u}}{2} - s\int_{|l|>c'\sqrt{\log n}} (|l| + \bar{u}/2)\phi(l-\bar{u})\,\mathrm{d}l \\ &\ge \frac{s\bar{u}}{2} - 4s\underbrace{\int_{c'\sqrt{\log n}/2}^{+\infty} l\phi(l)\,\mathrm{d}l}_{T}, \end{aligned}$$

where the last inequality follows from the fact that $\bar{u} \le c_0\sqrt{\log n} < \frac{c'\sqrt{\log n}}{4}$. We control the term T as

$$T = (2\pi)^{-\frac{1}{2}} \int_{c'\sqrt{\log n}/2}^{+\infty} xe^{-x^2/2}\,\mathrm{d}x \overset{(a)}{\le} (2\pi)^{-\frac{1}{2}} \int_{c'\sqrt{\log n}/2}^{+\infty} e^{-(x-1/2)^2/2}\,\mathrm{d}x \overset{(b)}{\le} \frac{1}{n^2} \overset{(c)}{\le} \frac{\bar{u}}{16},$$

where the step (a) follows from $x \le e^{x-1/2}$, (b) follows from the standard Gaussian tail bound $1 - \Phi(t) \le e^{-t^2/2}$ and the fact that c' is sufficiently large, and (c) follows from the assumption (18) of Corollary 10. It follows that $\mathbb{E}_\mu \tilde{w} \ge \frac{s\bar{u}}{4}$. On the other hand, since $\tilde{w}$ is a truncated version of w^{MLE}, we have

$$\max\{\text{Var}_\mu \tilde{w}, \text{Var}_\nu \tilde{w}\} \le \max\{\text{Var}_\mu w^{\text{MLE}}, \text{Var}_\nu w^{\text{MLE}}\} \le s(\bar{u}^2 + 1) \le 2c_0^2 s \log n.$$

Combining the above bounds, it is easy to check that the condition (6) of Theorem 2 is satisfied under the assumption (18) of Corollary 10. Similar lines of arguments establish the condition (5) of Theorem 2. This theorem therefore guarantees that the program (2) with the weight function $w = \tilde{w}$ recovers the true cluster matrix Y^* with probability at least $1 - n^{-10}$.

Now, by choosing the constant c' to be sufficiently large and using the Gaussian tail bound and the union bound, we are guaranteed that with probability at least $1 - n^{-10}$, $\tilde{w}(L_{ij}) = w^{\text{MLE}}(L_{ij})$ for all (i, j). In the intersection of the above two events (which occurs with probability at least $1 - 2n^{-10}$), the program (2) with the weight function $\tilde{w}$ is identical to that with the weight function w^{MLE}, both of which recover the true Y^*. The same holds for the program (3).

Appendix F. Proof of Corollary 15

In this section we prove Corollary 15 for clustering with Markov snapshots. The MLE weight in this case is given by

$$\begin{aligned} w^{\text{MLE}}(\bar{L}_{ij}) =& \frac{1}{T}\log\frac{\mu_0(L_{ij}^{(1)})\mu(L_{ij}^{(2)}|L_{ij}^{(1)})\ldots\mu(L_{ij}^{(T)}|L_{ij}^{(T-1)})}{\nu_0(L_{ij}^{(1)})\nu(L_{ij}^{(2)}|L_{ij}^{(1)})\ldots\nu(L_{ij}^{(T)}|L_{ij}^{(T-1)})} \\ =& \frac{1}{T}\log\frac{\mu_0(L_{ij}^{(1)})}{\nu_0(L_{ij}^{(1)})} + \frac{1}{T}\sum_{t=2}^{T}\log\frac{\mu(L_{ij}^{(t)}|L_{ij}^{(t-1)})}{\nu(L_{ij}^{(t)}|L_{ij}^{(t-1)})}. \end{aligned}$$

In the sequel, we will focus on an in-cluster pair (i, j) with label distribution μ and drop the subscript ij in L_{ij}. The same analysis holds for the cross-cluster pair (i, j).

It is convenient to consider an auxiliary Markov chain $X_1, \ldots X_T$ where each state is characterized by a label pair $X_t = (L^{(t-1)}, L^{(t)})$ for $t > 1$, and $X_1 = L^{(1)}$. We define the function f on the domain of X_t such that

$$f(L) = \log\frac{\mu_0(L)}{\nu_0(L)} \quad \text{and} \quad f(L, L') = \log\frac{\mu(L'|L)}{\nu(L'|L)}.$$

We therefore have

$$w^{\text{MLE}}(\bar{L}) = \frac{1}{T}\sum_{t=1}^{T} f(X_t).$$

It is straightforward to show that

$$\mathbb{E}_\mu(f(X_1)) = D(\mu_0\|\nu_0) \quad \text{and} \quad \mathbb{E}_\mu(f(X_t)) = \mathbb{E}_{\mu_0} D_l(\mu\|\nu) \ \text{ for } t > 1,$$

whence

$$\mathbb{E}_\mu(w^{\text{MLE}}) = \frac{1}{T}D(\mu_0\|\nu_0) + \Big(1 - \frac{1}{T}\Big)\mathbb{E}_{\mu_0} D_l(\mu\|\nu). \tag{46}$$

The rest of the proof concerns bounding $\text{Var}_\mu(w^{\text{MLE}})$. Following the proof of Lemma 22, the variance of $f(X_t)$ can be bounded by

$$\text{Var}_\mu(f(X_1)) \le 3(b+2)D(\mu_0\|\nu_0) \quad \text{and} \quad \text{Var}_\mu(f(X_t)) \le 3(b+2)\mathbb{E}_{\mu_0} D_l(\mu\|\nu) \quad (t > 1).$$

We now bound the covariance $\text{Cov}_\mu(f(X_t), f(X_{t+\tau+1}))$ for $t \ge 2$ and $\tau \ge 0$:

$$\begin{aligned} &\text{Cov}_\mu(f(X_t), f(X_{t+\tau+1})) \\ =& \mathbb{E}_\mu\left[\log\frac{\mu(L^{(t)}|L^{(t-1)})}{\nu(L^{(t)}|L^{(t-1)})}\log\frac{\mu(L^{(t+\tau+1)}|L^{(t+\tau)})}{\nu(L^{(t+\tau+1)}|L^{(t+\tau)})}\right] - \mathbb{E}_{\mu_0} D_l(\mu\|\nu)^2 \\ =& \sum_{L^{(t-1)}} \mu_0(L^{(t-1)}) \sum_{L^{(t)}} \mu(L^{(t)}|L^{(t-1)}) \log\frac{\mu(L^{(t)}|L^{(t-1)})}{\nu(L^{(t)}|L^{(t-1)})} \sum_{L^{(t+\tau)}} \Pr_\mu(L^{(t+\tau)}|L^{(t)}) D_{L^{(t+\tau)}}(\mu\|\nu) \\ &- \mathbb{E}_{\mu_0} D_l(\mu\|\nu)^2 \\ \overset{(a)}{\le}& \sum_{L^{(t-1)}} \mu_0(L^{(t-1)}) \sum_{L^{(t)}} \mu(L^{(t)}|L^{(t-1)}) \left|\log\frac{\mu(L^{(t)}|L^{(t-1)})}{\nu(L^{(t)}|L^{(t-1)})}\right| \sum_{L^{(t+\tau)}} \kappa\phi^\tau D_{L^{(t+\tau)}}(\mu\|\nu) \\ \le& \frac{\kappa\phi^\tau}{\min_l \mu_0(l)} b\mathbb{E}_{\mu_0} D_l(\mu\|\nu), \end{aligned}$$

where in the step (a) we use the geometric ergodicity assumption of μ, i.e.,

$$|\Pr_\mu(L^{(t+\tau)}|L^{(t)}) - \mu_0(L^{(t+\tau)})| \le \kappa\phi^\tau.$$

The same bound also applies to the case $t = 1$. Note that the covariance bound is independent of t and only dependent on τ.

We proceed to bound $\mathrm{Var}_\mu(w^{\mathrm{MLE}})$ as follows:

$$\begin{aligned}
\mathrm{Var}_\mu(w^{\mathrm{MLE}}) &= \frac{1}{T^2}\sum_{t=1}^{T}\mathrm{Var}_\mu(f(X_t)) + \frac{2}{T^2}\sum_{t=1}^{T-1}\sum_{t'=t+1}^{T}\mathrm{Cov}_\mu(f(X_t), f(X_{t'})) \\
&\le \frac{1}{T^2}\sum_{t=1}^{T}\mathrm{Var}_\mu(f(X_t)) + \frac{2}{T}\sum_{\tau=0}^{T-2}\frac{T-1-\tau}{T}\frac{\kappa\phi^\tau}{\min_l \mu_0(l)} b\mathbb{E}_{\mu_0} D_l(\mu\|\nu) \\
&\le \frac{3(b+2)}{T}\left(\frac{1}{T}D(\mu_0\|\nu_0) + \frac{T-1}{T}\mathbb{E}_{\mu_0} D_l(\mu\|\nu)\right) + \frac{2\kappa}{(1-\phi)\min_l \mu_0(l)} b\frac{\mathbb{E}_{\mu_0} D_l(\mu\|\nu)}{T} \\
&\le c\frac{(b+2)\Phi}{T}\left(\frac{1}{T}D(\mu_0\|\nu_0) + \frac{T-1}{T}\mathbb{E}_{\mu_0} D_l(\mu\|\nu)\right). \qquad (47)
\end{aligned}$$

With the above bounds (46) and (47), we complete the proof of Corollary 15 by applying Theorem 2.

Appendix G. Example of Markov Chain with Explicit Bound on Φ

The snapshots in the Markov model are not independent. Therefore, given T snapshots we do not expect a T-fold increase in the information as in the independent snapshot model. In the conditions given in Corollary 15, this penalty is characterized by the parameter Φ defined in Section 4.4. To provide a sense of what values it may take, we now derive an explicit bound for a simple class of 2-state sequences (i.e., $|\mathcal{L}| = 2$).

As in Section F, we first focus on an in-cluster pair (i, j) with label distribution μ, and drop the subscript ij in L_{ij}. Suppose the transition matrix for the distribution μ is

$$\begin{bmatrix} 1-p_0 & p_0 \\ p_1 & 1-p_1 \end{bmatrix},$$

where $0 < p_0, p_1 < 1$. If we identify the label set $\mathcal{L}$ with $\{\text{edge}, \text{non-edge}\}$, then p_0 can be thought of as the probability that an edge "flips" into a non-edge, and p_1 as the probability that a non-edge flips into an edge. Let $\rho := 1 - p_0 - p_1$. By eigen-decomposition we can show that the transition matrix after t transitions is

$$\begin{bmatrix} \frac{p_1}{p_0+p_1}\left(1 + \frac{p_0}{p_1}\rho^t\right) & \frac{p_0}{p_0+p_1}(1-\rho^t) \\ \frac{p_1}{p_0+p_1}\left(1-\rho^t\right) & \frac{p_0}{p_0+p_1}(1 + \frac{p_1}{p_0}\rho^t) \end{bmatrix},$$

and the stationary distribution on the two states, written as a vector, is given by

$$\mu_0 = \begin{bmatrix} \dfrac{p_1}{p_0+p_1} & \dfrac{p_0}{p_0+p_1} \end{bmatrix}.$$

This Markov chain of the observed sequence of an in-cluster pair satisfies the inequality

$$|\Pr_\mu(L^{(1+t)}|L^{(1)}) - \mu_0(L^{(1+t)})| \le |\rho|^t$$

for all integer $t \ge 1$.

Now suppose that the cross-cluster distribution ν is of a similar form as μ, i.e., has the transition matrix

$$\begin{bmatrix} 1-p_0' & p_0' \\ p_1' & 1-p_1' \end{bmatrix},$$

for some p_0' and p_1'. By similar arguments, ν has the stationary distribution $\nu_0 = [p_1'/(p_0' + p_1'), p_0'/(p_0' + p_1')]$, and satisfies $|\Pr_\nu(L^{(1+t)}|L^{(1)}) - \nu_0(L^{(1+t)})| \le |\rho'|^t$, where $\rho' := 1 - p_0' - p_1'$. Therefore, the geometric ergodicity condition in (23) of Section 4.4 holds with parameters $\kappa = 1$ and $\phi = \max\{|\rho|, |\rho'|\}$. The parameter Φ, having the value

$$\Phi = \frac{1}{\left(1 - \max\{|\rho|, |\rho'|\}\right)\min\left\{\frac{p_0}{p_0+p_1}, \frac{p_1}{p_0+p_1}, \frac{p_0'}{p_0'+p_1'}, \frac{p_1'}{p_0'+p_1'}\right\}}$$

is a constant independent of the parameters n, K, T etc.

The value of Φ determines how much new information is contained in a new snapshot. For example, suppose that $p_1 = p_0 \in (0, \frac{1}{2}]$ and $p_1' = p_0' \in (0, \frac{1}{2}]$, in which case the stationary distributions $\mu_0 = \nu_0 = [\frac{1}{2}\ \frac{1}{2}]$ are fixed, and hence

$$\Phi = \frac{1}{\min\{p_0, p_0'\}}.$$

The value of Φ is large when the flipping probabilities p_0, and p_0' are close to zero. In this case the next snapshot is almost always the same as the current snapshot, hence providing little extra information. On the other hand, if these probabilities are closer to $\frac{1}{2}$, say $p_0 = p_1 = \frac{1}{2}$ and $p_0' = p_1' = \frac{1}{4}$, then $\Phi = 4$ is small. In this case the snapshots have more independence and thus the next snapshot provides fresh information.

Also note that in the above case, the marginal distributions of in- and cross- cluster pairs are the same: $\mu_0 = \nu_0 = [\frac{1}{2}\ \frac{1}{2}]$. In this case, information about the clustering only comes from the "flipping" pattern in the snapshots. For example, if $p_0 = p_1 = \frac{1}{2}$ and $p_0' = p_1' = \frac{1}{4}$, then in a long sequence of snapshots, the fractions of "edge" and "non-edge" labels in a in-cluster pair and a cross-cluster pair are both close to 50%, but we will see more flippings on pairs in the same cluster.

References

Brendan P. W. Ames. Guaranteed clustering and biclustering via semidefinite programming. *Mathematical Programming*, 147(1–2):429–465, 2014. ISSN 0025-5610.

Brendan P. W. Ames and Stephen Vavasis. Nuclear norm minimization for the planted clique and biclique problems. *Mathematical Programming*, 129(1):69–89, 2011.

Anima Anandkumar, Rong Ge, Daniel Hsu, and Sham M. Kakade. A tensor spectral approach to learning mixed membership community models. *Journal of Machine Learning Research*, 15: 2239–2312, June 2014.

Sivaraman Balakrishnan, Mladen Kolar, Alessandro Rinaldo, Aarti Singh, and Larry Wasserman. Statistical and computational tradeoffs in biclustering. In *NIPS Workshop on Computational Trade-offs in Statistical Learning*, 2011.

Afonso S. Bandeira and Ramon van Handel. Sharp nonasymptotic bounds on the norm of random matrices with independent entries. *Annals of Probability*, 44(4):2479–2506, 2016.

Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56(1), 2004.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, January 2011.

Deepayan Chakrabarti, Ravi Kumar, and Andrew Tomkins. Evolutionary clustering. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 554–560. ACM, 2006.

Kamalika Chaudhuri, Fan Chung, and Alexander Tsiatas. Spectral clustering of graphs with general degrees in the extended planted partition model. In *Proceedings of the 25th Annual Conference on Learning Theory (COLT)*, pages 35.1–35.23, 2012.

Yudong Chen, Ali Jalali, Sujay Sanghavi, and Huan Xu. Clustering partially observed graphs via convex optimization. *Journal of Machine Learning Research*, 15:2213–2238, June 2014a.

Yudong Chen, Shiau Hong Lim, and Huan Xu. Weighted graph clustering with non-uniform uncertainties. In *International Conference on Machine Learning*, 2014b.

Yudong Chen, Sujay Sanghavi, and Huan Xu. Improved graph clustering. *IEEE Transactions on Information Theory*, 60(10):6440–6455, 2014c.

Yuxin Chen, Changho Suh, and Andrea J. Goldsmith. Information Recovery from Pairwise Measurements: A Shannon-Theoretic Approach. *ArXiv e-prints*, April 2015.

Yun Chi, Xiaodan Song, Dengyong Zhou, Koji Hino, and Belle L. Tseng. On evolutionary spectral clustering. *ACM Transactions on Knowledge Discovery from Data*, 3(4):17:1–17:30, December 2009.

Anne Condon and Richard M. Karp. Algorithms for graph partitioning on the planted partition model. *Random Structures and Algorithms*, 18(2):116–140, 2001.

Nathan Eagle and Alex (Sandy) Pentland. Reality mining: Sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268, March 2006. ISSN 1617-4909.

Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, 2010.

Wenjie Fu, Le Song, and Eric P. Xing. Dynamic mixed membership blockmodel for evolving networks. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 329–336, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1.

Mathieu Genois, Christian L. Vestergaard, Julie Fournet, Andre Panisson, Isabelle Bonmarin, and Alain Barrat. Data on face-to-face contacts in an office building suggest a low-cost vaccination strategy based on community linkers. *Network Science*, 3:326–347, September 2015.

Ryan G. Gomes, Peter Welinder, Andreas Krause, and Pietro Perona. Crowdclustering. In *Advances in Neural Information Processing Systems*, 2011.

Qiuyi Han, Kevin S. Xu, and Edoardo M. Airoldi. Consistent estimation of dynamic and multi-layer block models. In *International Conference on Machine Learning*, 2015.

Simon Heimlicher, Marc Lelarge, and Laurent Massoulié. Community detection in the labelled stochastic block model. In *NIPS Workshop on Algorithmic and Statistical Approaches for Large Social Networks*, 2012.

Paul W. Holland, Kathryn B. Laskey, and Samuel Leinhardt. Stochastic blockmodels: Some first steps. *Social Networks*, 5:109–137, 1983.

Varun Jog and Po-Ling Loh. Information-theoretic bounds for exact recovery in weighted stochastic block models using the Renyi divergence. *ArXiv e-prints*, September 2015.

Vikas Kawadia and Sameet Sreenivasan. Sequential detection of temporal communities by estrangement confinement. *Scientific Reports*, 2, 2012.

Mladen Kolar, Sivaraman Balakrishnan, Alessandro Rinaldo, and Aarti Singh. Minimax localization of structural information in large noisy matrices. In *Advances in Neural Information Processing Systems*, pages 909–917, 2011.

Marc Lelarge, Laurent Massoulié, and Jiaming Xu. Reconstruction in the Labeled Stochastic Block Model. In *IEEE Information Theory Workshop*, Seville, Spain, September 2013. URL `http://hal.inria.fr/hal-00917425`.

Shiau Hong Lim, Yudong Chen, and Huan Xu. Clustering from labels and time-varying graphs. In *Advances in Neural Information Processing Systems*, pages 1188–1196, 2014.

Claire Mathieu and Warren Schudy. Correlation clustering with noisy input. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms*, page 712, 2010.

Catherine Matias and Vincent Miele. Statistical clustering of temporal networks through a dynamic stochastic block model. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 2016.

Frank McSherry. Spectral partitioning of random graphs. In *Proceedings of 42nd IEEE Symposium on Foundations of Computer Science*, pages 529–537, 2001.

Nam P. Nguyen, Thang N. Dinh, Ying Xuan, and My T. Thai. Adaptive algorithms for detecting community structure in dynamic social networks. In *INFOCOM*, pages 2282–2290. IEEE, 2011.

Samet Oymak and Babak Hassibi. Finding dense clusters via low rank + sparse decomposition. arXiv:1104.5186v1, 2011.

Benjamin Recht, Maryam Fazel, and Pablo A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review*, 52(471), 2010.

Karl Rohe, Sourav Chatterjee, and Bin Yu. Spectral clustering and the high-dimensional stochastic block model. *Annals of Statistics*, 39:1878–1915, 2011.

Ohad Shamir and Naftali Tishby. Spectral clustering on a budget. In *AISTATS*, 2011.

Juliette Stehl, Nicolas Voirin, Alain Barrat, Ciro Cattuto, Lorenzo Isella, Jean-Franois Pinton, Marco Quaggiotto, Wouter Van den Broeck, Corinne Rgis, Bruno Lina, and Philippe Vanhems. High-resolution measurements of face-to-face contact patterns in a primary school. *PLOS ONE*, 6(8):e23176, 08 2011.

Jimeng Sun, Christos Faloutsos, Spiros Papadimitriou, and Philip S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2007.

Flemming Topsoe. Some inequalities for information divergence and related measures of discrimination. *IEEE Transactions on Information Theory*, 46(4):1602–1609, Jul 2000.

Joel A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009.

Ramya Korlakai Vinayak, Samet Oymak, and Babak Hassibi. Graph clustering with missing data: Convex algorithms and analysis. In *Advances in Neural Information Processing Systems*, pages 2996–3004, 2014.

Kevin S. Xu and Alfred O. Hero. Dynamic stochastic blockmodels for time-evolving social networks. *IEEE Journal of Selected Topics in Signal Processing*, 8(4):552–562, Aug 2014.

Jinfeng Yi, Rong Jin, Anil K. Jain, and Shaili Jain. Crowdclustering with sparse pairwise labels: A matrix completion approach. In *AAAI Workshop on Human Computation*, 2012.

Journal of Machine Learning Research 18 (2017) 1-41 Submitted 3/16; Revised 3/17; Published 5/17

Uniform Hypergraph Partitioning: Provable Tensor Methods and Sampling Techniques

Debarghya Ghoshdastidar debarghya.g@csa.iisc.ernet.in
Ambedkar Dukkipati ad@csa.iisc.ernet.in
Department of Computer Science & Automation
Indian Institute of Science
Bangalore - 560012, India

Editor: Edo Airoldi

Abstract

In a series of recent works, we have generalised the consistency results in the stochastic block model literature to the case of uniform and non-uniform hypergraphs. The present paper continues the same line of study, where we focus on partitioning weighted uniform hypergraphs—a problem often encountered in computer vision. This work is motivated by two issues that arise when a hypergraph partitioning approach is used to tackle computer vision problems:
(i) The uniform hypergraphs constructed for higher-order learning contain all edges, but most have negligible weights. Thus, the adjacency tensor is nearly sparse, and yet, not binary.
(ii) A more serious concern is that standard partitioning algorithms need to compute all edge weights, which is computationally expensive for hypergraphs. This is usually resolved in practice by merging the clustering algorithm with a tensor sampling strategy—an approach that is yet to be analysed rigorously.

We build on our earlier work on partitioning dense unweighted uniform hypergraphs (Ghoshdastidar and Dukkipati, ICML, 2015), and address the aforementioned issues by proposing provable and efficient partitioning algorithms. Our analysis justifies the empirical success of practical sampling techniques. We also complement our theoretical findings by elaborate empirical comparison of various hypergraph partitioning schemes.

Keywords: Hypergraph partitioning, planted model, spectral method, tensors, sampling, subspace clustering

1. Introduction

Over several decades, the study of networks or graphs has played a key role in analysing relational data or pairwise interactions among entities. While networks often arise naturally in social or biological contexts, there are several machine learning algorithms that construct graphs to capture the similarity among data instances. A classic example of this approach is the spectral clustering algorithm of Shi and Malik (2000) that performs image segmentation by partitioning a graph constructed on the image pixels, where the weighted edges capture the visual similarity of the pixels. In general, graph partitioning and related problems are quite popular in unsupervised learning (Ng et al., 2002; Cour et al., 2007), dimensionality reduction (Zhao and Liu, 2007), semi-supervised learning (Belkin et al., 2004) as well as transductive inference (Wang et al., 2008). In spite of the versatility of the graph based approaches, these methods are often incapable of handling complex networks that involve multi-way interactions. For instance, consider a market transaction database, where each transaction or purchase corresponds to a multi-way connection among commodities involved in the transaction (Guha et al., 1999). Such networks do not conform with a traditional graph structure, and need to be modelled as hypergraphs. Similar multi-way interactions have been considered in the case of molecular interaction networks (Michoel and Nachtergaele, 2012), VLSI circuits (Karypis and Kumar, 2000), tagged social networks (Ghoshal et al., 2009), categorical databases (Gibson et al., 2000), computer vision (Agarwal et al., 2005) among others. In this work, we consider the network clustering problem for hypergraphs, where all the edges are of same cardinality. Uniform hypergraph partitioning finds use in computer vision applications such as subspace clustering (Agarwal et al., 2005; Rota Bulo and Pelillo, 2013), geometric grouping (Govindu, 2005; Chen and Lerman, 2009) or higher-order matching (Duchenne et al., 2011).

Uniform hypergraphs have been in the limelight of theoretical research for more than a century with the problem of hypergraph colorability surfacing in early 20^{th} century (Bernstein, 1908) to recent works establishing sharp phase transitions in random hypergraphs (see references in Bapst and Coja-Oghlan, 2015). However, there has been much less interest in studying practical machine learning problems that deal with uniform hypergraphs. For instance, restricting our discussion to the network partitioning, one can immediately notice a sharp contrast in the theoretical understanding of the problem in the context of graph and hypergraphs. Spectral graph partitioning algorithms have been analysed from different perspectives since the works of McSherry (2001) and Ng et al. (2002). The seminal work of Rohe et al. (2011), which studied standard spectral clustering under the stochastic block model, drew the attention of both statisticians and computer scientists, and has led to significant advancements in understanding of when the partitions can be detected, and which algorithms achieve optimal error rates (see Abbe and Sandon, 2016, for the current state of the art). In contrast, a similar line of study in the context of hypergraphs is quite recent (Ghoshdastidar and Dukkipati, 2014, 2015b, 2017; Florescu and Perkins, 2016).

Florescu and Perkins (2016) recently solved the problem of optimally detecting two equal-sized partitions in a planted unweighted hypergraph governed by a single parameter. On the other hand, the primary focus of our works has been to analyse the consistency of hypergraph partitioning approaches used in practice under a more general planted model. In Ghoshdastidar and Dukkipati (2014), we presented a generic planted model for dense uniform hypergraphs, and analysed the tensor decomposition based clustering algorithm of Govindu (2005) under this model. Subsequently, in Ghoshdastidar and Dukkipati (2015b), we found that a wide class of so-called "higher order" clustering algorithms can be unified by a common framework of tensor trace maximisation, which is quite similar in spirit to the associativity maximisation problem posed in the case of graph partitioning (Shi and Malik, 2000). We further proposed to solve a relaxation of the problem using a simple, spectral scheme that was consistent, and achieved better error rates compared to our previously studied approach. An extension of the whole setting to the case of sparse non-uniform hypergraphs came next on our agenda (Ghoshdastidar and Dukkipati, 2017), and we proved consistency of a spectral approach for non-uniform hypergraph partitioning.

©2017 Debarghya Ghoshdastidar and Ambedkar Dukkipati.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-100.html.

Like graphs, sparsity turns out to be an important characteristics of real-world hypergraphs. While this fact complicates analysis of the algorithms (see Ghoshdastidar and Dukkipati, 2017; Florescu and Perkins, 2016), it definitely provides significant computational relief. For instance, it is easy to realise that for any network clustering scheme, the computational complexity is at least linear in the number of edges. Hence, for a m-uniform hypergraph on n vertices, any standard approach should have a $O(n^m)$ runtime unless the hypergraph is sparse. This is precisely the problem that one encounters in vision applications, where the network is not given a priori, but one constructs a weighted hypergraph using m-way similarities among data instances. Thus, one needs to spend $O(n^m)$ runtime to construct the entire adjacency tensor only to realise at the end that only few edges have significant weights, and will aid the partitioning scheme. This scenario motivates the study in our present work, where we allow the planted hypergraph to have weighted edges, and still be sparse (in the sense that most weights are close to zero). But, at the same time, the non-zero entries are not known a priori, and hence, efficient schemes are required to perform the partitioning by observing only a small subset of the $O(n^m)$ edge weights.

1.1 Contributions in this Paper

We build on our earlier work. To be precise, we study the approach presented in Ghoshdastidar and Dukkipati (2015b), which solves a relaxation of the tensor trace maximisation (TTM) problem that lies at the heart of a variety of higher order learning methods. On the other hand, the model under consideration is that of sparse planted uniform hypergraph similar to the one studied in Ghoshdastidar and Dukkipati (2017). However, unlike previous works, we do not restrict the edge weights to be binary, but arbitrary random variables lying in the interval $[0, 1]$. So, the sparsity parameter in our model reduces the mean edge weights, leading to a large amount of edges with negligibly small weights, and hence, creating computational challenges of identifying significant edges. The planted model is formally described in Section 2, while our spectral approach is briefly recapped in Section 3. It might come as a surprise to many that this work does not make use the wide range of tensor decomposition techniques that have now become standard tools in machine learning. In Section 2, we discuss in detail how our model violates the common structural assumptions used in the tensor literature.

Our first contribution is presented in Section 4, where we analyse the basic TTM approach under the above mentioned planted model for weighted m-uniform hypergraphs. We note that in Ghoshdastidar and Dukkipati (2015b), we had studied the problem only in the dense unweighted case, whereas similarity hypergraphs encountered in subspace clustering etc. are weighted and typically have large number of insignificant edges (sparse). Furthermore, we recall that spectral partitioning methods, for graphs or hypergraphs, typically require a final step of distance based clustering. While k-means is the practical choice at this stage, theoretical studies even in the block model literature often use alternative schemes that are easy to analyse. Adhering to our goal of studying practical methods, our analysis utilises guarantees of k-means algorithm (Ostrovsky et al., 2012) instead of resorting to standard assumptions (Lei and Rinaldo, 2015) or other schemes (Gao et al., 2015). In this general setting, Theorem 2 presents the error rate of the TTM approach under mild restrictions on sparsity. Furthermore, recent results of Florescu and Perkins (2016) for the

special case of bi-partitioning suggests that our analysis is nearly optimal. We also show that the performance of this method is similar to the normalised hypergraph cut approach studied in Ghoshdastidar and Dukkipati (2017), and is superior than tensor decomposition based partitioning of Ghoshdastidar and Dukkipati (2014).

The second and key contribution in this paper is the analysis of a sampled variant of the TTM approach given in Section 5. As noted above, any basic partitioning scheme would have a $\Omega(n^m)$ runtime merely due to the construction of the entire adjacency tensor. We consider a scenario where only N edges are randomly sampled (with replacement) according to some pre-specified distribution. Theorem 9 provides a lower bound on the sample size N so that the sampled variant achieves desired error rate. The proof of this result borrows ideas from matrix sampling techniques (Drineas et al., 2006), but mainly relies on a trick of rephrasing the problem such that matrix Bernstein inequality can be applied. The analysis provides quite striking conclusions. For instance, under a simplified setting, if hypergraph is dense and consists of a constant number of partitions, then it is sufficient to observe only $\Omega(n(\ln n)^2)$ uniformly sampled edges. For sparse hypergraphs, uniform sampling cannot improve upon the $\Omega(n^m)$ runtime, but a certain choice of sampling distribution works with only $N = \Omega(n(\ln n)^2)$. To the best of our knowledge, this is the first work that analyses graph / hypergraph partitioning with sampling, and such sampling rates have not been previously observed in any related tensor problem. Typically, most methods need to observe about $\Omega(n^{m/2})$ entries of the tensor in order to estimate its decomposition (see, for instance, Bhojanapalli and Sanghavi, 2015; Jain and Oh, 2014), whereas we find that much less observations are required for the purpose of clustering. Our analysis also justifies the popularity of the iterative sampling schemes (Chen and Lerman, 2009; Jain and Govindu, 2013) in the higher order clustering literature.

Our final contribution is purely algorithmic. We present an iteratively sampled variant of the TTM algorithm, and conduct an extensive numerical comparison of various methods in the context of both hypergraph partitioning and subspace clustering. Section 6 presents a wide variety of empirical studies that (i) validate our theoretical findings regarding relative merits of TTM over previously studied algorithms, (ii) compare spectral methods to other hypergraph partitioning algorithms, including popular hMETIS tool (Karypis and Kumar, 2000), and (iii) weigh the merits of hypergraph partitioning in subspace clustering applications, including benchmark problems. Such empirical studies, though often seen in the subspace clustering works, was long overdue in the higher order learning literature. We also hope that the implementations will help standardising subsequent studies in this direction.[1]

We also note here that to achieve clarity of presentation, the sections only contain the outline of proofs of the main results. The proofs of the intermediate lemmas and corollaries are given in the appendix that follows after the concluding section (Section 7).

1.2 Notations

We conclude this section by stating the standard terminology and notations that we follow in the rest of the paper. We denote tensors in bold faces ($\mathbf{A}, \mathbf{B}$ etc.), matrices in capitals (A, Z etc.), while vectors and scalars will be understood from the context. We use Trace($\cdot$)

1. Codes are available at: `http://sml.csa.iisc.ernet.in/SML/code/Feb16TensorTraceMax.zip` and also on the personal webpage of first author.

to write the sum of diagonal entries of a matrix or tensor, and $\|\cdot\|_2$ for Euclidean norm for vector and the spectral norm for matrix. For a matrix, say A, $A_{i\cdot}$ (or $A_{\cdot i}$) represents its i^{th} row (column), $\|A\|_F$ denotes its Frobenius norm, and $\lambda_i(A)$ is the i^{th} largest eigenvalue or singular value of A (depending on context). We also use the standard $O(\cdot)$, $o(\cdot)$ and $\Omega(\cdot)$ notations, where, unless specified otherwise, the corresponding quantities are viewed as function of n. In addition, $\mathbb{1}\{\cdot\}$ is the indicator function, and $\ln(\cdot)$ is natural logarithm.

Moreover, the results in this paper consider two sources of randomness—the random model for hypergraph, and random sampling of edges (or tensor entries). We make this distinction in the notation for expectation, variance and probability by specifying the underlying measure. For instance, $\mathsf{E}_H[\cdot]$ is expectation with respect to distribution of the planted model, and $\mathsf{E}_{S|H}[\cdot]$ is the expectation over sampling distribution conditioned on a given random hypergraph. Similar subscripts have been used for probability, $\mathsf{P}(\cdot)$, and variance, $\mathsf{Var}(\cdot)$. Note that for a matrix A, $\mathsf{E}_H[A]$ refers to its entry-wise expectation.

2. Formal Description of the Problems

We consider the following random model. Let $\mathcal{V} = \{1, 2, \dots n\}$ be a set of n nodes, and $\psi : \{1, 2, \dots, n\} \to \{1, 2, \dots, k\}$ be a (hidden) partition of the nodes into k classes. For a node i, we denote its class by ψ_i. For a fixed integer $m \geq 2$, let $\alpha_n \in [0, 1]$, and $\mathbf{B} \in [0, 1]^{k \times k \times \dots \times k}$ be a symmetric k-dimensional tensor of order m. Let $\mathcal{E}$ be the collection of all subsets of $\mathcal{V}$ of size m. A random weighted m-uniform hypergraph $(\mathcal{V}, \mathcal{E}, w)$ is generated through the random function $w : \mathcal{E} \to [0, 1]$ such that

$$\mathsf{E}[w(\{i_1, i_2, \dots, i_m\})] = \alpha_n \mathbf{B}_{\psi_{i_1} \psi_{i_2} \dots \psi_{i_m}} \tag{1}$$

for all $e = \{i_1, i_2, \dots, i_m\} \in \mathcal{E}$, and the collection of random variables $(w(e))_{e \in \mathcal{E}}$ are mutually independent. For convenience, we henceforth write w_e instead of $w(e)$. The above model extends the planted partition model for graphs (McSherry, 2001), and is a weighted variant of the planted uniform hypergraph model studied in our earlier works. In particular, if $\alpha_n = 1$ and w_e are independent Bernoulli random variables satisfying (1), then one retrieves the model of Ghoshdastidar and Dukkipati (2014). We present our results for weighted hypergraphs due to their extensive use in computer vision (e.g., Agarwal et al., 2005). Note that the edge distributions are governed by $\mathbf{B}$ and ψ, and hence, depend only on the class membership. In addition, α_n accounts for sparsity of the hypergraph. Under this setting, the objective of a hypergraph partitioning algorithm is to estimate ψ from a given random instance of the m-uniform hypergraph $(\mathcal{V}, \mathcal{E}, w)$. Throughout this paper, we are interested in bounding the error incurred by a partitioning algorithm, defined as a multi-class 0-1 loss

$$\mathsf{Err}(\psi, \psi') = \min_{\sigma} \sum_{i=1}^{n} \mathbb{1}\{\psi_i \neq \sigma(\psi'_i)\}\,, \tag{2}$$

where ψ and ψ' denote the true and the estimated partitions, respectively, and σ is any permutation on $\{1, 2, \dots, k\}$. Note that we also allow number of classes k to grow with n.

The first part of the paper builds on the tensor trace maximisation method of Ghoshdastidar and Dukkipati (2015b), and we prove statistical consistency of TTM under the above sparse planted partition model. To be precise, we show that under certain conditions

on α_n and $\mathbf{B}$, this method achieves $\mathsf{Err}(\psi, \psi') = o(n)$. In the block model terminology (Mossel et al., 2013), this statement implies that the algorithm is weakly consistent. Furthermore, if the hypergraph is dense ($\alpha_n = 1$), then we show that TTM can exactly recover the partitions, *i.e.*, $\mathsf{Err}(\psi, \psi') = o(1)$, and hence, exhibits strong consistency properties. From the recent work of Florescu and Perkins (2016), which studies the special case of bi-partitioning, one can see that our restrictions on α_n are nearly optimal (upto a difference of $(\ln n)^2$) in the case of bi-partitioning as one cannot detect partitions for sparser hypergraphs.

Next we study partitioning algorithms that compute weights of only N out of $\binom{n}{m}$ edges. For the theoretical analysis, we assume that there is a known probability mass function $(p_e)_{e \in \mathcal{E}}$, and N edges are sampled with replacement from this distribution. We are interested in finding the minimum N that guarantees weak consistency of the sampled TTM approach. We focus on two sampling distributions: (i) uniform sampling, and (ii) weighted sampling where $p_e \propto w_e$. Surprisingly, we see that if $\alpha_n = 1$ (dense case), only $N = \Omega(nk^{2m-1}(\ln n)^2)$ edges are sufficient for either sampling strategies. This leads to a drastic improvement in runtime, particularly if $k = O(1)$, and even in general, since typically k grows much slower than n. However, if α_n decays rapidly with n, then more samples are needed for the case of uniform sampling, whereas the alternative strategy still works for $N = \Omega(nk^{2m-1}(\ln n)^2)$. A comparison with tensor literature is not very meaningful, but known methods of the latter field typically need to observe $\Omega(n^{m/2})$ tensor entries (Bhojanapalli and Sanghavi, 2015). In practice, however, computing the weighted sampling distribution requires a single pass over the adjacency tensor, which still takes $O(n^m)$ time. But, we argue that practical iterative schemes essentially approximate this distribution without observing the entire tensor.

2.1 A Look at Alternative (or Possible) Approaches

This may be a good time to reflect on the history of both hypergraphs and tensors with the focus of understanding the theoretical or practical tools that either fields provide for the planted uniform hypergraph problem. Before proceeding further, it will be helpful to take a look at the adjacency tensor of the random (planted) hypergraph.

Let $\mathbf{A} \in [0, 1]^{n \times n \times \dots \times n}$ be the symmetric adjacency tensor of order m. Let $Z \in \{0, 1\}^{n \times k}$ be the assignment matrix of the latent partition ψ, *i.e.*, $Z_{ij} = \mathbb{1}\{\psi_i = j\}$. Then we have

$$\mathbf{A}_{i_1 i_2 \dots i_m} = \begin{cases} \sum_{j_1, \dots, j_m = 1}^{k} \alpha_n \mathbf{B}_{j_1 j_2 \dots j_m} Z_{i_1 j_1} \dots Z_{i_m j_m} + \mathbf{E}_{i_1 i_2 \dots i_m} & \text{for distinct } i_1, \dots, i_m \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where $\mathbf{E}$ is a symmetric random tensor with zero mean entries. For ease of understanding, one may ignore the $O(n^{m-1})$ entries of $\mathbf{A}$ with repeated indices to write

$$\mathbf{A} \approx \alpha_n (\mathbf{B} \times_1 Z \times_2 Z \times_3 \dots \times_m Z) + \text{noise term}, \tag{4}$$

where $\times_l$ denotes the mode-l product between a tensor and a matrix (De Lathauwer et al., 2000).[2] The basic problem is to detect Z from a given $\mathbf{A}$.

2. Consider a matrix $B \in \mathbb{R}^{p \times n_l}$ and a m^{th}-order tensor $\mathbf{A} \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_m}$. The mode-$l$ product of $\mathbf{A}$ and B is a m^{th}-order tensor, represented as $\mathbf{A} \times_l B \in \mathbb{R}^{n_1 \times \dots n_{l-1} \times p \times n_{l+1} \times \dots \times n_m}$, whose elements are
$$(\mathbf{A} \times_l B)_{i_1 \dots i_{l-1} j i_{l+1} \dots i_m} = \textstyle\sum_{i_l} \mathbf{A}_{i_1 \dots i_{l-1} i_l i_{l+1} \dots i_m} B_{j i_l}.$$

The above form is quite similar to the representation of a tensor in terms of its higher order singular value decomposition or HOSVD (De Lathauwer et al., 2000). In fact, it shows that the random adjacency tensor has a multilinear rank approximately $k \ll n$, and clearly suggests that the partitioning problem should be viewed as a tensor decomposition problem. This hint was quickly picked up by Govindu (2005), who proposed a spectral approach for higher order clustering based on HOSVD. Long after this work, tensor methods have gained significant popularity in machine learning in recent years. However, only few works (Bhaskara et al., 2014; Anandkumar et al., 2015) consider decomposition of tensor into asymmetric rank-one terms, while most of the machine learning literature (Anandkumar et al., 2014; Ma et al., 2016) consider decomposition into symmetric rank-one terms. To be more precise, (3) suggests that

$$\mathbf{A} \approx \sum_{j_1,..,j_m=1}^{k} \alpha_n \mathbf{B}_{j_1 j_2 \ldots j_m} Z_{.j_1} \otimes Z_{.j_2} \otimes \ldots \otimes Z_{.j_m} + \text{noise},$$

where $\otimes$ is the tensor outer product. Clearly the k^m rank-one terms are asymmetric. It is well known that such a tensor can be represented by a symmetric outer product decomposition only in an algebraically closed field (Comon et al., 2008), *i.e.*, one can write as sum of symmetric rank-one terms, but the vectors in the decomposition are not guaranteed to be real, and hence, will be of little use. We note that though the works of Bhaskara et al. (2014) and Anandkumar et al. (2015) are applicable, their incoherence assumption is clearly violated in the present context where the same vectors appear in all m modes, and with multiplicity greater than one.

Under simpler settings such as the one described later in Section 4.1, one can express $\mathbf{A}$ as a sum of $(k+1)$ symmetric terms of the form

$$\mathbf{A} \approx \sum_{j=1}^{k} \alpha_n p Z_{.j}^{\otimes m} + \alpha_n q v^{\otimes m} + \text{noise}, \tag{5}$$

where $v = \sum_j Z_{.j}$ and p, q are parameters defining $\mathbf{B}$. Such tensors, which have a finite symmetric CP-rank (Candecomp/Parafac), have been extensively studied in machine learning. For a single rank-one term, optimal detection rates are known under Gaussian noise (Richard and Montanari, 2014). Though there is no distributional assumption on the noise term in our case, (1) does imply that the variance of the noise is smaller than the signal, *i.e.*, specialised to this setting, our model lies in the detection region. Unfortunately, single rank-one term occurs for $k = 1$, which does not correspond to any meaningful hypergraph problem, and hence, such results are of little use in our case. It may seem that the case of $k \geq 2$ can still be tackled using tensor power iteration based approaches (Anandkumar et al., 2014, 2015), but the necessary incoherence criterion is violated even here since v has a significant overlap with each $Z_{.j}$. To summarise, it suffices to say that existing guarantees in the tensor literature are not directly applicable for solving (5), but it does not rule out the possibility that a careful analysis of power iterations (Anandkumar et al., 2014) or alternative approaches (Ma et al., 2016) may lead to alternative partitioning techniques. That being said, one should note that (5) is merely a special case of (3), where HOSVD still appears to be the natural answer.

Interestingly, uniform hypergraphs predate the tensor literature, and one may refer to Berge (1984) for early development. Even hypergraph partitioning came into practice (Schweikert and Kernighan, 1979) before tensor decompositions gained popularity. However, initial approaches to hypergraph partitioning in VLSI (Karypis and Kumar, 2000) and database (Gibson et al., 2000) communities relied on clever combination of heuristics with no known performance guarantees. Subsequent works in computer vision (Agarwal et al., 2005) and machine learning (Zhou et al., 2007) proposed spectral solutions for the problem. Such approaches are more amenable for a theoretical analysis. While the analysis in Ghoshdastidar and Dukkipati (2017) and Florescu and Perkins (2016) are somewhat based on the hypergraph cut approach of Zhou et al. (2007), the algorithm studied in this paper is closely related to work of Agarwal et al. (2005). The key idea of such spectral schemes is to reduce the hypergraph into a graph and then apply spectral clustering. Quite surprisingly, we show in this paper that both schemes perform better than HOSVD based partitioning both theoretically (Remark 5) and numerically (Section 6.1). This is counter-intuitive since one would expect significant information loss during the reduction to a graph. A careful look at the algorithm presented in the next section would reveal that this is not the case. While most information in modes $3, \ldots, m$ are lost, one can still estimate Z from the first two modes, which suffices for the purpose of detecting planted partitions. This observation is reinforced by the recent study of Florescu and Perkins (2016), where the authors show that a reduction based spectral approach can optimally detect two partitions all the way down to the limit of identifiability of the partitions.

We conclude this section with a brief mention of the wide variety of other higher order learning methods, which include tensor based clustering algorithms (Shashua et al., 2006; Chen and Lerman, 2009; Arias-Castro et al., 2011; Ochs and Brox, 2012), other unsupervised tensorial learning schemes (Duchenne et al., 2011; Nguyen et al., 2015), as well as related optimisation approaches (Leordeanu and Sminchisescu, 2012; Rota Bulo and Pelillo, 2013; Jain and Govindu, 2013), and can easily be represented as instances of the uniform hypergraph partitioning problem. In Ghoshdastidar and Dukkipati (2015b), we showed that most of the above methods can be unified by a general *tensor trace maximisation* (TTM) problem. A spectral solution to this problem is analysed in the present paper, and thus, we believe that some of our conclusions can also be extended to these alternative approaches. We also numerically compare with some of these methods in Section 6.

3. Tensor Trace Maximisation (TTM)

In this section, we briefly recap the work in Ghoshdastidar and Dukkipati (2015b) and then present the basic approach that will be later analysed and modified in the remainder of the paper. Let $(\mathcal{V}, \mathcal{E}, w)$ be a given weighted uniform hypergraph, where $\mathcal{E}$ is the collection of all sets of m vertices, and $w : \mathcal{E} \to [0, 1]$ associates a weight with every edge. We consider the problem of partitioning $\mathcal{V}$ into k disjoint sets, $\mathcal{V}_1, ..., \mathcal{V}_k$, such that the total weight of edges within each cluster is high, and the partition is 'balanced'. In the case of graphs, *i.e.*, for $m = 2$, the popular heuristic for achieving these two conditions is the normalised cut minimisation problem, or equivalently normalised associativity maximisation problem (Shi and Malik, 2000; von Luxburg, 2007). We extend this approach to uniform hypergraphs.

3.1 TTM Approach and Algorithm

We consider the problem of finding the partition of the vertices that maximises the normalised associativity. This is subsequently formulated in terms of a tensor trace maximisation objective. We define few terms. The degree of any node $v \in \mathcal{V}$ is the total weight of edges on which v is incident, *i.e.*, $\deg(v) = \sum_{e\in\mathcal{E}:v\in e} w_e$. For any collection of nodes $\mathcal{V}_1 \subseteq \mathcal{V}$, we define its volume as $\mathrm{vol}(\mathcal{V}_1) = \sum_{v\in\mathcal{V}_1} \deg(v)$ and its associativity as $\mathrm{assoc}(\mathcal{V}_1) = \sum_{e\in\mathcal{E}:e\subset\mathcal{V}_1} w_e$, which is the total weight of edges contained within $\mathcal{V}_1$. The normalised associativity of a partition $\mathcal{V}_1, ..., \mathcal{V}_k$ is given as

$$\text{N-Assoc}(\mathcal{V}_1, \ldots, \mathcal{V}_k) = \sum_{i=1}^{k} \frac{\mathrm{assoc}(\mathcal{V}_i)}{\mathrm{vol}(\mathcal{V}_i)} \,. \tag{6}$$

Observe that the above definitions coincide with the corresponding terms in graph literature (Shi and Malik, 2000). We now follow the popular goal of finding clusters that maximises the normalised associativity (6). In the case of graphs, it is well known that the problem can be reformulated in terms of the adjacency matrix of the graphs, which results in a matrix trace maximisation problem (von Luxburg, 2007). Furthermore, a spectral relaxation allows one to find an approximate solution for the problem by computing the k dominant eigenvectors of the normalised adjacency matrix.[3] A similar approach is possible in the case of uniform hypergraphs. Let $\mathbf{A}$ be the adjacency tensor (of order m), *i.e.*,

$$\mathbf{A}_{i_1 i_2 \ldots i_m} = \begin{cases} w_{\{i_1,i_2,\ldots,i_m\}} & \text{if } i_1, i_2, \ldots, i_m \text{ are distinct,} \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Define $\beta_1, ..., \beta_m \in [0, 1]$ with $\sum_{l=1}^{m} \beta_l = 1$, and $Y^{(1)}, \ldots, Y^{(m)} \in \mathbb{R}^{n\times k}$ with $Y_{ij}^{(l)} = \left(\frac{\mathbb{1}\{i\in\mathcal{V}_j\}}{\mathrm{vol}(\mathcal{V}_j)}\right)^{\beta_l}$. Then, one can rewrite (6) (see appendix for details) as

$$\text{N-Assoc}(\mathcal{V}_1, \ldots, \mathcal{V}_k) = \frac{1}{m!}\mathrm{Trace}\left(\mathbf{A} \times_1 Y^{(1)^T} \times_2 Y^{(2)^T} \times_3 \ldots \times_m Y^{(m)^T}\right) \,, \tag{8}$$

where $\times_l$ denotes the mode-l product. Thus, for some chosen parameters $\beta_1, \ldots, \beta_m$, one can pose the associativity maximisation problem as a tensor trace maximisation (TTM).

In Ghoshdastidar and Dukkipati (2015b), we showed that the above optimisation has connections with the tensor eigenvalue problem (Lim, 2005) and the tensor diagonalisation problem (Comon, 2001). More interestingly, it also lies at the heart of several higher-order learning algorithms. For instance, $\beta_1 = \ldots = \beta_m = \frac{1}{m}$ results in the method of Shashua et al. (2006), while the same strategy when used with $k = 1$ has been used to successively extract clusters (Rota Bulo and Pelillo, 2013; Leordeanu and Sminchisescu, 2012). A similar idea lies in some tensor matching algorithms (Duchenne et al., 2011; Nguyen et al., 2015). Another strategy is to set $\beta_1 = \beta_2 = \frac{1}{2}$ and $\beta_3 = \ldots = \beta_m = 0$. This squeezes the tensor into a matrix, which allows one to use subsequently graph partitioning tools. Our algorithm described below takes this route, and similar ideas have been previously used (Agarwal et al., 2005; Arias-Castro et al., 2011). This reduction also corresponds to the clique expansion of a hypergraph, where every m-way edge is replaced by $\binom{m}{2}$ pairwise edges.

3. Dominant eigenvectors are the orthonormal eigenvectors corresponding to the largest k eigenvalues.

We list our basic spectral approach in Algorithm TTM, and we later study the consistency of TTM and its sampled variants. Note that a spectral relaxation of the problem is a two-fold procedure, where first we construct a matrix from the affinity tensor $\mathbf{A}$, and then relax the problem into a matrix spectral decomposition type objective. This principle is reminiscent of the classical technique for studying spectral properties of hypergraphs (Bolla, 1993), and is closely related to approach of the clustering graph approximations of hypergraphs (Agarwal et al., 2006).

Algorithm TTM : Spectral relaxation of tensor trace maximisation problem

Input: Affinity tensor $\mathbf{A}$ of the m-uniform hypergraph $(\mathcal{V}, \mathcal{E}, w)$, where $|\mathcal{V}| = n$.

1: Read $\mathbf{A}$, and compute the $n \times n$ matrix A as $A_{ij} = \sum_{i_3,\ldots,i_m=1}^{n} \mathbf{A}_{iji_3\ldots i_m}$.

2: Let $D \in \mathbb{R}^{n\times n}$ be diagonal with $D_{ii} = \sum_{j=1}^{n} A_{ij}$, and $L = D^{-1/2}AD^{-1/2}$.

3: Compute k dominant eigenvectors of L, denoted by $X \in \mathbb{R}^{n\times k}$.

4: Normalise rows of X to have unit norm, and denote this matrix as $\overline{X}$.

5: Run k-means on the rows of $\overline{X}$.

Output: Partition of $\mathcal{V}$ that correspond to the clusters obtained from k-means.

A careful look at computational complexity of the algorithm will be helpful for our discussions in Section 5. To this end, we note that our analysis assumes the use k-means approach of Ostrovsky et al. (2012), which has a complexity of $O(k^2n + k^4)$ since the data is embedded in a k-dimensional space. Furthermore, Steps 2 to 4 involve only matrix operations with the eigenvector computation being the most expensive operation. One may compute the k dominant eigenvectors using power iterations, which can be done provably in $O(kn^2 \ln(kn))$ runtime (Boutsidis et al., 2015). However, the computational bottleneck of the algorithm is Step 1, which has complexity of $m^2|\mathcal{E}| = O(m^2n^m)$. This is not surprising since any network partitioning method should have have complexity at least linear in the number of edges. But it gets quite challenging in computer vision problems, where one often requires to consider higher order relations, for instance $m = 4$ used in Duchenne et al. (2011), $m = 5$ in Chen and Lerman (2009) and even $m = 8$ in Govindu (2005). The aim of Section 5 is to reduce this complexity to m^2N by sampling only $N \ll n^m$ edges.

4. Consistency of Algorithm TTM under Planted Partition Model

The first task in our agenda is to analyse the basic TTM approach. This section extends the results in Ghoshdastidar and Dukkipati (2015b) to the the planted partition model for sparse weighted hypergraphs described in Section 2. Recall that our aim is to derive an upper bound on $\mathsf{Err}(\psi, \psi')$, where ψ and ψ' denote the true and the estimated clusters. Moreover, for the purpose of analysis we assume that the k-means step is performed using Lloyd's approach with the seeding described in (Ostrovsky et al., 2012). The reason for this consideration is the known theoretical guarantee for this method.

We briefly recall the planted model for weighted m-uniform hypergraphs described in Section 2. An underlying function ψ groups the n nodes into k clusters, and ψ_i denotes the true cluster of node i. For any edge $e = \{i_1, i_2, \ldots, i_m\}$, its weight w_e is a random variable taking values in $[0,1]$ with mean given by (1), where the parameters $\alpha_n \in [0,1]$ and symmetric m-way tensor $\mathbf{B} \in [0,1]^{k \times k \times \ldots \times k}$, respectively, govern the mean edge weight and the relative weights of edges formed among nodes from different classes.[4] For instance, in Section 4.1, we consider an example where given $p, q \in [0,1]$, the tensor $\mathbf{B}$ is constructed such that $\mathsf{E}[w_e] = \alpha_n(p+q)$ if all nodes in the edge belong to the same cluster, and $\mathsf{E}[w_e] = \alpha_n q$ if the participating nodes are from different clusters. Thus, in this case, the mean weight of edges residing within in each cluster is larger than inter-cluster edge weights. In addition to above, we also assume that all edge weights $(w_e)_{e \in \mathcal{E}}$ are mutually independent.

Consider a random m-uniform hypergraph $(\mathcal{V}, \mathcal{E}, w)$ generated according to the above model. As a consequence of (1), the expected affinity tensor of the hypergraph

$$\mathsf{E}_H[\mathbf{A}_{i_1 i_2 \ldots i_m}] = \begin{cases} \alpha_n \mathbf{B}_{\psi_{i_1} \psi_{i_2} \ldots \psi_{i_m}} & \text{if } i_1, i_2, \ldots, i_m \text{ are distinct, and} \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

has a block structure, ignoring entries with repeated indices. Obviously, the k^m blocks are aligned with the underlying clusters, which gives rise to the representation mentioned in (3). Algorithm TTM first squeezes the adjacency tensor $\mathbf{A}$ to a $n \times n$ matrix A. To analyse the algorithm in the expected case, let $\mathcal{A} = \mathsf{E}_H[A]$ and $\mathcal{D} = \mathsf{E}_H[D]$, where A and D are the matrices computed in Algorithm TTM. Observe that if the algorithm had access to the expected affinity tensor (9), then $\mathcal{A}$ corresponds to the matrix computed in the first step of the algorithm, and $\mathcal{D}_{ii} = \sum_{j=1}^{n} \mathcal{A}_{ij}$. From the definition of the model, it can be seen that $\mathcal{A}_{ii} = 0$ for all i, and for $i \neq j$,

$$\mathcal{A}_{ij} = (m-2)! \sum_{\substack{i_3 < i_4 < \ldots < i_m, \\ i,j \notin \{i_3, \ldots, i_m\}}} \alpha_n \mathbf{B}_{\psi_i \psi_j \psi_{i_3} \ldots \psi_{i_m}} , \tag{10}$$

where the factor $(m-2)!$ takes into account all permutations of $\{i_3, \ldots, i_m\}$. The key observation here is that $\mathcal{A}_{ij} = \mathcal{A}_{i'j'}$ whenever $\psi_i = \psi_{i'}$ and $\psi_j = \psi_{j'}$, which holds since, under the present model, nodes in the same cluster are statistically identical. Thus, one can define a matrix $G \in \mathbb{R}^{k \times k}$ such that $\mathcal{A}_{ij} = G_{\psi_i \psi_j}$ for all $i \neq j$. This implies that, ignoring the diagonal entries, $\mathcal{A}$ is essentially of rank k.

Let $Z \in \{0,1\}^{n \times k}$ be the assignment matrix corresponding to partition ψ, *i.e.*, $Z_{ij} = \mathbb{1}\{i \in \psi(j)\}$, and let the sizes of the k clusters be $n_1 \geq n_2 \geq \ldots \geq n_k$. We define

$$\delta = \lambda_k(G) \min_{1 \leq i \leq n} \frac{n_{\psi_i}}{\mathcal{D}_{ii}} - \max_{1 \leq i,j \leq n} \left| \frac{G_{\psi_i \psi_i}}{\mathcal{D}_{ii}} - \frac{G_{\psi_j \psi_j}}{\mathcal{D}_{jj}} \right| , \tag{11}$$

where $\lambda_k(G)$ is the smallest eigenvalue of G. The following lemma, proved in the appendix, shows that if $\delta > 0$, then Algorithm TTM correctly identifies the underlying clusters in the expected case.

4. Note that α_n plays the role of a sparsity parameter commonly introduced to define sparse stochastic block models (Lei and Rinaldo, 2015), and a smaller α_n increases the complexity of the problem. However, unlike the unweighted case, all edges are present in our setting but most of the weights are very small.

Lemma 1 *Let $\mathcal{L} = \mathcal{D}^{-1/2} \mathcal{A} \mathcal{D}^{-1/2}$. If δ in (11) satisfies $\delta > 0$, then there exists an orthonormal matrix $U \in \mathbb{R}^{k \times k}$ such that the k leading orthonormal eigenvectors of $\mathcal{L}$ correspond to the columns of the matrix $\mathcal{X} = Z(Z^T Z)^{-1/2} U$.*

It is easy to see that $\mathcal{X}$ has k distinct rows, each corresponding to a true cluster. Hence, clustering the rows of $\mathcal{X}$ (or its row normalised form) using k-means gives an accurate clustering of the nodes. In the random case, however, the dominant eigenvectors of L computed in TTM need not always reflect the true assignment matrix Z. The following result shows that under certain conditions on the model parameters, the eigenvectors are still close to $\mathcal{X}$, and hence, the number of mis-clustered nodes (2) grows slowly.

Theorem 2 *Let $(\mathcal{V}, \mathcal{E}, w)$ be a random m-uniform hypergraph on $|\mathcal{V}| = n$ vertices generated from the model described above. Define $d = \min_{1 \leq i \leq n} \mathsf{E}_H[\deg(i)]$ and, without loss of generality, assume that the cluster sizes are $n_1 \geq n_2 \geq \ldots \geq n_k$. Let δ be as defined in (11). There exists an absolute constant $C > 0$, such that, if $\delta > 0$ and*

$$\delta^2 d > \frac{C k n_1 (\ln n)^2}{n_k} \tag{12}$$

for all large n, then with probability $(1 - o(1))$, the partitioning error for TTM is

$$\mathsf{Err}(\psi, \psi') = O\left(\frac{k n_1 \ln n}{\delta^2 d} \right). \tag{13}$$

The bound in (13) along with the condition in (12) immediately suggests that TTM is weakly consistent, *i.e.*, $\mathsf{Err}(\psi, \psi') = o(n)$ or the fractional of mis-clustered vertices vanishes as $n \to \infty$. However, in certain (dense) cases, even $\mathsf{Err}(\psi, \psi') \to 0$ as we will discuss later. Note that d grows with n though this dependence is not made explicit in the notation. We also allow k to vary with n. The condition on $\delta^2 d$ in (12) ensures that the hypergraph is sufficiently dense so that the following three conditions hold, respectively: (i) the matrix A computed in Algorithm TTM concentrates near its expectation, (ii) the k dominant eigenvectors of L contain information about the partition, and (iii) the k-means step provides a near optimal solution. While restricting d from below in (12) essentially limits the sparsity of the hypergraph, the quantity δ on the other hand quantifies the complexity of the model.

The threshold for identifying two partitions, derived in Florescu and Perkins (2016), shows that the condition in (12) differs from the threshold for identifiability only by logarithmic factors. These extra $\ln n$ factors arise since (i) we consider weighted hypergraphs, and (ii) we do not substitute k-means by alternative strategies. Note that even works on stochastic block model (Lei and Rinaldo, 2015; Gao et al., 2015) do not consider these two factors, and hence, even in the case of graphs, it is not known till date whether the logarithmic terms can be avoided when these practical aspects are included in the analysis. We point out that the present analysis incorporates the guarantees for k-means derived in Ostrovsky et al. (2012), which was also used in our earlier work (see Lemma 4.8 of Ghoshdastidar and Dukkipati, 2017).

4.1 A Special Case

To gain insights into the implications of Theorem 2, we consider the following special case of the planted partition model. The partition ψ is defined such that the k clusters are of equal

size. Moreover, the tensor $\mathbf{B}$ in (1) is given by $\mathbf{B}_{j_1 j_2 \ldots j_m} = (p+q)$ if $j_1 = j_2 = \ldots = j_m$, and q otherwise, where $p, q \in [0,1]$ with $q \leq (1-p)$. Thus, in this model, edges residing within each cluster have a high weight (in the expected sense) as compared to other edges.[5] This model corresponds to the decomposition of $\mathbf{A}$ mentioned in (5). We state the following consistency result for dense hypergraphs.

Corollary 3 *Let $\alpha_n = 1$ and $k = O\left(\frac{n^{1/4}}{\ln n}\right)$. Then with probability $(1 - o(1))$,*

$$\mathsf{Err}(\psi, \psi') = O\left(\frac{n^{(3-m)/2}}{(\ln n)^{2m-3}}\right) . \tag{14}$$

According to the notions of consistency defined in Mossel et al. (2013), it can be seen that for $m = 2$, Algorithm TTM is weakly consistent, *i.e.*, $\mathsf{Err}(\psi, \psi') = o(n)$. We note here that, in this sense, the algorithm is not worse than spectral clustering that is also known to be weakly consistent (Rohe et al., 2011). However, for $m \geq 3$, $\mathsf{Err}(\psi, \psi') = o(1)$ for Algorithm TTM, which implies that it is strongly consistent in this case. In other words, the algorithm can exactly recover the partitions for large n. This conclusion is intuitively acceptable since in this case, uniform hypergraphs for large m have a large number of edges that provides 'more' information about the partition, providing a smaller error rate.

In the sparse regime, the question one is interested in is the minimum level of sparsity under which weak consistency of an algorithm can be proved. The following result answers this question. For the case of graphs ($m = 2$), Lei and Rinaldo (2015) showed that weak consistency is achieved by spectral clustering for $\alpha_n \geq \frac{C \ln n}{n}$, which matches our result upto a factor of $(\ln n)^2$. In fact, our proof also allows the difference to be reduced to a factor of $\omega(\ln n)$, but this difference has negligible effect in practice.

Corollary 4 *Let $k = O(\ln n)$. There exists an absolute constant $C > 0$, such that, if*

$$\alpha_n \geq \frac{C(\ln n)^{2m+1}}{n^{m-1}} , \tag{15}$$

then $\mathsf{Err}(\psi, \psi') = O\left(\frac{n}{(\ln n)^2}\right) = o(n)$ with probability $(1 - o(1))$.

While the stochastic block model has been extensively studied for graphs, the existing hypergraph literature provides consistency results for only two other approaches:

- a uniform hypergraph partitioning method that uses a higher order singular value decomposition (HOSVD) of the adjacency tensor (Govindu, 2005; Ghoshdastidar and Dukkipati, 2014), and
- a non-uniform hypergraph partitioning approach that solves a spectral relaxation of the normalised hypergraph cut (NH-Cut) problem (Zhou et al., 2007; Ghoshdastidar and Dukkipati, 2017). Florescu and Perkins (2016) also use a similar method.

5. The model considered here may be viewed as the four parameter stochastic block model (Rohe et al., 2011) defined by the parameters (n, k, p_n, q_n), where n nodes are divided into k partitions of equal size. Edges within a cluster occur with probability $(p_n + q_n)$, while inter-cluster edges occur with probability q_n. Here, we set $p_n = \alpha_n p$ and $q_n = \alpha_n q$ for some constants $p, q \in [0,1]$ with $q \leq (1-p)$.

We comment on the theoretical performance of TTM in comparison with these two approaches. In particular, we focus on the settings of Corollaries 3 and 4. The following remark is quite surprising since both TTM and NH-Cut reduce the hypergraphs to graphs, and hence, apparently incur some loss. Yet both outperform HOSVD, which appears to be the most natural solution according to the representation in (3). This fact is also validated numerically in Section 6.1.

Remark 5 *Under the setting of Corollary 3, the error bound for the NH-Cut algorithm is*

$$\mathrm{Err}_{\text{NH-Cut}}(\psi, \psi') = O\left(\frac{n^{(3-m)/2}}{(\ln n)^{2m-3}}\right)$$

with probability $(1 - o(1))$, while the corresponding bound for HOSVD algorithm is

$$\mathrm{Err}_{\text{HOSVD}}(\psi, \psi') = O\left(\frac{n^{(4-m)/2}}{(\ln n)^{2m-1}}\right) .$$

Thus the performance of NH-Cut is similar to TTM, and both methods have a smaller error bound than HOSVD.

Similarly, in the case of Corollary 4, the lower bound on sparsity for NH-Cut is same as in (15) *up to a constant scaling. However, HOSVD achieves weak consistency only for*

$$\alpha_n \geq \frac{C'(\ln n)^{m+1.5}}{n^{(m-1)/2}}$$

for some $C' > 0$. This is larger than the allowable sparsity for TTM or NH-Cut.

4.2 Proof of Theorem 2

Here, we give an outline of the proof of Theorem 2 using a series of technical lemmas. The proofs of these results are given in the appendix. The proof has a modular structure which consists of (i) deriving certain conditions on the model parameters such that Algorithm TTM incurs no error in the expected case, (ii) subsequent use of matrix concentration inequalities and spectral perturbation bounds to claim that (almost surely) the dominant eigenvectors in the random case do not deviate much from the expected case, and (iii) finally, the proof of correctness of the k-means step.

Recall that the first step of the proof is taken care of by Lemma 1. For convenience, define $\mathcal{D}_{\min} = \min_{1 \leq i \leq n} \mathcal{D}_{ii}$. One can see that $\mathcal{D}_{\min} = (m-1)!d$. Hence, for the subsequent analysis as well as for all the proofs, it is more convenient to expand (12) as

$$\mathcal{D}_{\min} > \max\left\{C_1 \ln n, \frac{C_2 \ln n}{\delta^2}, \frac{C_3 k n_1 \ln n}{n_k \epsilon^2 \delta^2}\right\} , \tag{16}$$

where the constants $C_1, C_2, C_3 > 0$ take into account the additional factor of $(m-1)!$. Note that the last term dominates, and corresponds to (12) when $\epsilon = (\ln n)^{-1/2}$. The subsequent results show that the eigenvector matrix X computed from a random realisation of the hypergraph is close to $\mathcal{X}$ almost surely, and hence, one can expect a good clustering in the random case.

Lemma 6 proves a concentration bound for the normalised affinity matrix L computed in Algorithm TTM. The proof, given in the appendix, relies on an useful characterisation of the matrix A. To describe this representation, we define for each edge $e \in \mathcal{E}$, a matrix $R_e \in \{0,1\}^{n\times n}$ as $(R_e)_{ij} = 1$ if $i,j \in e, i \neq j$, and zero otherwise. Quite similar to the representation of (10), one can note that

$$A = (m-2)! \sum_{e\in\mathcal{E}} w_e R_e \ . \tag{17}$$

This characterisation is quite useful since the independence of $(w_e)_{e\in\mathcal{E}}$ ensures that A is represented as a sum of independent random matrices, and hence, one can use matrix concentration inequalities (Tropp, 2012) to derive a tail bound for $\|A - \mathcal{A}\|_2$.

Lemma 6 *If there exists n_0 such that $\mathcal{D}_{\min} > 9(m-1)!\ln n$ for all $n \geq n_0$, then with probability $\big(1 - O(n^{-2})\big)$,*

$$\|L - \mathcal{L}\|_2 \leq 12\sqrt{\frac{(m-1)!\ln n}{\mathcal{D}_{\min}}} \ . \tag{18}$$

The above result directly leads to a bound on the perturbation of the eigenvectors as shown in Lemma 4.7 of Ghoshdastidar and Dukkipati (2017). The result adapted to our setting is stated below.

Lemma 7 *Assume there is an n_0 such that $\mathcal{D}_{\min} > 9(m-1)!\ln n$ and $\delta > 24\sqrt{\frac{(m-1)!\ln n}{\mathcal{D}_{\min}}}$ for all $n \geq n_0$. Then the following statements hold with probability $\big(1 - O(n^{-2})\big)$.*

1. *The matrix X does not have any row with zero norm, and hence, its row normalised form, denoted by $\overline{X}$, is well-defined.*
2. *There is an orthonormal matrix $Q \in \mathbb{R}^{k\times k}$ such that*

$$\big\|\overline{X} - ZQ\big\|_F \leq \frac{24}{\delta}\sqrt{\frac{(m-1)!2kn_1\ln n}{\mathcal{D}_{\min}}} \ . \tag{19}$$

Finally, we analyse the k-means step of the algorithm, where the rows of $\overline{X}$ are assigned to k centres. Define $S \in \mathbb{R}^{n\times k}$ such that $S_{i\cdot}$ denotes the centre to which $\overline{X}_{i\cdot}$ is assigned. Also define the collection of nodes $\mathcal{V}_{err} \subset \mathcal{V}$ such that

$$\mathcal{V}_{err} = \left\{ i \in \mathcal{V} : \|S_{i\cdot} - Z_{i\cdot}Q\|_2 \geq \frac{1}{\sqrt{2}} \right\} \ . \tag{20}$$

The following result, adapted from Ghoshdastidar and Dukkipati (2017), shows that on one hand $\mathcal{V}_{err}$ contains all the mis-labelled nodes, whereas, on the other, it proves that under the conditions of Theorem 2, the k-means algorithm of Ostrovsky et al. (2012) finds a near optimal solution for which $|\mathcal{V}_{err}|$ can be bounded from above.

Lemma 8 *Under the conditions stated in* (16)*, for small enough ϵ,*

$$\mathsf{Err}(\psi,\psi') \leq |\mathcal{V}_{err}| \leq 8(1+\epsilon^2)^2\|\overline{X} - ZQ\|_F^2 \tag{21}$$

with probability $\big(1 - O(n^{-2} + \sqrt{\epsilon})\big)$.

Theorem 2 follows by setting $\epsilon = (\ln n)^{-1/2}$, and using the bound on $\|\overline{X} - ZQ\|_F$ in (19).

5. Sampling Techniques for Algorithm TTM

We now present the second, and key, contribution in this work. Recall from the discussions in Section 3 that the overall computational complexity of TTM is $O(m^2n^m + kn^2\ln(kn) + k^2n + k^4)$, where the first term clearly dominates for $m \geq 3$. A practical solution to this problem is to simply compute the weights of few edges, or equivalently, sample few entries of the adjacency tensor $\mathbf{A}$. This strategy has often been used in computer vision, but to the best of our knowledge, there is no known theoretical study of the approach. The only relevant theoretical works (Bhojanapalli and Sanghavi, 2015; Jain and Oh, 2014) are in a different context, where the authors study factorisation of partially observed tensors. While the latter work assumes an uniform sampling, the former presents distributions that are more adapted to the tensor. We later compare these results with our findings.

In contrast, practical higher order learning methods exhibit considerable variety in sampling techniques. Govindu (2005) used a sampling that uniformly selects fibers of the tensor, which is similar in spirit to the well known column sampling technique for matrices. Ideas along the same lines, and also a Nyström approximation, for the HOSVD based approach were suggested in Ghoshdastidar and Dukkipati (2015a). A more efficient technique of iterating between sampling and clustering was used in Jain and Govindu (2013) and Chen and Lerman (2009), where one starts with a naive sampling to get approximate partitions and then iteratively improves the result by sampling edges aligned with partitions. Matching algorithms (Duchenne et al., 2011) exploit side information to prioritise edges with significant weights. Other heuristics have also been suggested in some works.

We formally study the following problem. Suppose we are given a certain distribution $(p_e)_{e\in\mathcal{E}}$ on the set of all m-way edges $\mathcal{E}$. Let N edges be sampled with replacement according to the given distribution. We aim to find the minimum sample size required such that corresponding partitioning algorithm (with edge sampling) is still weakly consistent. In this paper, we assume that the core partitioning approach is TTM, and the sampling only affects Step 1 of the algorithm, where we replace A by its sample estimate, denoted by $\widehat{A}$. A requirement of the estimator should be its unbiasedness, *i.e.*, $\mathsf{E}_{S|H}[\widehat{A}] = A$, where the expectation is with respect to the sampling distribution given an instance of the random hypergraph. Based on (17), we propose to use an unbiased estimator of the form

$$\widehat{A} = \frac{(m-2)!}{N}\sum_{e\in\mathcal{I}} \frac{w_e}{p_e} R_e \ , \tag{22}$$

where $\mathcal{I} \subset \mathcal{E}$ with $|\mathcal{I}| = N$ is the collection of sampled edges (with possible duplicates). The matrix $R_e \in \{0,1\}^{n\times n}$ is such that $(R_e)_{ij} = 1$ if $i,j \in e, i \neq j$, and zero otherwise. The overall method is listed below, and one can easily see that its runtime is $O(m^2N + kn^2\ln(kn) + k^2n + k^4)$.

5.1 Consistency of Sampled Variants of Algorithm TTM

We now analyse the performance of Algorithm Sampled TTM for any given edge sampling distribution $(p_e)_{e\in\mathcal{E}}$. The main message of the following result is that the algorithm remains weakly consistent even if we use very small number of sampled edges, *i.e.*, $N = o(n^m)$. However, the minimum sample size required depends on the sampling distribution.

Algorithm Sampled TTM : TTM where a sampled set of edge weights are observed

Input: Distribution $(p_e)_{e\in\mathcal{E}}$ on the set of all edges $\mathcal{E}$;
Affinity tensor $\mathbf{A}$, which is not observed, but requested entries can be observed.

1: Sample a collection of N edges $\mathcal{I} \in \mathcal{E}$ with replacement.
2: Observe entries of $\mathbf{A}$ corresponding to edges in $\mathcal{I}$, and compute $\widehat{A}$ using (22).
3: Run Steps 2-5 of Algorithm TTM using $\widehat{A}$ instead of A.

Output: Partition of $\mathcal{V}$ that correspond to the clusters obtained from k-means.

Theorem 9 *Let N edges be sampled with replacement according to probability distribution $(p_e)_{e\in\mathcal{E}}$, and let $\beta > 0$ be such that $\mathsf{P}_H\left(\max_{e\in\mathcal{E}} \frac{w_e}{p_e} > \beta\right) = o(1)$. Let δ be as defined in (11). There exist absolute constants $C, C' > 0$, such that, if $\delta > 0$,*

$$\delta^2 d > C\frac{kn_1(\ln n)^2}{n_k} \quad \text{and} \quad N > C'\left(1+\frac{2\beta}{d}\right)\frac{kn_1(\ln n)^2}{n_k\delta^2} \tag{23}$$

for all large n, then with probability $(1-o(1))$,

$$\mathsf{Err}(\psi,\psi') = O\left(\frac{kn_1\ln n}{\delta^2}\left(\frac{1}{d}+\frac{1}{N}+\frac{2\beta}{Nd}\right)\right) = o(n). \tag{24}$$

Note that the above probability is with respect to both the randomness of the hypergraph and edge sampling. The above result is similar to Theorem 2 except for the additional condition and error term associated with the number of sampled edges N. We mention here that the constant C in (23) is different from the one used in Theorem 2, but the quantity δ remains the same, and does not depend on the sampling distribution. Since, the above error rate is $o(n)$, we can immediately conclude that the lower bound on N in (23) is sufficient to ensure weak consistency of the sampled variant. The result also shows that if we fix a particular sampling strategy, then smaller N is needed for denser and easier models (large $\delta^2 d$). This can be explained since for sparse hypergraphs, most edges have zero or negligibly small weights, and do not provide 'sufficient information' about the true partition.

On the other hand, the sampling strategy plays a crucial role in the lower bound for N, but the dependence is only via the ratio of the edge weight to the sampling probability. We note that β is a high probability upper limit of this ratio,[6] and (23) suggests that a better sampling distribution is one for which β is smaller. To clarify this observation, we state the result for two particular sampling distributions: (i) uniform sampling, and (ii) sampling each edge e with probability proportional to its weight, *i.e.*,

$$p_e = \frac{w_e}{\sum_{e'\in\mathcal{E}} w_{e'}} \qquad \text{for all } e \in \mathcal{E}. \tag{25}$$

One can easily see that $\beta = \sum_e w_e$ in the latter case, whereas $\beta = \binom{n}{m}\max_e w_e$ for uniform sampling. For ease of exposition, we restrict ourselves to the special case described in Section 4.1, and demonstrate the effect of these distributions on sample size.

6. The probability is with respect to the randomness of the planted model, and arises since the edge weights are random.

Corollary 10 *Consider the setting described in Section 4.1. Define quantity ξ such that $\xi = 1$ for uniform sampling, and $\xi = \alpha_n$ for the weighted sampling of (25). There exist constants $C, C' > 0$, such that, if*

$$\alpha_n > C\frac{k^{2m-1}(\ln n)^2}{n^{m-1}} \quad \text{and} \quad N > C'\frac{\xi nk^{2m-1}(\ln n)^2}{\alpha_n}, \tag{26}$$

then $\mathsf{Err}(\psi,\psi') = o(n)$ with probability $(1-o(1))$.

For simplicity, let us start with the case where $k = O(1)$. Then one has the lower bound $N = \Omega\left(\frac{n(\ln n)^2}{\alpha_n}\right)$ for uniform sampling, and $N = \Omega(n(\ln n)^2)$. Thus, the both sampling techniques have similar performance in the dense case ($\alpha_n = 1$), the gap between the lower bounds increase when α_n decays with n. In fact, in the most sparse setting possible in (23), $\alpha_n = O(\frac{(\ln n)^2}{n^{m-1}})$ and so, uniform sampling works only when one samples $\Omega(n^m)$ edges.[7] But with weighted sampling, one still needs only $\Omega(n(\ln n)^2)$ edges.

Possibility of achieving consistency with such a low sample size is quite remarkable, and has not been yet observed in any other tensor problem. For instance, it may be argued that one can directly use the factorisation techniques for partially observed tensors studied in Jain and Oh (2014) and Bhojanapalli and Sanghavi (2015), to guarantee sampling rates derived in these works. We show here that this may not be a good strategy since such sampling can be often much larger than the rate derived in Corollary 10. We note here that the comparison is not entirely fair since, on one hand, we require only the clusters instead of the complete factorisation, whereas on the other hand, the results in related tensor sampling works are usually tied to an incoherence assumption that is violated in our setting.

For the comparison, we recall that in the setting of Corollary 10, $\mathbf{A}$ has an approximate CP-decomposition of rank $(k+1)$ as shown in (5). Furthermore, most works on tensors do not consider the case where entries decay with dimension, and so, we may assume $\alpha_n = 1$. The aforementioned works consider the problem of tensor factorisation, where the tensor is partly observed by means of some sampling. Jain and Oh (2014) show that to obtain an accurate tensor factorisation, it is suffice to observe $\Omega(k^5 n^{m/2}(\ln n)^4)$ uniformly sampled entries. Bhojanapalli and Sanghavi (2015) use a different sampling distribution, which essentially assigns more weight for larger entries quite similar to (25), and then prove a similar bound on the sample size (upto logarithmic factors). The key difference of such bounds with Corollary 10 is that the m in the exponential is tied to n in other works, whereas it is tied to k in our case—this improves efficiency significantly when k grows much slower than n, which is clearly the case in clustering.

We elaborate on this further by considering the extreme values of k and α_n possible in Corollaries 3 and 4, respectively. First, let $\alpha_n = 1$ and $k = O\left(\frac{n^{1/4}}{\ln n}\right)$, as in Corollary 3. Then both uniform and weighted sampling can guarantee weak consistency if $N = \Omega\left(n^{0.5m+0.75}(\ln n)^{3-2m}\right)$. In contrast, the sample size from (Jain and Oh, 2014) is $N = \Omega(n^{0.5m+1.25}(\ln n)^{-1})$, which is worse by a factor of about $\sqrt{n}$. Turning to the setting of Corollary 4, we have $k = O(\ln n)$ and let α_n be at its lower bound in (15). Then, the above

7. Note that this observation is only true for weighted hypergraphs, where a small α_n implies that most edges have very small, but positive, weights. On the other hand, unweighted hypergraphs with small α_n implies that only few edges are present, and sampling is not required if these edges are given a priori.

result shows that while uniform sampling is poor, by sampling significant edges frequently, one needs only $N = \Omega\left(n(\ln n)^{2m+1}\right)$ edges for consistent partitioning. In contrast, the weighted sampling of Bhojanapalli and Sanghavi (2015) still needs $\Omega(n^{0.5m}(\ln n)^7)$ samples, which is much larger.

5.2 Proof of Theorem 9

We will further discuss the implications and limitations of weighted sampling, but first, we provide an outline for the proof of Theorem 9. Recall that the sampled variant differs from core TTM algorithm only in the use of $\widehat{A}$ (22) instead of A. Let us define $\widehat{D}, \widehat{L}$ for this case corresponding to D, L. That is, $\widehat{D}_{ii} = \sum_j \widehat{A}_{ij}$ and $\widehat{L} = \widehat{D}^{-1/2}\widehat{A}\widehat{D}^{-1/2}$. Note that $\widehat{A}, \widehat{D}$ are both unbiased estimates of A, D. The proof follows the lines of the proof of Theorem 2. It is easy to see that the only difference is in Lemma 6, where instead of $\|L - \mathcal{L}\|_2$, we now need to compute a bound on $\|\widehat{L} - \mathcal{L}\|_2$. Observe that

$$\|\widehat{L} - \mathcal{L}\|_2 \leq \|\widehat{L} - L\|_2 + \|L - \mathcal{L}\|_2 \ ,$$

where the second term is bounded due to Lemma 6. We have the following bound for the first part, which can be derived using matrix Bernstein inequality.

Lemma 11 *For large n and under the conditions in* (23)*, with probability* $1 - o(1)$*,*

$$\|\widehat{L} - L\|_2 \leq 12\sqrt{\frac{\ln n}{N}\left(1 + \frac{2\beta}{d}\right)} \ . \tag{27}$$

This bound combined with Lemma 6 implies

$$\|\widehat{L} - \mathcal{L}\|_2 \leq 12\sqrt{\frac{\ln n}{d}} + 12\sqrt{\frac{\ln n}{N}\left(1 + \frac{2\beta}{d}\right)} \ . \tag{28}$$

Let us denote the above upper bound by γ_n. Then one can restate Lemmas 7 and 8 as follows.

Lemma 7*. If $\delta > 2\gamma_n$ for all large n, then with probability $(1 - o(1))$,

$$\left\|\overline{X} - ZQ\right\|_F \leq \frac{2\gamma_n\sqrt{2kn_1}}{\delta} \ . \tag{29}$$

Lemma 8*. If $\delta \geq 2\gamma_n\sqrt{\frac{8kn_1 \ln n}{n_k}}$, then with probability $1 - o(1)$,

$$\mathsf{Err}(\psi, \psi') = O\left(\frac{kn_1\gamma_n^2}{\delta^2}\right) \ . \tag{30}$$

Here, the stronger condition is required for the k-means error bound. Now, observe that we can bound γ_n^2 as

$$\gamma_n^2 \leq 288\left(\frac{\ln n}{d} + \frac{\ln n}{N}\left(1 + \frac{2\beta}{d}\right)\right).$$

The above bound immediately implies the error bound in (24), whereas the condition in Lemma 8* is satisfied if (23) holds. Thus, the claim of Theorem 9 follows.

5.3 TTM with Iterative Sampling

In this section, we discuss practical strategies for sampling. The purpose of the section is two-fold. We first relate the weighted sampling strategy (25) to heuristics used in practice. We then suggest a practical variant of Algorithm Sampled TTM to solve the problem of subspace clustering. Our experimental results in next section will validate the efficacy of this method in realistic problems.

We recall that Corollary 10 led to the conclusion that, in general, weighted sampling (25) achieves a runtime that is smaller than that of uniform sampling by a factor of α_n. It is obvious that specifying this distribution involves computing all edge weights, which in turn, requires a single pass over the adjacency tensor. Even the weighted sampling of Bhojanapalli and Sanghavi (2015) suffers from the same issue. However, even this is not acceptable in practice as a single pass also has computational complexity of $O(n^m)$, and hence, Sampled TTM based on (25) is mainly of theoretical interest. But our analysis leads to an important conclusion—*sample edges with larger weights more frequently.*

This is essentially the idea commonly used in most tensor based algorithms. In the case of matching algorithms, one uses an efficient nearest neighbour search to sample the larger tensor entries (Duchenne et al., 2011). On the other hand, the subspace clustering literature has acknowledged the idea of iterative sampling (Chen and Lerman, 2009; Jain and Govindu, 2013), where one uses an alternating strategy of finding clusters using a sampled set of edges, and then re-sampling edges for which at least $(m-1)$ nodes belong to a cluster. It is not hard to realise that both sampling techniques give higher preference to edges with large weights, and hence, as a consequence of Corollary 10, both methods are expected to perform better than uniform sampling. Thus Corollary 10 provides a theoretical justification for why such heuristics work, thereby answering an open question posed by Chen and Lerman (2009).

We now turn to the problem of designing a practical variant of TTM based on the above discussion. We use the conclusions of Corollary 10 and present an iterative version of Algorithm TTM for the purpose of subspace clustering. We henceforth refer to this algorithm as *tensor trace maximisation with iterative sampling*, or simply Tetris. An additional reason for presenting this algorithm is to address a paradoxical situation that arose in our previous work (Ghoshdastidar and Dukkipati, 2015b). While the theoretical results suggest that TTM perform better than HOSVD based techniques (Govindu, 2005; Chen and Lerman, 2009), experiments on large benchmark problems did not align with the same conclusion. We later realised that this disparity occurred because we had combined TTM with a naive sampling technique, but had compared with the practical iterative variant of HOSVD. The numerical comparisons in this paper using Tetris resolves this issue, and shows that TTM is indeed more favourable.

We now present Tetris for solving the subspace clustering problem (Soltanolkotabi et al., 2014). In this problem, one is given a collection of n points $Y_1, Y_2, \ldots, Y_n \in \mathbb{R}^{r_a}$ in an high dimensional *ambient* space. However, there are k subspaces, each of dimension at most $r < r_a$, such that one can represent Y_i as

$$Y_i = \widetilde{Y}_i + \eta_i \ ,$$

where $\widetilde{Y}_i$ lies in one of the k subspaces, and η_i is a noise term. The objective of a subspace clustering algorithm is to group $Y_1, \ldots, Y_n$ into k disjoint clusters such that each cluster corresponds to exactly one of the k low-dimensional subspaces. A hypergraph or tensor based

subspace clustering approach (Agarwal et al., 2005; Govindu, 2005) involves construction of a weighted m-uniform hypergraph such that $m \geq (r+2)$ and the weight of an edge $e = \{i_1, \ldots, i_m\}$ is given by

$$w_e = w(\{i_1, \ldots, i_m\}) = \exp\left(-\frac{f_r(Y_{i_1}, \ldots, Y_{i_m})}{\sigma^2}\right) . \tag{31}$$

Here, $f_r(\cdot)$ computes the error of fitting a r-dimensional subspace for the given m points, and σ is a scaling parameter. Different choices for $f_r(\cdot)$ has been considered in the literature based on Euclidean distance of points from the estimated subspace (Govindu, 2005; Jain and Govindu, 2013), polar curvature of the points (Chen and Lerman, 2009) among others. Chen and Lerman (2009) also proposed a heuristic for estimating σ at each iteration.

We present Algorithm Tetris for the subspace clustering problem. We fix the order of the tensor as $m = (r+2)$, and define $f_r(\cdot)$ in terms of polar curvature (see Equations 1-3 of Chen and Lerman, 2009). We also incorporate the convergence criteria and the estimation procedure for σ used by Chen and Lerman (2009), which are not explicitly stated below. Furthermore, to standardise with their approach, Tetris uses a one-sided degree normalisation and computes left singular vectors of the normalised adjacency matrix.

Algorithm Tetris : TTM with iterative sampling for subspace clustering

Input: Dataset $Y = [Y_1, \ldots, Y_n]$; k = Number of subspaces;
r = Maximum subspace dimension; and
c = A hyper-parameter controlling number of sampled edges ($N = nc$)

1: Set $m = r + 2$.
2: Uniformly sample c subsets of Y, each containing $(m-1)$ points.
3: Initialise $\widehat{A} \in \mathbb{R}^{n \times n}$ to a zero matrix.
4: **for** $j = 1$ to c **do**
5: Consider j^{th} subset of Y with the points $Y_{j_1}, \ldots, Y_{j_{m-1}}$.
6: **for** $i = 1$ to n **do**
7: Compute the weight w_e for the edge $e = \{Y_i, Y_{j_1}, \ldots, Y_{j_{m-1}}\}$ using (31).
8: Update $\widehat{A}_{ij_l} = \widehat{A}_{ij_l} + w_e$ for all $l = 1, \ldots, m-1$.
9: **end for**
10: **end for**
11: Let $\widehat{D} \in \mathbb{R}^{n \times n}$ be diagonal with $\widehat{D}_{ii} = \sum_{j=1}^{n} \widehat{A}_{ij}$, and $\widehat{L} = \widehat{D}^{-1}\widehat{A}$.
12: Compute k dominant left singular vectors of $\widehat{L}$, denoted by $\widehat{X} \in \mathbb{R}^{n \times k}$.
13: Normalise rows of $\widehat{X}$ to have unit norm.
14: Run k-means on the rows of the normalised matrix, and partition Y into k clusters.
15: From each obtained cluster, sample c/k subsets, each of size $(m-1)$.
16: Repeat from Step 3, and iterate until convergence.

Output: Clustering of Y into k disjoint clusters.

6. Experimental Validation

In this section, we present numerical illustrations related to uniform hypergraph partitioning. The numerical results are categorised into four parts. The first set of results is based on

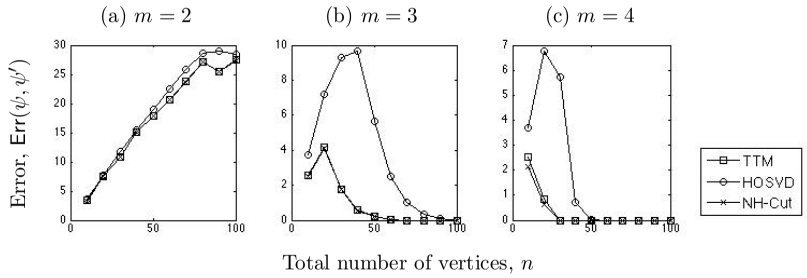


Figure 1: Number of vertices mis-clustered by TTM, HOSVD and NH-Cut as n increases. The figures from left to right correspond to cases with $m = 2$, 3 and 4, respectively.

the setting of Corollary 3, and validates our theoretical observations about TTM. We next compare the performance of TTM with several uniform hypergraph partitioning methods for some small scale problems. The reason for restricting our study to small problems is because, in such cases, the hypergraphs can be completely specified, and edge sampling can be avoided. The rest of the section focuses on practical versions of the subspace clustering problem, where we compare sampled variants of TTM with state of the art subspace clustering algorithms. Our experiments include both synthetic subspace clustering problems (Park et al., 2014) and benchmark motion segmentation problem (Tron and Vidal, 2007).

6.1 Comparison of Spectral Algorithms

We first compare the performance of TTM with the HOSVD based algorithm (Govindu, 2005; Ghoshdastidar and Dukkipati, 2014) and the NH-Cut algorithm (Zhou et al., 2007; Ghoshdastidar and Dukkipati, 2017). This study is based on the model related to Corollary 3, where a m-uniform hypergraph is generated on n vertices. We assume here that $\alpha_n = 1$, $k = 2$, and the true clusters are of equal size. The edges occur with following probabilities. If all vertices in an edge do not belong to the same cluster, then the edge probability is $q = 0.2$, else it is $(p+q)$ for some $p \in (0, 1-q)$ specified below.

In Figure 1, we show results for three examples, where p is fixed at $p = 0.1$, m is varied over $m = 2, 3, 4$, and the total number of vertices n grows from 10 to 100. For each case, 50 planted hypergraphs are generated, and subsequently partitioned by TTM, HOSVD and NH-Cut. The mean error, $\mathsf{Err}(\psi, \psi')$, is reported for each algorithm as a function of n. Figure 1 shows that the performance of TTM and NH-Cut are similar, and the errors incurred by these methods are significantly smaller than that of HOSVD. This observation validates Corollary 3 and Remark 5. It can also be seen empirically that all three methods have a sub-linear error rate for $m = 2$, *i.e.*, they are weakly consistent, whereas, $\mathsf{Err}(\psi, \psi') = o(1)$ for $m \geq 3$.

We consider another example on bi-partitioning 3-uniform hypergraphs, where we fix $q = 0.2$ but the gap p is decreased as 0.1, 0.05 and 0.025. Figure 2 shows the errors, averaged over 50 runs, incurred by the three methods as the hypergraph grows. Note that

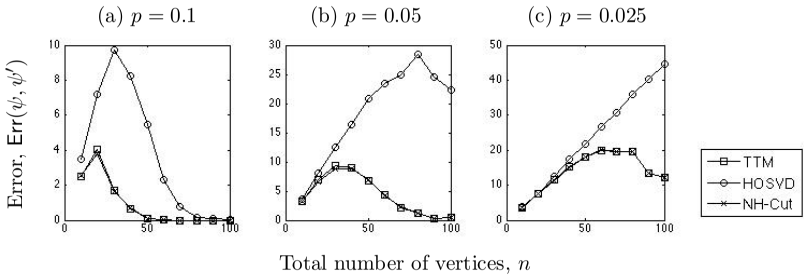


Figure 2: Number of vertices mis-clustered by TTM, HOSVD and NH-Cut as n increases. The figures from left to right correspond to cases with $p = 0.1$, 0.05 and 0.025, respectively.

the problem becomes harder as p reduces, and the performance of HOSVD is highly affected. But, the effect is much less in case of TTM and NH-Cut. This follows from Theorem 2, where one can observe that, in the present context $\mathsf{Err}(\psi, \psi')$ varies as $1/p^2$. Same holds for NH-Cut, but in the case of HOSVD, $\mathsf{Err}(\psi, \psi')$ varies as $1/p^4$ making the algorithm more sensitive to reduction in probability gap.

6.2 Comparison of Hypergraph Partitioning Methods

We consider similar studies with other hypergraph partitioning methods, such as methods based on symmetric non-negative tensor factorisation (SNTF) (Shashua et al., 2006), higher order game theoretic clustering (HGT) (Rota Bulo and Pelillo, 2013) and the hMETIS algorithm widely used in VLSI community (Karypis and Kumar, 2000). For the latter two methods, we have used implementations provided by the authors.

We first compare the different algorithms under a planted model for 3-uniform hypergraphs with $k = 3$ planted clusters of equal size. As before, we assume the hypergraph to be dense, $\alpha_n = 1$, and the inter-cluster edges occur with probability $q = 0.2$. We study the performance of the methods as the number of vertices n, and the probability gap p varies. The fractional clustering error, $\frac{1}{n}\mathsf{Err}(\psi, \psi')$, averaged over 50 runs, is reported in Figure 3.

The figure shows the previously observed trends for TTM, NH-Cut and HOSVD. In addition, it is observed that SNTF and hMETIS provide nearly similar, but marginally worse results than TTM. However, HGT uses a greedy strategy for extracting individual clusters, and hence, often identifies a majority of the vertices as outliers, thereby resulting in poor performance.

Since partitioning algorithms find use in a variety of applications, we compare the performance of the above algorithms for the subspace clustering problem. In particular, we consider the line clustering problem in an ambient space of dimension 3. We randomly generate three one-dimensional subspaces, and sampled n/k random points from each subspace. As mentioned in the previous section, the data points in subspace clustering problems are typically perturbed by noise. To simulate this behaviour, we add a zero mean Gaussian

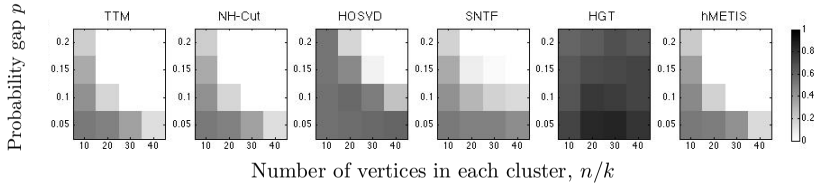


Figure 3: Fractional error incurred by hypergraph partitioning algorithms under a planted model. The cluster size, (n/k), and the probability gap p are varied. The colour bar indicates the shade corresponding to different levels of error, with darker shade representing larger error.

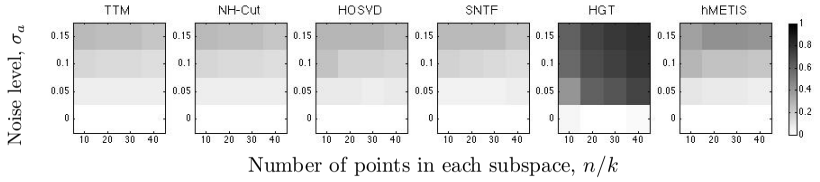


Figure 4: Fractional error incurred by hypergraph partitioning algorithms in clustering noisy points from three intersecting lines. The cluster size, (n/k), and the noise level σ_a are varied. The colour bar indicates the shade corresponding to different levels of error, with darker shade representing larger error.

noise vector to each point. The covariance of the noise vectors is given as $\sigma_a I$, where we vary σ_a to control the difficulty of the problem. We construct a weighted 3-uniform similarity hypergraph based on polar curvature of triplet of points, which is partitioned by the different methods. The fractional clustering errors are presented in Figure 4. As expected, all the methods can identify the exact subspace in the absence of noise, and the errors increase for larger σ_a. Apart from HGT, a good performance is observed from all the methods.

One can observe that the above comparisons were based on very small problems, where the hypergraph consists of at most 120 vertices. This restriction was imposed since specification of the entire affinity tensor is computationally infeasible for large hypergraphs. To demonstrate the performance of TTM in practical settings, we study its sampled variants in the subsequent sections.

6.3 Comparison of Subspace Clustering Algorithms: Synthetic Data

We now compare our method against the state of the art subspace clustering algorithms. We consider sampled variants of the TTM algorithm, *i.e.*, TTM with uniform sampling and TTM with iterative sampling (Tetris). We note that from practical consideration, we do

not consider aforementioned hypergraph partitioning methods that require computation of the entire tensor. The clustering algorithms under consideration include:

- k-means algorithm for clustering based on Euclidean distance,
- k-flats algorithm (Bradley and Mangasarian, 2000) which generalises k-means to subspace clustering,
- sparse subspace clustering (SSC) (Elhamifar and Vidal, 2013), which finds clusters by estimating the subspaces,
- subspace clustering using low-rank representation (LRR) (Liu et al., 2010),
- thresholding based subspace clustering (TSC) (Heckel and Bölcskei, 2013),
- faster variant of SSC using orthogonal matching pursuit (SSC-OMP) (Dyer et al., 2013),
- greedy subspace clustering using nearest subspace neighbour search and spectral clustering (NSN+Spectral) (Park et al., 2014),
- spectral curvature clustering (SCC) (Chen and Lerman, 2009), which is an iterative variant of HOSVD,
- sparse Grassmann clustering (SGC) (Jain and Govindu, 2013),[8] yet another variation of HOSVD where some information about the eigenvectors computed in previous iterations is retained,
- Algorithm Tetris, and
- Algorithm TTM with uniform sampling, which is derived by performing a single iteration of Steps 1-14 of Algorithm Tetris.[9]

We first focus on the problem of clustering randomly generated subspaces.[10] In an ambient space of dimension $r_a = 5$, we randomly generate $k = 5$ subspaces each of dimension $r = 3$. From each subspace, we randomly sample n/k points and perturb every point with a 5-dimensional Gaussian noise vector with mean zero and covariance $\sigma_a I$. In Figure 5, we report the fractional error, $\frac{1}{n}\mathsf{Err}(\psi, \psi')$, incurred by various subspace clustering algorithms when (n/k) and σ_a are varied. The results are averaged over 50 independent trials. We note that for existing methods, we fix the parameters as mentioned in Park et al. (2014). For Tetris and SGC, the parameters are set to the same values as SCC, where $c = 100k$ and σ as in (31) is determined by the algorithm. In case of uniformly sampled TTM, we fix σ to be same as the value determined by Tetris. To demonstrate that sampling more edges lead to error reduction, we consider uniform sampling for two values $c = 100k$ and $200k$.

Figure 5 shows that Tetris and SGC clearly outperform other methods over a wide range of settings. In particular, it can be seen that greedy methods like NSN is accurate

8. For this method, we have used our implementation.
9. Here, the edge sampling is not exactly uniform since we only select the c subsets of size $(m-1)$ uniformly.
10. The experimental setup has been adapted from Park et al. (2014), and the codes are available at: `http://sml.csa.iisc.ernet.in/SML/code/Feb16TensorTraceMax.zip`

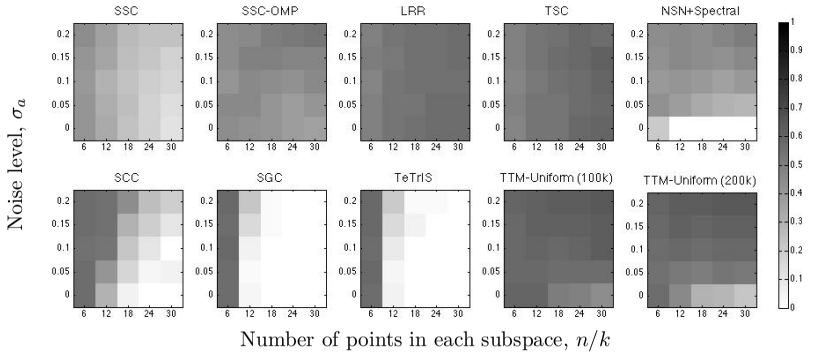


Figure 5: Fractional error incurred by subspace clustering algorithms for synthetic data. The number of points in each subspace, (n/k), and the variance of the noise vector σ_a is varied. The colour bar indicates the shades for different levels of error.

in the absence of noise, but a drastic increase in error occurs when the data is noisy. The effect of noise is much less in hypergraph based methods like SCC, SGC or Tetris. One can also observe that the hypergraph based methods do not work well when there are very few points in each cluster (for example, 6). This is expected since, by definition, these algorithms construct 5-uniform hypergraphs ($m = r + 2$) in this case, and hence, there are very few edges ($\binom{6}{5} = 6$) with large weight for each cluster. However, with increase in number of points, there is a rapid decay in the clustering error. This also shows the consistency of these methods empirically. To this end, it seems that NSN or SSC should be recommended for small scale problems (smaller n/k), whereas Tetris or SGC should be the algorithm of choice for larger n and possible presence of noise. Finally, we also observe that TTM with uniform sampling, even with twice the number of samples, performs quite poorly as compared to Tetris. However, with increase in the number of sampled edges, some extent of error reduction is observed.

6.4 Comparison of Subspace Clustering Algorithms: Motion Segmentation

The Hopkins 155 database (Tron and Vidal, 2007) contains a number of videos capturing motion of multiple objects or rigid bodies. In each video, few features are tracked along the frames, each giving rise to a motion trajectory that resides in a space of dimension twice the number of frames. One can show that under particular camera models, all trajectories corresponding to a particular rigid body motion span a subspace of dimension at most four (Tomasi and Kanade, 1992). Thus, the problem of segmenting different motions in a video can be posed as a subspace clustering problem.

The Hopkins database contains 120 sequences, each containing two motions, and 35 three motion sequences. We run above mentioned subspace clustering algorithms for purpose of

Algorithm	2 motion (120 sequences)			3 motion (35 sequences)		
	Mean (%)	Median (%)	Time (s)	Mean (%)	Median (%)	Time (s)
k-means	19.58	17.92	0.03	26.13	20.48	0.05
k-flats	13.19	10.01	0.38	15.45	14.88	0.76
SSC	1.53	0.00	0.80	4.40	0.56	1.51
LRR	2.13	0.00	0.94	4.03	1.43	1.29
SSC-OMP	16.93	13.28	0.72	27.61	23.79	1.23
TSC	18.44	16.92	0.19	28.58	29.67	0.51
NSN+Spec	3.62	0.00	0.08	8.28	2.76	0.17
SCC	2.53	0.03	0.45	6.40	1.46	0.76
SGC	3.50	0.41	0.54	9.08	5.05	0.89
Tetris	1.31	0.02	0.50	5.71	1.19	0.90

Table 1: Mean and median of clustering error and computational time for different subspace clustering algorithms on Hopkins 155 database.

motion segmentation. For existing approaches, the parameters specified in Park et al. (2014) have been used, and for Tetris and SGC, we use the parameters for SCC. TTM with uniform sampling is not considered due to its higher error rate. Table 1 reports the mean and median of the percentage errors incurred by different algorithms, where these statistics are computed over all 2-motion and 3-motion sequences. In order to remove the effect of randomisation due to sampling (for SCC, SGC, Tetris) or initialisation (for k-means, k-flats, NSN), we average the results over 20 independent trials. The average computational time (in seconds) of each algorithm for each video is also reported.[11]

Table 1 shows that Tetris performs quite well in comparison with state of the art subspace clustering algorithms. In particular, Tetris achieves least mean error for the two motion problem. The computational time for Tetris is also much smaller than other accurate methods like SSC and LRR. The mean error achieved by Tetris is also smaller than SCC in either cases. We note here that the best known results for Hopkins 155 database is achieved by the algorithm in Jung et al. (2014), which uses techniques based on epipolar geometry, and hence, it is not a subspace clustering algorithm. Smaller errors have also been reported in the literature when one construct larger tensors, $m = 8$ (Jain and Govindu, 2013), or uses manual tuning of hyper-parameters (Ghoshdastidar and Dukkipati, 2015a). However, in either cases, computational time increases considerably.

7. Conclusion

In this paper, we studied the problem of partitioning uniform hypergraphs that arises in several applications in computer vision and databases. We formalised the problem by defining a normalised associativity of a partition in a uniform hypergraph that extends a

11. We note that the reported time is based on the fact that we have used Matlab implementations of the algorithms, run on a Mac OS X operating system with 2.2 GHz Intel Core i7 processor and 16 GB memory.

similar notion used in graph literature. We showed that the task of finding a partition that maximises normalised associativity is equivalent to a tensor trace maximisation problem. We proposed a tensor spectral algorithm (TTM) to solve this problem, and following the lines of our previous works, we showed that the TTM algorithm is consistent under a planted partition model. To this end, we extended the existing model of Ghoshdastidar and Dukkipati (2014, 2017) by allowing both sparsity of edges as well as the possibility of weighted edges. Accounting for these factors makes the model more appropriate in the context of computer vision applications. We derived error bounds for the TTM approach under the planted partition model. Our bounds indicate that under mild assumptions on the sparsity of the hypergraph, TTM is weakly consistent, and the error bound for TTM is comparable that of NH-Cut, but better than the error rates for HOSVD. This fact is also validated numerically.

Weighted uniform hypergraphs have been particularly interesting in the computer vision since hypergraphs provide a natural way to represent multi-way similarities. Yet, it is computationally expensive to compute the entire affinity tensor of the hypergraph. As a consequence, several tensor sampling strategies have come into existence. We provide the first theoretical analysis of such sampling techniques in the context of uniform hypergraph partitioning. Our result suggests that consistency can be achieved even with very few sampled edges provided that one assigns a higher sampling probability for edges with larger weight. The derived sampling rate is much lower than that known in tensor literature (Bhojanapalli and Sanghavi, 2015; Jain and Oh, 2014), and our analysis also justifies the superior performance of popular sampling heuristics (Chen and Lerman, 2009; Duchenne et al., 2011). We finally proposed a iteratively sampled variant of TTM, and empirically demonstrated the potential of this method in subspace clustering and motion segmentation applications.

We conclude with the remark that this paper was motivated by practical aspects of hypergraph partitioning, and our aim was to present an approach that can be analysed theoretically, and at the same time, can compete with practical methods. While the paper manages to achieve this goal, there are several questions that still remain unanswered. We list some of the important open problems:

(i) What is the threshold for detecting planted partitions in hypergraphs?
This question was recently answered for a special model in the case of bi-partitioning (Florescu and Perkins, 2016), but more general cases have still not been explored. For instance, it would interesting to know any extension of the results in Abbe and Sandon (2016) for unweighted hypergraphs. However, this would still not provide any information in the weighted case. We noted that one of the reasons for our condition in (12) being worse by logarithmic factors was our analysis is for weighted hypergraphs. It is not known yet whether one can do better for weighted hypergraphs without making distributional assumptions on the edge weights.

(ii) What are the complicated examples? How can we deal with these cases?
Our error bound leads to a counter-intuitive conclusion that even after collapsing the tensor to a matrix, one can get better performance than direct tensor decomposition (HOSVD). The caveat here is that this result is stated for a special case, and still leaves the possibility that that there are models which do not satisfy the condition $\delta > 0$, and hence, TTM does

not work. In Ghoshdastidar and Dukkipati (2017), we constructed few examples, but it is not known whether HOSVD works in these cases. We also mention that the use of other tensor methods (such as power iterations) have not been analysed yet. While we showed that existing results cannot be applied due to the difference in assumptions, we do not claim that alternative tensor methods are inapplicable.

On a similar note, it is known in the case of graph partitioning that the eigen gap (quantified by δ in our results) is not the correct quantity that reflects the presence of a partition. For instance, there are planted graph models, which clearly reflect a separation of the classes, and yet δ is not positive. It would be interesting to see how one can deal with such examples in the case of hypergraphs. An alternative approach in such cases is to quantify the separation by the minimum difference (or distance) between the k rows of $\mathbf{B}$, which is a matrix for graphs. For uniform hypergraphs, a similar quantity would be the difference between the $(m-1)$-order slices of the tensor $\mathbf{B}$, but it is not clear yet how such a condition can be incorporated into the analysis of hypergraph partitioning.

(iii) What is the minimum sample size for achieving weak consistency?
Similar in spirit to the first problem, one can ask what is the minimum sampling rate required to achieve weak consistency irrespective of the partitioning algorithm. We have not made any attempt to answer this question, and surprisingly, the question is open even in graph partitioning. While column sampling techniques (Drineas et al., 2006) are often used in conjunction with spectral clustering, the error rate for this combination is not known.

(iv) How well does TTM extend for other higher order learning problems?
While TTM unifies a wide variety of higher order learning methods, the special cases studies in this paper were restricted to clustering problems. However, other problems like tensor matching can also be modelled as partitioning uniform hypergraph with two partitions of widely different sizes (Duchenne et al., 2011). In principle, this problem has a flavour quite similar to that of the well known planted clique problem (Alon et al., 1998). The state of the art algorithms in tensor matching rely on tensor power iterations, and in Ghoshdastidar and Dukkipati (2017), we showed that a naive use of NH-Cut does not fare well. We feel that in this particular setting, better theoretical guarantees can be achieved by power iteration based approaches.

Acknowledgments

We thank the reviewers for useful suggestions and key references. During this work, D. Ghoshdastidar was supported by the 2013 Google Ph.D. Fellowship in Statistical Learning Theory. This work is also partly supported by the Science and Engineering Research Board through the grant SERB:SB/S3/EECE/093/2014.

Appendix A. Proofs of Technical Lemmas and Corollaries

Here, we sequentially present the proofs of lemmas and corollaries stated in the paper. We begin with the calculations that lead to (8).

A.1 Proof of Equation (8)

We recall that the entries of $Y^{(l)}$ is simply $Y^{(l)}_{ij} = \frac{\mathbb{1}\{i \in \mathcal{V}_j\}}{(\mathrm{vol}(\mathcal{V}_j))^{\beta_l}}$. The claimed relation follows by computing diagonal entries of $\mathbf{A} \times_1 {Y^{(1)}}^T \times_2 {Y^{(2)}}^T \times_3 \ldots \times_m {Y^{(m)}}^T$, which are given by

$$\begin{aligned}\left(\mathbf{A} \times_1 {Y^{(1)}}^T \times_2 \ldots \times_m {Y^{(m)}}^T\right)_{j\ldots j} &= \sum_{i_1,\ldots,i_m=1}^{n} \mathbf{A}_{i_1\ldots i_m} Y^{(1)}_{i_1 j} \ldots Y^{(m)}_{i_m j} \\ &= m! \sum_{i_1<\ldots<i_m} \mathbf{A}_{i_1\ldots i_m} \frac{\mathbb{1}\{i_1,\ldots,i_m \in \mathcal{V}_j\}}{(\mathrm{vol}(\mathcal{V}_j))^{\beta_1+\ldots+\beta_m}} ,\end{aligned}$$

where we use the symmetry of the terms to group all $m!$ permutations of every distinct $i_1,\ldots,i_m$. Since $\sum_l \beta_l = 1$, the denominator is simply $\mathrm{vol}(\mathcal{V}_j)$, whereas the indicator in the numerator counts only the edges $e \subset \mathcal{V}_j$ and $\mathbf{A}_{i_1\ldots i_m}$ provides the weights. So the above quantity is simply $m!\frac{\mathrm{assoc}(\mathcal{V}_j)}{\mathrm{vol}(\mathcal{V}_j)}$, and summing over all diagonal entries results in the normalised associativity, scaled by $m!$.

A.2 Proof of Corollary 3

We begin by computing $\mathcal{A}$ as defined in (10)

$$\mathcal{A}_{ij} = \begin{cases} (m-2)!\alpha_n \left(p\binom{\frac{n}{k}-2}{m-2} + q\binom{n-2}{m-2}\right) & \text{if } i \neq j, \psi_i = \psi_j \\ (m-2)!\alpha_n q\binom{n-2}{m-2} & \text{if } i \neq j, \psi_i \neq \psi_j \\ 0 & \text{if } i = j. \end{cases}$$

From the definition of G, $\mathcal{D}_{\min}$ and δ, one can compute that

$$\mathcal{D}_{\min} = (m-1)!\alpha_n \left(p\binom{\frac{n}{k}-1}{m-1} + q\binom{n-1}{m-1}\right) , \tag{32}$$

and

$$\delta = \lambda_k(G)\frac{n}{k\mathcal{D}_{\min}} = \frac{(m-2)!\alpha_n pn}{k\mathcal{D}_{\min}}\binom{\frac{n}{k}-2}{m-2} \tag{33}$$

We need to validate that the conditions in (12), or equivalently (16), are satisfied. Given $\alpha_n = 1$, one can see that $\mathcal{D}_{\min} = \Theta(n^{m-1})$ easily satisfies first condition of (16) for large n.[12] Also

$$\delta^2\mathcal{D}_{\min} = \Theta\left(\left(\frac{n}{k}\right)^{2m-2}\frac{1}{\mathcal{D}_{\min}}\right) = \Theta\left(\frac{n^{m-1}}{k^{2m-2}}\right) = \Omega\left(n^{(m-1)/2}(\ln n)^{2m-2}\right) ,$$

taking into account that $k = O\left(\frac{n^{1/4}}{\ln n}\right)$. Thus, the second condition in (16) also holds for large n and for all $m \geq 2$. Subsequently, one can applying the bound in (13) to claim that $\mathsf{Err}(\psi,\psi') = O\left(\frac{n^{(3-m)/2}}{(\ln n)^{2m-3}}\right)$ with probability $(1-o(1))$.

12. The notation $f_n = \Theta(n)$ denotes that there exists constants c, C such that $cn \leq f_n \leq Cn$ for all large n.

A.3 Proof of Corollary 4

For $k = O(\ln n)$, one can verify that (12) holds if $\mathcal{D}_{\min} = \Omega\left(\frac{(\ln n)^3}{\delta^2}\right)$. From (32) and (33), we have $\mathcal{D}_{\min} = \Theta(\alpha_n n^{m-1})$ and $\delta^2 \mathcal{D}_{\min} = \Omega\left(\alpha_n n^{m-1} (\ln n)^{2-2m}\right)$. Hence, choosing $\alpha_n \geq \frac{C(\ln n)^{2m+1}}{n^{m-1}}$ for sufficiently large C ensures that (16) is satisfied. Subsequently with probability $\left(1 - O\left(n^{-2} + (\ln n)^{-1/4}\right)\right) = (1 - o(1))$, we obtain an error bound

$$\mathsf{Err}(\psi, \psi') = O\left(\frac{n \ln n}{\delta^2 \mathcal{D}_{\min}}\right) = O\left(\frac{n \ln n}{\alpha_n n^{m-1} (\ln n)^{2-2m}}\right) = O\left(\frac{n}{(\ln n)^2}\right) = o(n) \ ,$$

which completes the proof.

A.4 Proof of Lemma 1

The proof is along the lines of the proof of Lemma 4.5 in Ghoshdastidar and Dukkipati (2017). We still include a sketch of the proof since the quantities involved are different from the terms dealt in the mentioned paper. From the discussions following (10), one can see that the matrix $\mathcal{A}$ may be expressed as

$$\mathcal{A} = Z G Z^T - J \ ,$$

where $J \in \mathbb{R}^{n \times n}$ is diagonal with $J_{ii} = G_{\psi_i \psi_i}$. Following the arguments of Ghoshdastidar and Dukkipati (2017), one can show that there is a matrix $\mathcal{G} \in \mathbb{R}^{k \times k}$ with eigen decomposition $\mathcal{G} = U \Lambda_1 U^T$ such that $\mathcal{L}\mathcal{X} = \mathcal{X}\Lambda_1$, where $\mathcal{X} = Z(Z^T Z)^{-1/2} U$, and it satisfies $\mathcal{X}^T \mathcal{X} = I$. Thus, the columns of $\mathcal{X}$ are orthonormal eigenvectors of $\mathcal{L}$ corresponding to the eigenvalues in Λ_1. It is also known that the other $(n-k)$ orthonormal eigenvectors correspond to eigenvalues from the set $\left\{-\frac{J_{ii}}{\mathcal{D}_{ii}} : 1 \leq i \leq n\right\}$.

Thus, to prove the claim we need to ensure that $\mathcal{X}$ corresponds to the dominant eigenvectors of $\mathcal{L}$, or in other words,

$$\lambda_k(\mathcal{G}) > \max_{1 \leq i \leq n} \left(-\frac{J_{ii}}{\mathcal{D}_{ii}}\right) = - \min_{1 \leq i \leq n} \frac{G_{\psi_i \psi_i}}{\mathcal{D}_{ii}} \ . \tag{34}$$

A lower bound on $\lambda_k(\mathcal{G})$ can be derived from Rayleigh's principle and Weyl's inequality as

$$\lambda_k(\mathcal{G}) \geq \lambda_k(G) \min_{1 \leq i \leq n} \frac{(Z^T Z)_{\psi_i \psi_i}}{\mathcal{D}_{ii}} - \max_{1 \leq i \leq n} \frac{G_{\psi_i \psi_i}}{\mathcal{D}_{ii}} \ .$$

Noting that $(Z^T Z)_{jj} = n_j$, size of cluster-j, one can readily see from above that if $\delta > 0$, then (34) is satisfied and hence, $\mathcal{X}$ contains the dominant eigenvectors. It is also useful to note that $\delta \leq \lambda_k(\mathcal{L}) - \lambda_{k+1}(\mathcal{L})$, *i.e.*, δ is a lower bound on the eigen gap between the k^{th} and $(k+1)^{th}$ largest eigenvalues of $\mathcal{L}$. This fact is used later in the proof of Lemma 7.

A.5 Proof of Lemma 6

We begin the proof with the claims that if $\mathcal{D}_{\min} > 9(m-1)! \ln n$, then

$$\mathsf{P}_H \left(\max_{1 \leq i \leq n} \left| \frac{D_{ii}}{\mathcal{D}_{ii}} - 1 \right| > 3 \sqrt{\frac{(m-1)! \ln n}{\mathcal{D}_{\min}}} \right) \leq \frac{2}{n^2} \ , \tag{35}$$

and

$$\mathsf{P}_H \left(\|\mathcal{D}^{-1/2}(A - \mathcal{A})\mathcal{D}^{-1/2}\|_2 > 3 \sqrt{\frac{(m-1)! \ln n}{\mathcal{D}_{\min}}} \right) \leq \frac{2}{n^2} \ . \tag{36}$$

We now bound $\|L - \mathcal{L}\|_2$ as

$$\|L - \mathcal{L}\|_2 \leq \|\mathcal{D}^{-1/2} A \mathcal{D}^{-1/2} - D^{-1/2} A D^{-1/2}\|_2 + \|\mathcal{D}^{-1/2} A \mathcal{D}^{-1/2} - \mathcal{D}^{-1/2} \mathcal{A} \mathcal{D}^{-1/2}\|_2 \ .$$

We expand the first term as

$$\begin{aligned} &\|\mathcal{D}^{-1/2} A \mathcal{D}^{-1/2} - D^{-1/2} A D^{-1/2}\|_2 \\ &\qquad \leq \|(\mathcal{D}^{-1/2} - D^{-1/2}) A \mathcal{D}^{-1/2} + D^{-1/2} A (\mathcal{D}^{-1/2} - D^{-1/2})\|_2 \\ &\qquad \leq \|(\mathcal{D}^{-1} D)^{1/2} - I\|_2 \|(D\mathcal{D}^{-1})^{1/2}\|_2 + \|(\mathcal{D}^{-1} D)^{1/2} - I\|_2, \end{aligned}$$

where the last inequality follows since $\|D^{-1/2} A D^{-1/2}\|_2 = 1$. Also, note that

$$\|(\mathcal{D}^{-1} D)^{1/2} - I\|_2 = \max_{1 \leq i \leq n} \left| \sqrt{\frac{D_{ii}}{\mathcal{D}_{ii}}} - 1 \right| \leq \max_{1 \leq i \leq n} \left| \frac{D_{ii}}{\mathcal{D}_{ii}} - 1 \right| .$$

Combining above arguments, we can write

$$\|L - \mathcal{L}\|_2 \leq \max_{1 \leq i \leq n} \left| \frac{D_{ii}}{\mathcal{D}_{ii}} - 1 \right| \left(2 + \max_{1 \leq i \leq n} \left| \frac{D_{ii}}{\mathcal{D}_{ii}} - 1 \right| \right) + \|\mathcal{D}^{-1/2} A \mathcal{D}^{-1/2} - \mathcal{D}^{-1/2} \mathcal{A} \mathcal{D}^{-1/2}\|_2. \tag{37}$$

Using the bounds in (35) and (36) along with the fact that $3\sqrt{\frac{(m-1)! \ln n}{\mathcal{D}_{\min}}} < 1$, one arrives at the claim.

We now prove the concentration bound in (35). Observe that

$$D_{ii} = \sum_{j=1}^{n} A_{ij} = \sum_{i_2, \ldots, i_m = 1}^{n} \mathbf{A}_{i i_2 \ldots i_m} = (m-1)! \sum_{e \in \mathcal{E} : e \ni i} w_e \ ,$$

where the last equality holds since the summation over all $i_2, \ldots, i_m$ counts each edge containing node-i $(m-1)!$ times. Since, D_{ii} is a sum of independent random variables, we can use Bernstein inequality to obtain for any $t > 0$,

$$\begin{aligned} \mathsf{P}_H \left(|D_{ii} - \mathcal{D}_{ii}| > t \mathcal{D}_{ii} \right) &= \mathsf{P}_H \left(\left| \sum_{e \in \mathcal{E} : e \ni i} w_e - \mathsf{E}_H[w_e] \right| > \frac{t \mathcal{D}_{ii}}{(m-1)!} \right) \\ &\leq 2 \exp \left(\frac{-\left(\frac{t \mathcal{D}_{ii}}{(m-1)!}\right)^2}{2 \sum\limits_{e \in \mathcal{E} : e \ni i} \mathsf{Var}_H(w_e) + \frac{2}{3} \frac{t \mathcal{D}_{ii}}{(m-1)!}} \right) \ . \end{aligned} \tag{38}$$

Since $w_e \in [0, 1]$, we have

$$\sum_{e \in \mathcal{E} : e \ni i} \mathsf{Var}_H(w_e) \leq \sum_{e \in \mathcal{E} : e \ni i} \mathsf{E}_H[w_e] = \frac{\mathcal{D}_{ii}}{(m-1)!} \ .$$

Substituting this in (38), we have

$$\mathsf{P}_H\left(|D_{ii}-\mathcal{D}_{ii}|>t\mathcal{D}_{ii}\right)\leq 2\exp\left(-\frac{t^2\mathcal{D}_{ii}}{3(m-1)!}\right)\leq 2\exp\left(-\frac{t^2\mathcal{D}_{\min}}{3(m-1)!}\right).$$

The bound in (35) follows from above by setting $t=3\sqrt{\frac{(m-1)!\ln n}{\mathcal{D}_{\min}}}$, and using a union bound over all $i=1,\ldots,n$.

Finally, we derive (36) using matrix Bernstein inequality (Tropp, 2012, Theorem 1.4), which states that for independent, symmetric, random matrices $Y_1,\ldots,Y_M\in\mathbb{R}^{n\times n}$ with $\mathsf{E}_H[Y_i]=0$ and $\|Y_i\|_2\leq R$ almost surely, one has

$$\mathsf{P}\left(\left\|\sum_{i=1}^{M}Y_i\right\|_2>t\right)\leq n\exp\left(\frac{-t^2}{2\|\sum_i\mathsf{E}[Y_i^2]\|_2+\frac{2Rt}{3}}\right)\tag{39}$$

for any $t>0$. Owing to the representation in (17), one can write

$$\mathcal{D}^{-1/2}(A-\mathcal{A})\mathcal{D}^{-1/2}=\sum_{e\in\mathcal{E}}(m-2)!\,(w_e-\mathsf{E}_H[w_e])\,\mathcal{D}^{-1/2}R_e\mathcal{D}^{-1/2}$$

as a sum of independent, zero mean random matrices. One can verify that $\|R_e\|_2\leq(m-1)$, and hence,

$$\left\|(m-2)!\,(w_e-\mathsf{E}_H[w_e])\,\mathcal{D}^{-1/2}R_e\mathcal{D}^{-1/2}\right\|_2\leq\frac{(m-1)!}{\mathcal{D}_{\min}}\,.$$

In addition, one can bound

$$\begin{aligned}
&\left\|\sum_{e\in\mathcal{E}}\mathsf{E}_H\left[\left((m-2)!\,(w_e-\mathsf{E}_H[w_e])\,\mathcal{D}^{-1/2}R_e\mathcal{D}^{-1/2}\right)^2\right]\right\|_2\\
&=((m-2)!)^2\left\|\sum_{e\in\mathcal{E}}\mathsf{Var}_H(w_e)\mathcal{D}^{-1/2}R_e\mathcal{D}^{-1}R_e\mathcal{D}^{-1/2}\right\|_2\\
&=((m-2)!)^2\left\|\sum_{e\in\mathcal{E}}\mathsf{Var}_H(w_e)\left(\mathcal{D}^{-1}R_e\right)^2\right\|_2\\
&\leq((m-2)!)^2\max_{1\leq i\leq n}\sum_{j=1}^{n}\sum_{e\in\mathcal{E}}\mathsf{Var}_H(w_e)\left(\left(\mathcal{D}^{-1}R_e\right)^2\right)_{ij}\\
&\leq\frac{((m-2)!)^2}{\mathcal{D}_{\min}}\max_{1\leq i\leq n}\frac{1}{\mathcal{D}_{ii}}\sum_{e\in\mathcal{E}}\mathsf{E}_H(w_e)\sum_{j=1}^{n}\left(R_e^2\right)_{ij}\,.
\end{aligned}$$

Here, the first inequality holds due to Gerschgorin's theorem. Observing that the row sum of R_e^2 is at most $(m-1)^2$, the expression can be simplified to show that the quantity is bounded from above by $\frac{(m-1)!}{\mathcal{D}_{\min}}$. Setting $t=3\sqrt{\frac{(m-1)!\ln n}{\mathcal{D}_{\min}}}$ in (39), and combining above arguments, one arrives at (36). This completes the proof.

A.6 Proof of Corollary 10

We begin by characterising β for either sampling methods. Note that $\beta\geq\max_e\frac{w_e}{p_e}$. Since, $|\mathcal{E}|=\binom{n}{m}$, it follows that for uniform sampling $p_e=\binom{n}{m}^{-1}$ for all e, and hence, an appropriate choice of $\beta=\binom{n}{m}$. On the other hand, for the sampling in (25), $\max_e\frac{w_e}{p_e}=\sum_e w_e$. Using Bernstein inequality, one may easily bound this term from above by $2\sum_e\mathsf{E}_H[w_e]\leq 2\alpha_n\binom{n}{m}$, where the bound holds with probability $(1-n^{-2})$.

Thus, ignoring constants factors, one may set $\beta=\xi n^m$, where $\xi=1$ for uniform sampling and α_n for weighted sampling. The conditions in (26) follow directly from (23) and d,δ computed in the proof of Corollary 3.

A.7 Proof of Lemma 11

Let β be defined as in Theorem 9 and $D_{\min}=\min_{1\leq i\leq n}D_{ii}$. Assume that

$$\mathcal{D}_{\min}>36(m-1)!\ln n\quad\text{and}\quad N>9\left(1+\frac{2\beta(m-1)!}{\mathcal{D}_{\min}}\right)\ln n\,.\tag{40}$$

Also let Γ denote the event

$$\Gamma=\left\{D_{\min}>\frac{\mathcal{D}_{\min}}{2}\right\}\bigcap\left\{\max_{e\in\mathcal{E}}\frac{w_e}{p_e}\leq\beta\right\}.$$

Then, conditioned on a given random hypergraph and the event Γ, we claim that the following bounds hold with probability $(1-\frac{2}{n^2})$,

$$\max_{1\leq i\leq n}\left|\frac{\widehat{D}_{ii}}{D_{ii}}-1\right|\leq 3\sqrt{\frac{\ln n}{N}\left(1+\frac{2\beta(m-1)!}{\mathcal{D}_{\min}}\right)}\,.\tag{41}$$

and

$$\|D^{-1/2}(\widehat{A}-A)D^{-1/2}\|_2\leq 3\sqrt{\frac{\ln n}{N}\left(1+\frac{2\beta(m-1)!}{\mathcal{D}_{\min}}\right)}\,.\tag{42}$$

Assuming that the above hold, we now derive a bound on $\|\widehat{L}-L\|_2$ in the following way. First, note that the bounds in (41) and (42) are with respect to a conditional probability measure, and need to be converted into a bound with respect to the joint probability measure $\mathsf{P}_{S,H}$. This is not hard to derive as one can see in the case of (42), where one can write

$$\begin{aligned}
&\mathsf{P}_{S,H}\left(\|D^{-1/2}(\widehat{A}-A)D^{-1/2}\|_2>3\sqrt{\frac{\ln n}{N}\left(1+\frac{2\beta(m-1)!}{\mathcal{D}_{\min}}\right)}\right)\\
&=\mathsf{E}_H\left[\mathsf{P}_{S|H}\left(\|D^{-1/2}(\widehat{A}-A)D^{-1/2}\|_2>3\sqrt{\frac{\ln n}{N}\left(1+\frac{2\beta(m-1)!}{\mathcal{D}_{\min}}\right)}\right)\right]\\
&\leq\mathsf{P}_H(\Gamma)\,\mathsf{E}_{H|\Gamma}\left[\mathsf{P}_{S|H,\Gamma}\left(\|D^{-1/2}(\widehat{A}-A)D^{-1/2}\|_2>3\sqrt{\frac{\ln n}{N}\left(1+\frac{2\beta(m-1)!}{\mathcal{D}_{\min}}\right)}\right)\right]+\mathsf{P}_H(\Gamma^c)\\
&=O\left(\frac{1}{n^2}\right)+\mathsf{P}_H(\Gamma^c)\,,
\end{aligned}\tag{43}$$

where the inequalities follow by observing that all the quantities are smaller than one, and the first term is bounded due to (42). For bounding $\mathsf{P}_H(\Gamma^c)$, note that from (35), it follows that with probability $(1 - O(n^{-2}))$, for all $i = 1, \ldots, n$,

$$D_{ii} > \mathcal{D}_{ii}\left(1 - 3\sqrt{\frac{(m-1)!\ln n}{\mathcal{D}_{\min}}}\right).$$

Hence, if $\mathcal{D}_{\min} > 36(m-1)!\ln n$, then

$$D_{\min} > \mathcal{D}_{\min}\left(1 - 3\sqrt{\frac{(m-1)!\ln n}{\mathcal{D}_{\min}}}\right) > \frac{\mathcal{D}_{\min}}{2}.$$

This fact, along with the assumption on β, shows that $\mathsf{P}_H(\Gamma^c) = o(1)$, and so, the upper bound on $\|D^{-1/2}(\widehat{A} - A)D^{-1/2}\|_2$ holds with probability $(1 - o(1))$ even with respect to joint probability measure. Similar result also holds for (41). Subsequently, we follow the arguments leading to (37) to conclude that

$$\begin{aligned}\|\widehat{L} - L\|_2 &\le \max_{1\le i\le n}\left|\frac{\widehat{D}_{ii}}{D_{ii}} - 1\right|\left(2 + \max_{1\le i\le n}\left|\frac{\widehat{D}_{ii}}{D_{ii}} - 1\right|\right) + \|D^{-1/2}(\widehat{A} - A)D^{-1/2}\|_2 \\ &\le 12\sqrt{\frac{\ln n}{N}\left(1 + \frac{2\beta(m-1)!}{\mathcal{D}_{\min}}\right)},\end{aligned}$$

where the last inequality holds with probability $(1 - O(n^{-2}))$ under the conditions stated in (40). To complete the proof, we derive the bounds (41) and (42), which again rely on the use of Bernstein inequality. For this, observe that

$$\widehat{D}_{ii} = \frac{(m-2)!}{N}\sum_{j=1}^{n}\sum_{e\in\mathcal{I}}\frac{w_e}{p_e}(R_e)_{ij} = \frac{(m-1)!}{N}\sum_{e\in\mathcal{I}}\frac{w_e}{p_e}\mathbb{1}\{i\in e\},$$

where for each $e \in \mathcal{I}$,

$$\begin{aligned}\mathsf{E}_{S|H,\Gamma}\left[\frac{w_e}{p_e}\mathbb{1}\{i\in e\}\right] &= \sum_{e'\in\mathcal{E}:e'\ni i} p_{e'}\frac{w_{e'}}{p_{e'}} = \frac{D_{ii}}{(m-1)!}, \\ \mathsf{Var}_{S|H,\Gamma}\left[\frac{w_e}{p_e}\mathbb{1}\{i\in e\}\right] &= \sum_{e'\in\mathcal{E}:e'\ni i}\frac{w_{e'}^2}{p_{e'}} - \left(\frac{D_{ii}}{(m-1)!}\right)^2 \le \left(\beta - \frac{D_{ii}}{(m-1)!}\right)\frac{D_{ii}}{(m-1)!},\end{aligned}$$

and almost surely with respect to $\mathsf{P}_{S|H,\Gamma}$,

$$\left|\frac{w_e}{p_e}\mathbb{1}\{i\in e\} - \frac{D_{ii}}{(m-1)!}\right| \le \left(\beta + \frac{D_{ii}}{(m-1)!}\right).$$

Define $t = 3\sqrt{\frac{\ln n}{N}\left(1 + \frac{2\beta(m-1)!}{\mathcal{D}_{\min}}\right)}$. Since the samples $e \in \mathcal{I}$ are independent and identically distributed, we can use Bernstein inequality to write

$$\begin{aligned}&\mathsf{P}_{S|H,\Gamma}\left(|\widehat{D}_{ii} - D_{ii}| > tD_{ii}\right) \\ &\quad= \mathsf{P}_{S|H,\Gamma}\left(\left|\sum_{e\in\mathcal{I}}\frac{w_e}{p_e}\mathbb{1}\{i\in e\} - \frac{D_{ii}}{(m-1)!}\right| > \frac{NtD_{ii}}{(m-1)!}\right) \\ &\quad\le 2\exp\left(\frac{-\frac{N^2t^2D_{ii}^2}{(m-1)!}}{2N\left(\beta - \frac{D_{ii}}{(m-1)!}\right)\frac{D_{ii}}{(m-1)!} + \frac{2}{3}\frac{NtD_{ii}}{(m-1)!}\left(\beta + \frac{D_{ii}}{(m-1)!}\right)}\right) \\ &\quad\le 2\exp\left(\frac{-\frac{ND_{ii}t^2}{(m-1)!}}{\frac{2}{3}\left(4\beta - 2\frac{D_{ii}}{(m-1)!}\right)}\right) \\ &\quad\le 2\exp\left(\frac{-\frac{N\mathcal{D}_{\min}t^2}{2(m-1)!}}{\frac{2}{3}\left(4\beta + 2\frac{\mathcal{D}_{\min}}{(m-1)!}\right)}\right) \le \frac{2}{n^3}.\end{aligned}$$

The inequalities are derived using above relations, and the definition of Γ. From above, (41) follows from union bound.

To prove (42), observe from (22) that

$$D^{-1/2}\widehat{A}D^{-1/2} = \frac{1}{N}\sum_{e\in\mathcal{I}}(m-2)!\frac{w_e}{p_e}D^{-1/2}R_eD^{-1/2}$$

is a sum of independent random matrices with

$$\mathsf{E}_{S|H,\Gamma}\left[(m-2)!\frac{w_e}{p_e}D^{-1/2}R_eD^{-1/2}\right] = D^{-1/2}AD^{-1/2}$$

and

$$\begin{aligned}\left\|(m-2)!\frac{w_e}{p_e}D^{-1/2}R_eD^{-1/2} - D^{-1/2}AD^{-1/2}\right\|_2 &\le (m-2)!\beta\|D^{-1/2}R_eD^{-1/2}\|_2 + 1 \\ &\le \left(\frac{2\beta(m-1)!}{\mathcal{D}_{\min}} + 1\right).\end{aligned}$$

The first bound uses the fact $\|D^{-1/2}AD^{-1/2}\|_2 = 1$ and the second follows since $D_{\min} > \frac{1}{2}\mathcal{D}_{\min}$ and $\|R_e\|_2 \le (m-1)$. We can also bound the norm of the variance term as

$$\begin{aligned}&\left\|\mathsf{E}_{S|H,\Gamma}\left[\left((m-2)!\frac{w_e}{p_e}D^{-1/2}R_eD^{-1/2} - D^{-1/2}AD^{-1/2}\right)^2\right]\right\|_2 \\ &= \left\|-\left(D^{-1/2}AD^{-1/2}\right)^2 + ((m-2)!)^2\sum_{e\in\mathcal{E}}\frac{w_e^2}{p_e}D^{-1/2}R_eD^{-1}R_eD^{-1/2}\right\|_2 \\ &\le 1 + \frac{((m-2)!)^2\beta}{D_{\min}}\left\|\sum_{e\in\mathcal{E}}w_eD^{-1}(R_e)^2\right\|_2 \le \left(1 + \frac{2\beta(m-1)!}{\mathcal{D}_{\min}}\right).\end{aligned}$$

Using these relations and the matrix Bernstein inequality, the bound in (42) can be derived quite similar to the derivation of (36).

References

E. Abbe and C. Sandon. Achieving the KS threshold in the general stochastic block model with linearized acyclic belief propagation. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

S. Agarwal, J. Lim, L. Zelnik-Manor, P. Perona, D. Kriegman, and S. Belongie. Beyond pairwise clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 838–845, 2005.

S. Agarwal, K. Branson, and S. Belongie. Higher order learning with graphs. In *International Conference on Machine Learning (ICML)*, pages 17–24, 2006.

N. Alon, M. Krivelvich, and B. Sudakov. Finding a large hidden clique in a random graph. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 594–598, 1998.

A. Anandkumar, R. Ge, D. Hsu, and S. M. Kakade. A tensor approach to learning mixed membership community models. *Journal of Machine Learning Research*, 15:2239–2312, 2014.

A. Anandkumar, R. Ge, and M. Janzamin. Learning overcomplete latent variable models through tensor methods. In *Conference on Learning Theory (COLT)*, 2015.

E. Arias-Castro, G. Chen, and G. Lerman. Spectral clustering based on local linear approximations. *Electronic Journal of Statistics*, 5:1537–1587, 2011.

V. Bapst and A. Coja-Oghlan. The condensation phase transition in the regular k-SAT model. *arXiv preprint*, arXiv:1507.03512, 2015.

M. Belkin, I. Matveeva, and P. Niyogi. Regularization and semi-supervised learning on large graphs. In *Conference on Learning Theory (COLT)*, 2004.

C. Berge. *Hypergraphs: Combinatorics of Finite Sets*, volume 45. Elsevier, 1984.

F. Bernstein. Zur theorie der trigonometrischen Reihen. *Leipz. Bet.*, 60:325–338, 1908.

A. Bhaskara, M. Charikar, A. Moitra, and A. Vijayaraghavan. Smoothed analysis of tensor decompositions. In *ACM Symposium on Theory of Computing (STOC)*, pages 594–603, 2014.

S. Bhojanapalli and S. Sanghavi. A new sampling technique for tensors. *arXiv preprint*, arXiv:1502.05023, 2015.

M. Bolla. Spectra, Euclidean representations and clusterings of hypergraphs. *Discrete Mathematics*, 117(1):19–39, 1993.

C. Boutsidis, A. Gittens, and P. Kambadur. Spectral clustering via the power method - provably. In *International Conference on Machine Learning (ICML)*, 2015.

P. S. Bradley and O. L. Mangasarian. k-plane clustering. *Journal of Global Optimization*, 16:23–32, 2000.

G. Chen and G. Lerman. Spectral curvature clustering. *International Journal of Computer Vision*, 81(3):317–330, 2009.

P. Comon. From source separation to blind equalization: Contrast based approaches. In *International Conference on Image and Signal Processing*, 2001.

P. Comon, G. Golub, L. H. Lim, and B. Mourrain. Symmetric tensors and symmetric tensor rank. *SIAM Journal of Matrix Analysis and Applications*, 30(3):1254–1279, 2008.

T. Cour, P. Srinivasan, and J. Shi. Balanced graph matching. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

P. Drineas, R. Kannan, and M. W. Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1):132–157, 2006.

O. Duchenne, F. Bach, I.-S. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33 (12):2383–2395, 2011.

E. L. Dyer, A. C. Sankaranarayanan, and R. G. Baraniuk. Greedy feature selection for subspace clustering. *Journal of Machine Learning Research*, 14(1):2487–2517, 2013.

E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2765–2781, 2013.

L. Florescu and W. Perkins. Spectral thresholds in the bipartite stochastic block model. In *Conference on Learning Theory (COLT)*, pages 943–959, 2016.

C. Gao, Z. Ma, A. Y. Zhang, and H. H. Zhou. Achieving optimal misclassification proportion in stochastic block model. *arXiv preprint*, arXiv:1505.03772, 2015.

G. Ghoshal, V. Zlatic, G. Caldarelli, and M. E. J. Newman. Random hypergraphs and their applications. *Physical Review E*, 79(066118), 2009.

D. Ghoshdastidar and A. Dukkipati. Consistency of spectral partitioning of uniform hypergraphs under planted partition model. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

D. Ghoshdastidar and A. Dukkipati. Spectral clustering using multilinear SVD: Analysis, approximations and applications. In *AAAI Conference on Artificial Intelligence*, 2015a.

D. Ghoshdastidar and A. Dukkipati. A provable generalized tensor spectral method for uniform hypergraph partitioning. In *International Conference on Machine Learning (ICML)*, 2015b.

D. Ghoshdastidar and A. Dukkipati. Consistency of spectral hypergraph partitioning under planted partition model. *The Annals of Statistics*, 45(1):289–315, 2017.

D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamical systems. *VLDB Journal*, 8(3-4):222–236, 2000.

V. M. Govindu. A tensor decomposition for geometric grouping and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1150–1157, 2005.

S. Guha, R. Rastogi, and K. Shim. Rock: A robust clustering algorithm for categorical attributes. In *International Conference on Data Engineering*, pages 512–521, 1999.

R. Heckel and H. Bölcskei. Subspace clustering via thresholding and spectral clustering. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2013.

P. Jain and S. Oh. Provable tensor factorization with missing data. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1431–1439, 2014.

S. Jain and V. M. Govindu. Efficient higher-order clustering on the Grassmann manifold. In *IEEE International Conference on Computer Vision (ICCV)*, 2013.

H. Jung, J. Ju, and J. Kim. Rigid motion segmentation using randomized voting. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1210–1217, 2014.

G. Karypis and V. Kumar. Multilevel k-way hypergraph partitioning. *VLSI Design*, 11(3): 285–300, 2000.

J. Lei and A. Rinaldo. Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43, 2015.

M. Leordeanu and C. Sminchisescu. Efficient hypergraph clustering. In *International Conference on Artificial Intelligence and Statistics*, 2012.

L.-H. Lim. Singular values and eigenvalues of tensors: A variational approach. In *IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, pages 129–132, 2005.

G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *International Conference on Machine Learning (ICML)*, 2010.

T. Ma, J. Shi, and D. Steurer. Polynomial-time tensor decompositions with sum-of-squares. In *Annual Symposium on Foundations of Computer Science (FOCS)*, 2016.

F. McSherry. Spectral partitioning of random graphs. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 529–537, 2001.

T. Michoel and B. Nachtergaele. Alignment and integration of complex networks by hypergraph-based spectral clustering. *Physical Review E*, 86(056111), 2012.

E. Mossel, J. Neeman, and A. Sly. A proof of the block model threshold conjecture. *arXiv preprint*, arXiv:1311.4115, 2013.

A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems (NIPS)*, pages 849–856, 2002.

Q. Nguyen, A. Gautier, and M. Hein. A flexible tensor block coordinate ascent scheme for hypergraph matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5270–5278, 2015.

P. Ochs and T. Brox. Higher order motion models and spectral clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

R. Ostrovsky, Y. Rabani, L. J. Schulman, and C. Swamy. The effectiveness of Lloyd-type methods for the k-means problem. *Journal of the ACM*, 59(6):28:1–28, 2012.

D. Park, C. Caramanis, and S. Sanghavi. Greedy subspace clustering. In *Advances in Neural Information Processing Systems (NIPS)*, volume 27, pages 2753–2761, 2014.

E. Richard and A. Montanari. A statistical model for tensor PCA. In *Advances in Neural Information Processing Systems*, pages 2897–2905, 2014.

K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878–1915, 2011.

S. Rota Bulo and M. Pelillo. A game-theoretic approach to hypergraph clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6):1312–1327, 2013.

G. Schweikert and B. W. Kernighan. A proper model for the partitioning of electrical circuits. In *9th Design Automation Workshop*, pages 57–62, Dallas, 1979.

A. Shashua, R. Zass, and T. Hazan. Multi-way clustering using super-symmetric non-negative tensor factorization. In *European Conference on Computer Vision*, pages 595–608, 2006.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

M. Soltanolkotabi, E. Elhamifar, and E. J. Candes. Robust subspace clustering. *The Annals of Statistics*, 42(2):669–699, 2014.

C. Tomasi and T. Kanade. Shape and motion from image streams under orthography. *International Journal of Computer Vision*, 9(2):137–154, 1992.

R. Tron and R. Vidal. A benchmark for the comparison of 3-D motion segmentation algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

U. von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

J. Wang, T. Jebara, and S.-F. Chang. Graph transduction via alternating minimization. In *International Conference on Machine Learning (ICML)*, pages 1144–1151, 2008.

Z. Zhao and H. Liu. Spectral feature selection for supervised and unsupervised learning. In *International Conference on Machine Learning (ICML)*, pages 1151–1157, 2007.

D. Zhou, J. Huang, and B. Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1601–1608, 2007.

Journal of Machine Learning Research 18 (2017) 1-24 Submitted 3/16; Revised 4/17; Published 5/17

Reconstructing Undirected Graphs from Eigenspaces

Yohann De Castro yohann.decastro@math.u-psud.fr
Laboratoire de Mathématiques d'Orsay, Univ. Paris-Sud, CNRS, Université Paris-Saclay, F-91405 Orsay, France

Thibault Espinasse espinasse@math.univ-lyon1.fr
Institut Camille Jordan (CNRS UMR 5208), Université Claude Bernard Lyon 1, F-69622 Villeurbanne, France

Paul Rochet paul.rochet@univ-nantes.fr
Laboratoire de Mathématiques Jean Leray (CNRS UMR 6629), Université de Nantes, F-44322 Nantes, France

Editor: Amir Globerson

Abstract

We aim at recovering the weighted adjacency matrix W of an undirected graph from a perturbed version of its eigenspaces. This situation arises for instance when working with stationary signals on graphs or Markov chains observed at random times. Our approach relies on minimizing a cost function based on the Frobenius norm of the commutator $\mathsf{AB}-\mathsf{BA}$ between symmetric matrices A and B. We describe a particular framework in which we have access to an estimation of the eigenspaces and provide support selection procedures from theoretical and practical points of view. In the Erdős-Rényi model on N vertices with no self-loops, we show that identifiability (*i.e.*, the ability to reconstruct W from the knowledge of its eigenspaces) follows a sharp phase transition on the expected number of edges with threshold function $N\log N/2$. Simulated and real life numerical experiments assert our methodology.

Keywords: Support recovery; Identifiability; Stationary signal processing; Graphs; Backward selection algorithm

1. Presentation

Networks have become a natural and popular way to model interactions in applications such as information technology (Rossi and Latouche, 2013), social life (Jiang et al., 2013; Matias et al., 2015), genetics (Giraud et al., 2012) or ecology (Thomas et al., 2015; Miele and Matias, 2017). In this paper, we investigate the reconstruction of an undirected weighted graph of size N from incomplete information on its set of edges (for instance, one knows that the target graph has no self-loops) and an estimation of the eigenspaces of its adjacency matrix W. This situation depicts any model where one knows in advance a linear operator K that commutes with W.

For instance, several authors (Espinasse et al., 2014; Girault, 2015; Perraudin and Vandergheynst, 2016; Marques et al., 2016) have introduced a definition of stationarity for signal processing on graphs. In the Gaussian framework, they have shown that this definition implies that the covariance operator K is jointly diagonalizable with the Laplacian (Perraudin and Vandergheynst, 2016) or some weighted symmetric adjacency matrix W supported on the graph (Espinasse et al., 2014; Marques et al., 2016).

Another framework adapted to our methodology concerns time-varying Markov processes, which are used to model numerous phenomena such as chemical reactions (Anderson and Kurtz, 2011) or waiting lines in queuing theory (Gaver Jr, 1959), see also Pittenger (1982); MacRae (1977); Barsotti et al. (2014). In some cases, one may observe at random times a Markov chain with transition matrix P. The transition matrix Q of the resulting Markov chain can be shown to be a function of P. Thus, the transitions on the original process can be recovered from an estimation of Q given that P and Q commute. Several models are presented in Section 3 while the general model is given in Section 2.1.

Section 2.2 is concerned with identifiability issues, *i.e.*, the capacity to solve such problems. We exhibit sufficient and necessary conditions on the ability to reconstruct an undirected graph with no self-loops from the knowledge of the eigenspaces of W. These conditions allow us to derive a sharp phase transition on identifiability in the Erdős-Rényi model.

In Section 4.1, we introduce and theoretically assert new estimation schemes based on the Frobenius norm of the commutator $\mathsf{AB}-\mathsf{BA}$ between symmetric matrices A and B. More precisely, we assume that we have access to an estimation $\widehat{\mathsf{K}}$ of K and we consider the empirical contrast given by the commutator, namely $\mathsf{A}\mapsto\|\widehat{\mathsf{K}}\mathsf{A}-\mathsf{A}\widehat{\mathsf{K}}\|$, where $\|\cdot\|$ denotes the Frobenius norm. Using backward-type procedures based on this empirical contrast, we build in Section 4 an estimator of the graph structure, *i.e.*, its set of edges $S^\star$ referred to as the support. Numerical experiments on simulated data (Section 5) and actual data (Section 6) assess the performances of our new estimation method. A discussion and related questions are presented in Section 7.

Related topics encompass spectral, least-squares and moment methods for graph reconstruction (Verzelen et al., 2015; Guédon and Vershynin, 2015; Klopp et al., 2017; Bubeck et al., 2016), Graphical Models (Verzelen, 2008; Giraud et al., 2012), or Vectorial AutoRegressive process (Hyvärinen et al., 2010). In the specific cases of Ornstein-Uhlenbeck processes and non-linear diffusions, the interesting papers Bento et al. (2010) and Bento and Ibrahimi (2014) tackle a related problem which is to estimate W along a trajectory, see Section 3.6 for further details. Note that the framework of the present paper addresses processes observed at random times—with possibly unknown distribution—which are not covered by Bento et al. (2010) and Bento and Ibrahimi (2014).

2. Model and Identifiability

2.1 The Model

Consider a symmetric matrix $\mathsf{W}\in\mathbb{R}^{N\times N}$, viewed as the weighted adjacency matrix of an undirected graph with N vertices. We investigate the eigenspaces of W in a situation where we have no direct information on the spectrum of the graph. Depicting this situation, we assume that the information on the target W stems from an unknown transformation $\mathsf{K}=f(\mathsf{W})\in\mathbb{R}^{N\times N}$ or, in more realistic scenarios, from a perturbed version $\widehat{\mathsf{K}}$ of K. Precisely, let $f:x\mapsto\sum_{n=0}^{\infty}a_nx^n$ be an injective function, analytical on the spectrum of W, the matrix K is given by

$$\mathsf{K}=f(\mathsf{W}):=\sum_{n=0}^{\infty}a_n\mathsf{W}^n. \tag{1}$$

In this setting, the transformation $\mathsf{K}=f(\mathsf{W})$ preserves the eigenspaces. In particular, W and K commute, *i.e.*, $\mathsf{WK}=\mathsf{KW}$, since they share the same eigenspaces.

Our goal is to reconstruct W from a perturbed observation of its eigenspaces, provided by an estimator $\widehat{\mathsf{K}}$ of K. The key point is then to use extra information given by the location of some zero entries of W. Hence, we assume that one knows in advance a set $F\subset[1,N]^2$ of "forbidden" entries such that

$$\forall(i,j)\in F,\quad \mathsf{W}_{ij}=0 \tag{$\mathbf{H}_\mathrm{F}$}$$

Equivalently, the set F is disjoint from the set of edges of the target graph. Throughout this paper, a special interest is given to the case $F=F_{\mathrm{diag}}:=\{(i,i)\ :\ 1\le i\le N\}$ conveying that there are no self-loops in W.

©2017 Yohann De Castro, Thibault Espinasse, and Paul Rochet.
License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-146.html.

2.2 Identifiability

For $S \subseteq [1,N]^2$, denote by $\mathcal{E}(S)$ the set of symmetric matrices A whose support is included in S, which we write $\mathrm{Supp}(\mathsf{A}) \subseteq S$. Given the set F of forbidden entries defined via ($\mathbf{H}_{\mathrm{F}}$), the matrix of interest W is sought in the set $\mathcal{E}(\overline{F})$ with $\overline{F}$ the complement of F. In some cases, typically for F sufficiently large, most matrices $\mathsf{W} \in \mathcal{E}(\overline{F})$ are uniquely determined by their eigenspaces. For those $\mathsf{W} \in \mathcal{E}(\overline{F})$, there is no matrix $\mathsf{A} \in \mathcal{E}(\overline{F})$ non collinear with W that commutes with W. This property is encapsulated by the notion of *F-identifiability* as follows.

Definition 1 (F-identifiability) *We say that a symmetric matrix W is F-identifiable if, and only if, the only solutions A with $\mathrm{Supp}(\mathsf{A}) \subseteq \overline{F}$ to $\mathsf{AW} = \mathsf{WA}$ are of the form $\mathsf{A} = t\mathsf{W}$ for some $t \in \mathbb{R}$. Equivalently,*

$$\left\{\mathsf{A} \in \mathbb{R}^{N\times N} \ : \ \mathsf{A} = \mathsf{A}^\top,\ \mathsf{AW} = \mathsf{WA} \text{ and } \mathrm{Supp}(\mathsf{A}) \subseteq \overline{F}\right\} = \left\{t\mathsf{W} \ : \ t \in \mathbb{R}\right\} \tag{2}$$

A matrix W is F-identifiable if the set of symmetric matrices with the same eigenvectors as W and whose support is included in $\overline{F}$ is the line spanned by W.

Remark 2 *The dimension of the commutant, defined by*

$$\mathrm{Com}(\mathsf{W}) := \left\{\mathsf{A} \in \mathbb{R}^{N\times N} \ : \ \mathsf{A} = \mathsf{A}^\top,\ \mathsf{AW} = \mathsf{WA}\right\},$$

is entirely determined by the multiplicity of the eigenvalues of W. Indeed, letting $\lambda_1, \dots, \lambda_s$ denote the different eigenvalues of W and $\ell_1, \dots, \ell_s$ their multiplicities, one can show that

$$N \leq \dim\left(\mathrm{Com}(\mathsf{W})\right) = \sum_{j=1}^{s} \frac{\ell_j(\ell_j+1)}{2} \leq \frac{N(N+1)}{2}.$$

Now, the F-identifiability of W can be stated equivalently as $\dim\left(\mathrm{Com}(\mathsf{W}) \cap \mathcal{E}(\overline{F})\right) = 1$, observing that the left hand side of (2) is exactly $\mathrm{Com}(\mathsf{W}) \cap \mathcal{E}(\overline{F})$. Using a simple inclusion/exclusion formula, one can check that the condition

$$|F| \geq \dim\left(\mathrm{Com}(\mathsf{W})\right) - 1$$

is necessary for the F-identifiability, where $|F|$ denotes the cardinality of F. In particular, a matrix W with repeated eigenvalues requires a large set F of forbidden entries to be F-identifiable.

Proposition 3 (Lemma 2.1 in Barsotti et al. (2014)) *Let $S \subseteq \overline{F}$, the set of F-identifiable matrices in $\mathcal{E}(S)$ is either empty or a dense open subset of $\mathcal{E}(S)$.*

This proposition conveys that the F-identifiability of a matrix W is essentially a condition on its support S. The proof uses the fact that non F-identifiable matrices in $\mathcal{E}(S)$ can be expressed as the zeros of a particular analytic function, we refer to Barsotti et al. (2014) for further details. By abuse of notation, we say that a support $S \subseteq \overline{F}$ is *F-identifiable* if almost every matrix in $\mathcal{E}(S)$ is F-identifiable.

Characterizing the F-identifiability appears to be a challenging issue since it can be viewed as understanding the eigen-structure of graphs through their common support. The special case of the diagonal F_{diag} as the set of forbidden entries turns out to be particularly interesting. Indeed, the F_{diag}-identifiability, or diagonal identifiability, can be reasonably assumed in many practical situations since it entails that W lives on a simple graph, with no self-loops. In Theorem 16 (see Appendix A.1), we introduce necessary and sufficient conditions on the target support $\mathrm{Supp}(\mathsf{W})$ for diagonal identifiability. Defining the *kite* graph ∇_N of size $N \geq 3$ as the graph (V,E) with vertices $V = [1,N]$ and edges $E = \{(k,k+1),\ 1 \leq k \leq N-1\} \cup \{(N-2,N)\}$ (see Figure 1), one simple sufficient condition on diagonal identifiability reads as follows—a proof in given in Section A.2.

Proposition 4 *If the graph $G = ([1,N],S)$ contains the kite graph ∇_N as a subgraph, then S is diagonally identifiable.*

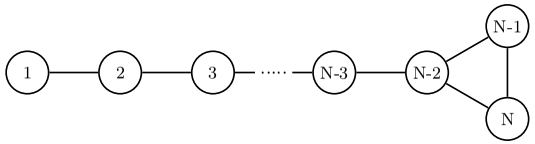


Figure 1: The kite graph ∇_N on N vertices.

Denote $G(N,p)$ the Erdős-Rényi model on graphs of size N where the edges are drawn independently with respect to the Bernoulli law of parameter p. Using Theorem 16, one can prove that $\log N/N$ is a sharp threshold for diagonal identifiability in the Erdős-Rényi model (see Section A.4). This can be stated as follows.

Theorem 5 *Diagonal identifiability in the Erdős-Rényi model occurs with a sharp phase transition with threshold function $\log N/N$: for any $\varepsilon > 0$, it holds*

- *If $p_N \geq (1+\varepsilon)\log N/N$ and $G_N \sim G(N,p_N)$ then the probability that $\mathrm{Supp}(G_N)$ is diagonally identifiable tends to 1 as N goes to infinity.*
- *If $p_N \leq (1-\varepsilon)\log N/N$ and $G_N \sim G(N,p_N)$ then the probability that $\mathrm{Supp}(G_N)$ is diagonally identifiable tends to 0 as N goes to infinity.*

In practice, one may expect that any target graph of size N with no self-loops and degree bounded from below by $\log N$ is diagonally identifiable. In this case, it might be recovered from its eigenspaces. Conversely, small degree graphs (*i.e.*, graphs with some vertices of degree much smaller than $\log N$) may not be identifiable. In this case, there is no hope to reconstruct it from its eigenspaces since there exists another small degree undirected weighted graph with the same eigenspaces.

3. Some Concrete Models

3.1 Markov chains

We begin with an example treated in the companion papers Barsotti et al. (2014, 2016). Consider a Markov chain $(X_n)_{n\in\mathbb{N}}$ with finite state space $[1,N]$ and transition matrix $\mathsf{P} \in \mathbb{R}^{N\times N}$. Let $(T_k)_{k\geq 1}$ be a sequence of random times such that the time gaps $\tau_k := T_{k+1} - T_k$ are i.i.d random variables independent of $(X_n)_{n\in\mathbb{N}}$. One can show that the sequence $Y_k = X_{T_k}$ is also a Markov chain with transition matrix $\mathsf{Q} = \mathbb{E}[\mathsf{P}^{\tau_1}] =: f(\mathsf{P})$ where f is the generating function of τ_k. Indeed, this follows from noticing that

$$\begin{aligned}
\mathbb{P}[Y_{k+1} = j | Y_k = i] &= \mathbb{P}[X_{T_{k+1}} = j | X_{T_k} = i] \\
&= \sum_{t\geq 0} \mathbb{P}[X_{T_k+t} = j, \tau_k = t | X_{T_k} = i] \\
&= \sum_{t\geq 0} \mathbb{P}[X_t = j | X_0 = i]\mathbb{P}[\tau_k = t] \\
&= \sum_{t\geq 0} \mathbb{P}[\tau_1 = t](\mathsf{P}^t)_{ij}.
\end{aligned}$$

Under regularity conditions, $\mathsf{Q} = f(\mathsf{P})$ can be estimated from the Y_k's and one may recover P from Q without any information on the distribution of the time gaps τ_k.

3.2 Vectorial AutoRegressive process

Consider a stationary Vectorial AutoRegressive process of order one $(X_n)_{n\in\mathbb{Z}}$ verifying

$$X_{n+1} = \mathsf{W}X_n + \varepsilon_n\,,$$

with ε_i i.i.d. centered random variables. Define as above $Y_k = X_{T_k}$ where again T_k are random times such that the time gaps $\tau_k = T_{k+1} - T_k$ are i.i.d. with generating function f and independent of $(X_n)_{n\in\mathbb{Z}}$. Then, it holds

$$\mathbb{E}[Y_{k+1}|Y_k] = \mathbb{E}[\mathbb{E}[Y_{k+1}|Y_k,\tau_k]|Y_k] = \sum_{j=0}^{\infty} \mathsf{W}^j Y_k \mathbb{P}(\tau_k = j) = f(\mathsf{W})Y_k,$$

which allows us to estimate $\mathsf{K} = f(\mathsf{W})$ and ultimately recover W.

3.3 Ornstein-Uhlenbeck process

The same property holds for the continuous time version of this process, namely a vectorial Ornstein-Uhlenbeck process observed at random times verifying

$$\mathrm{d}X_t = \mathsf{W}X_t\mathrm{d}t + \mathrm{d}B_t.$$

In this case, one can check that the random process $Y_k := X_{T_k}$ where the T_k's are random times with i.i.d. gaps $\tau_k = T_{k+1} - T_k$ satisfies

$$\mathbb{E}[Y_{k+1}|Y_k] = f(\mathsf{W})Y_k\,,$$

for f the Laplace transform of τ_1, that is, $f(\mathsf{W}) = \mathbb{E}[\exp(-\tau_1\mathsf{W})]$. Indeed, note that

$$\forall t,u\in\mathbb{R},\quad \mathbb{E}[X_{t+u}|X_u] = \exp(-t\mathsf{W})X_u$$

so that

$$\mathbb{E}[Y_{k+1}|Y_k] = \mathbb{E}[\mathbb{E}[X_{T_k+\tau_k}|X_{T_k},\tau_k]|X_{T_k}] = \mathbb{E}[\exp(-\tau_k\mathsf{W})X_{T_k}|X_{T_k}] = \mathbb{E}[\exp(-\tau_k\mathsf{W})]Y_k\,,$$

by independence of τ_k and Y_{k-1}.

3.4 Seasonal VAR structure

Consider a seasonal VAR structure: let T be a positive integer, $(u_k)_{k\in\mathbb{Z}}, (v_k)_{k\in\mathbb{Z}}$ periodic sequences of period T and

$$\forall k\in\mathbb{Z}\,,\quad Y_{k+1} = u_kY_k + v_k\mathsf{W}Y_k + \varepsilon_k\,,$$

where ε_k are independent and centered random variables. We may observe the model only at time gap intervals T with some error, *i.e.*, $X_t = Y_{tT+k_0} + \eta_t$ with η_t centered and independent random variables. This falls into the general frame

$$\mathbb{E}[X_t|X_{t-1}] = f(\mathsf{W})X_{t-1}\quad\text{where}\quad f(x) := \prod_{k=1}^{T}(u_k - v_kx)\,.$$

In this case, $\mathsf{K} = f(\mathsf{W})$ can be estimated from the observations.

3.5 Gaussian Graphical models

Our model is related to Gaussian Graphical models for which an overview can be found in the thesis Verzelen (2008). The reader may also consult the pioneering paper Friedman et al. (2008). One may consider the target W as the precision matrix, *i.e.*, the inverse of the covariance matrix K, having some non zero entries described by a graph of dependencies. Using $f(x) = x^{-1}$, this falls into our setting, trying to recover the "dependency" graph given by the precision matrix W from the estimation of the covariance matrix K. Of course, in this case, it is better to use the knowledge of f, which certainly improves estimation. Nevertheless, our procedure allows us to estimate the function f and heuristically validate the hypothesis $f(x) = x^{-1}$.

3.6 Spatial AutoRegressive Gaussian fields

Note that Gaussian AutoRegressive processes on $\mathbb{Z}$ verify that the precision operator may be written as a polynomial of the adjacency operator of $\mathbb{Z}$. One natural way to extend this property (see for instance Espinasse et al. (2014)) is to define centered Gaussian AutoRegressive fields on a graph through the same relation between the covariance operator K and the adjacency operator W (or the discrete Laplacian, depending on the framework) : $\mathsf{K}^{-1} = P(\mathsf{W})$, with P a polynomial of degree p. In this framework, Graphical models methods will infer the graph of path of length p, whereas our method aims at recovering W. Note that this framework extends to ARMA spatial fields where K writes as a rational fraction of W, and the property of commutativity between W and K still holds.

In the previous cases, we assumed that we can not estimate directly W. For spatio-temporal processes, this means that we do not have access to a full trajectory. It may be the case when the sample is drawn at random times, or when we sample with respect to the stationary measure of the process—for instance when observation times are a lot larger than the typical evolution time's scale of the process. If the whole trajectory is available, it is better to use this extra information, see for instance Bento et al. (2010) for the Ornstein-Uhlenbeck case and Bento and Ibrahimi (2014) for the non-linear diffusion case.

4. Estimating the Support

4.1 Empirical Contrast: the Commutator

The methodology presented in the paper relies on the fact that the target matrix W commutes with the matrix K, in view of $\mathsf{K} := f(\mathsf{W})$, as defined in Eq. (1). Because W is symmetric, it has real eigenvalues $\lambda_1,...,\lambda_N$ (here listed with repetitions, if any) and is diagonalizable in an orthogonal basis. That is, letting Λ denote the diagonal matrix with diagonal entries λ_k, there exists an orthogonal matrix U such that $\mathsf{W} = \mathsf{U}\Lambda\mathsf{U}^\top$. With this notation, one verifies easily from Eq. (1) that $\mathsf{K} = \mathsf{U}\mathsf{D}\mathsf{U}^\top$, where $\mathsf{D} := f(\Lambda)$ is the diagonal matrix with diagonal entries $f(\lambda_i), i = 1,\dots,N$. Since f is assumed one-to-one on the spectrum of W, the matrices W and K share the same eigenspaces associated to λ_k and $f(\lambda_k)$ respectively :

$$\{v\in\mathbb{R}^N : \mathsf{W}v = \lambda_k v\} = \{v\in\mathbb{R}^N : \mathsf{K}v = f(\lambda_k)v\},$$

for all λ_k. Moreover, the dimension of each eigenspace is equal to the multiplicity of the corresponding eigenvalue λ_k in the spectrum of W. Additionally, when F-identifiability holds, the only solutions A with $\mathrm{Supp}(\mathsf{A}) \subseteq \overline{F}$ to $\mathsf{A}\mathsf{K} = \mathsf{K}\mathsf{A}$ are of the form $\mathsf{A} = t\mathsf{W}$ for some $t\in\mathbb{R}$.

Remark 6 (Matrix perturbation theory) *In practice, we do not observe* K *but an estimation* $\widehat{\mathsf{K}}$ *(which we assume symmetric) with perturbed eigen-decomposition* $\widehat{\mathsf{K}} = \widehat{\mathsf{U}}\widehat{\mathsf{D}}\widehat{\mathsf{U}}^\top$. *Nevertheless, by continuity of the eigen-decomposition, we know that* $\widehat{\mathsf{K}}$ *being close to* K *implies*

simultaneously the proximity of their eigenvalues and eigenspaces (see for instance Mirsky's inequality (Stewart and Sun, 1990, Corollary 4.12) and Wedin's $\sin(\theta)$ theorem (Stewart and Sun, 1990, P. 260) for the details). Thus, if we consider A such that $\mathsf{A}\widehat{\mathsf{K}} = \widehat{\mathsf{K}}\mathsf{A}$, then A has the same eigenspaces as $\widehat{\mathsf{K}}$ which in turns, are expected to be close to the eigenspaces of W.

Given an estimator $\widehat{\mathsf{K}}$ of K, remark that W verifies

$$\frac{\|\mathsf{W}\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\mathsf{W}\|}{\|\mathsf{W}\|} = \frac{\|\mathsf{W}(\widehat{\mathsf{K}} - \mathsf{K}) - (\widehat{\mathsf{K}} - \mathsf{K})\mathsf{W}\|}{\|\mathsf{W}\|} \leq 2\|\widehat{\mathsf{K}} - \mathsf{K}\|\,. \tag{3}$$

Hence, in view of (3) and the discussion above, we aim to estimate W by minimizing the following cost function

$$\mathsf{A} \mapsto \frac{\|\mathsf{A}\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\mathsf{A}\|}{\|\mathsf{A}\|}, \quad \mathsf{A} \in \mathcal{E}(\overline{F}) \setminus \{0\}.$$

This empirical criterion was first used in Barsotti et al. (2014), in a Markov Chain context, to reflect that W is expected to nearly commute with $\widehat{\mathsf{K}}$.

4.2 The ℓ_0-approach

Given an estimator $\widehat{\mathsf{K}}$ of K build from a sample of size n and a set of forbidden entries F reflecting ($\mathbf{H}_{\mathrm{F}}$), we construct an estimator $\widehat{S}$ of the target support $S^\star := \mathrm{Supp}(\mathsf{W})$ as a minimizer of the criterion Q given by

$$\forall S \subseteq \overline{F}, \quad Q(S) := \min_{\mathsf{A} \in \mathcal{E}(S)\setminus\{0\}} \frac{\|\mathsf{A}\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\mathsf{A}\|}{\|\mathsf{A}\|} + \lambda_n |S|,$$

for some tuning parameter $\lambda_n > 0$ and defining the minimum of an empty set as ∞. Recall that $\mathcal{E}(S)$ is the set of symmetric matrices A such that $\mathrm{Supp}(\mathsf{A}) \subseteq S$. Our estimator of the true support $S^\star$ is defined as

$$\widehat{S} \in \arg\min_{S \subseteq \overline{F}} Q(S)$$

Furthermore, we assume that the estimator $\widehat{\mathsf{K}}$ converges toward K in probability with

$$\forall t > 0\,, \quad \mathbb{P}\big\{\|\widehat{\mathsf{K}} - \mathsf{K}\| \geq t\big\} \leq R_n(t), \tag{$\mathbf{H}_2$}$$

where $R_n, n \in \mathbb{N}$ is a sequence of non-increasing functions that converge pointwise toward 0 as n goes to ∞.

Theorem 7 *Assume that ($\mathbf{H}_2$) and ($\mathbf{H}_{\mathrm{F}}$) hold. If W is F-identifiable, then*

$$\mathbb{P}\big\{\widehat{S} \neq S^\star\big\} \leq R_n\Big(\frac{c_0(S^\star) - \lambda_n |S^\star|}{4}\Big) + R_n\Big(\frac{\lambda_n}{2}\Big)\,,$$

where

$$c_0(S^\star) := \min_{\substack{S \neq S^\star \\ |S| \leq |S^\star|}} \min_{\mathsf{A} \in \mathcal{E}(S)} \frac{\|\mathsf{A}\mathsf{K} - \mathsf{K}\mathsf{A}\|}{\|\mathsf{A}\|} > 0\,. \tag{4}$$

A proof of Theorem 7 is given in Section B.1.

Corollary 8 *Under the assumptions of Theorem 7, if*

$$\lambda_n \to 0 \quad \text{and} \quad \sum_{n \in \mathbb{N}} R_n\Big(\frac{\lambda_n}{2}\Big) < +\infty\,,$$

then $\widehat{S} \to S^\star$ almost surely.

Note that, based on the upper bound in Theorem 7, a good scaling may be $\lambda_n^\star = \frac{c_0(S^\star)}{|S^\star|+4}$ which interestingly does not depend on n. This leads to the upper bound

$$\mathbb{P}\big\{\widehat{S} \neq S^\star\big\} \leq 2R_n\Big(\frac{c_0(S^\star)}{2|S^\star| + 8}\Big) \xrightarrow{n\to\infty} 0$$

which is optimal up to a constant less than 2. This oracle choice $\lambda_n^\star$ is of course irrelevant in practice since both $c_0(S^\star)$ and $|S^\star|$ are unknown. Alternatively, we may choose a sequence λ_n decreasing slowly to 0 to ensure both conditions of Corollary 8.

4.3 Edge significance based on the commutator criterion

The exponential complexity of the ℓ_0-approach making it generally infeasible in practice, a backward methodology provides a computationally feasible alternative to the support reconstruction problem. Starting from the maximal acceptable support $\overline{F}$, the idea of the backward procedure is to remove the least significant entries one at a time and stop when every entry is significant. Using the corresponding small case letter to denote the vectorization of a matrix, *e.g.*, $a = \mathrm{vec}(\mathsf{A}) = (\mathsf{A}_{11}, ..., \mathsf{A}_{N1}, ..., \mathsf{A}_{1N}, ..., \mathsf{A}_{NN})^\top$, significancy can be leveraged using the Frobenius norm of the commutator operator $a \mapsto \Delta(\mathsf{K})a = \mathrm{vec}(\mathsf{K}\mathsf{A} - \mathsf{A}\mathsf{K})$, where

$$\Delta(\mathsf{K}) = \mathrm{I} \otimes \mathsf{K} - \mathsf{K} \otimes \mathrm{I} \quad \in \mathbb{R}^{N^2 \times N^2}$$

and $\otimes$ denotes the Kronecker product. Indeed, searching for the target W in the commutant of K reduces to searching for $w = \mathrm{vec}(\mathsf{W})$ in $\ker(\Delta(\mathsf{K}))$, the kernel of $\Delta(\mathsf{K})$. Because the Frobenius norm coincides with the Euclidean norm of the vectorization, the functions $\mathsf{A} \mapsto \|\widehat{\mathsf{K}}\mathsf{A} - \mathsf{A}\widehat{\mathsf{K}}\|^2$ and $a \mapsto \|\Delta(\widehat{\mathsf{K}})a\|^2$ can be used indistinctly as cost functions.

Assumptions

Assume the three following hypotheses ($\mathbf{H}_\Sigma$), ($\mathbf{H}_1$) and ($\mathbf{H}_{\mathrm{Id}}$).

∘ Deriving the asymptotic law of least-squares estimators, we may assume that the estimate $\widehat{\mathsf{K}}$ is such that

$$\sqrt{n}(\widehat{k} - k) \xrightarrow[n\to\infty]{d} \mathcal{N}(0, \Sigma), \tag{$\mathbf{H}_\Sigma$}$$

where Σ is a $N^2 \times N^2$ covariance matrix (either known or that can be estimated). For instance, one can think of $\widehat{\mathsf{K}}$ as the empirical covariance when observing a sample of vectors of covariance K. This condition is verified for instance in the framework considered in Barsotti et al. (2014, 2016). Note that asymptotic normality is a standard ground base investigating any least-squares procedure.

∘ In order to exclude the trivial solution $a = 0$, the target W is assumed normalized

$$\mathbf{1}^\top w = 1, \tag{$\mathbf{H}_1$}$$

where $\mathbf{1}$ has all its entries equal to one. Because the available information on W is of spectral nature and as such, is scale-invariant, a normalization of some kind is crucial for the reconstruction. Here, the condition $\mathbf{1}^\top w = 1$ achieves two goals: preventing the null matrix form being a solution and making the problem identifiable.

Remark 9 *The main drawback of this normalization concerns the situation where the entries of W sum up to zero, in which case the normalization is impossible. If the context suggests that the solution may be such that $\mathbf{1}^\top w = 0$, a different affine normalization $\mathbf{v}^\top w = 1$ (with any fixed vector $\mathbf{v}$) must be used, without major changes in the methodology. In practice, one may consider the vector $\mathbf{v}$ at random (for instance with isotropic law), so that ($\mathbf{H}_1$) is almost surely fulfilled for any fixed target w. Finally, observe that if W has non-negative entries, then the normalization ($\mathbf{H}_1$) is always feasible.*

◦ For S a support included in $\overline{F}$, we aim at a solution in the affine space

$$\mathcal{A}_S := \{a = \mathrm{vec}(\mathsf{A}) : \mathrm{Supp}(\mathsf{A}) \subseteq S,\ \mathsf{A} = \mathsf{A}^\top\ ,\ \mathbf{1}^\top a = 1\}.$$

with linear difference space given by

$$\mathcal{L}_S := \{a = \mathrm{vec}(\mathsf{A}) : \mathrm{Supp}(\mathsf{A}) \subseteq S,\ \mathsf{A} = \mathsf{A}^\top\ ,\ \mathbf{1}^\top a = 0\}\,.$$

By abuse of notation, $\mathcal{A}_S$ may refer both to the space of matrices or their vectorizations. To find the target support $S^\star$, one must exploit the fact that the vector w lies in the intersection of $\ker(\Delta(\mathsf{K}))$ and $\mathcal{A}_{\overline{F}}$. Actually, w can be recovered if the intersection is reduced to the singleton $\{w\}$. In this case, the matrix W and its support $S^\star$ are F-identifiable. Hence, we assume that

$$\ker(\Delta(\mathsf{K})) \cap \mathcal{L}_{\overline{F}} = \{0\}, \tag{$\mathbf{H}_{\mathrm{Id}}$}$$

which is implied by F-identifiability, see Definition 1.

Asymptotic normality and significance test

The framework under consideration can be viewed as a heteroscedastic linear regression model with noisy design for which $w = \mathrm{vec}(\mathsf{W})$ is the parameter of interest. Indeed, consider for each support $S \subseteq \overline{F}$ a full-ranked matrix $\Phi_S \in \mathbb{R}^{N^2 \times \dim(\mathcal{A}_S)}$ whose column vectors form a basis of $\mathcal{L}_S$. Assuming that W is F-identifiable and taking $S \subseteq \overline{F}$, the operator $\Delta(\mathsf{K})\Phi_S$ is one-to-one. In this case, evaluating the commutator $a \mapsto \Delta(\mathsf{K})a$ over $\mathcal{A}_S$ reduces to considering the map

$$b \mapsto \Delta(\mathsf{K})(a_0 - \Phi_S b)\,, \quad b \in \mathbb{R}^{\dim(\mathcal{A}_S)},$$

with a_0 chosen arbitrarily in $\mathcal{A}_S$. When replacing the unknown $\Delta(\mathsf{K})$ with its estimate $\Delta(\widehat{\mathsf{K}})$, the minimization of the criterion $a \mapsto \|\Delta(\widehat{\mathsf{K}})a\|^2$ over $\mathcal{A}_S$ can be written similarly as a linear regression framework where the parameter of interest is estimated by

$$\widehat{\beta}_S \in \arg\min_{b \in \mathbb{R}^{\dim(\mathcal{A}_S)}} \|\Delta(\widehat{\mathsf{K}})(a_0 - \Phi_S b)\|^2. \tag{5}$$

We recognize a linear model with response $y = \Delta(\widehat{\mathsf{K}})a_0$ and noisy design matrix $X = \Delta(\widehat{\mathsf{K}})\Phi_S$. In this setting, remark that $w = a_0 - \Phi_S\beta$ with β the unique solution to $\Delta(\mathsf{K})(a_0 - \Phi_S\beta) = 0$. Denoting by $\mathsf{M}^\dagger$ the pseudo-inverse of a matrix M, we deduce the following result.

Theorem 10 *If $S^\star \subseteq S$, the estimator $\widehat{\beta}_S$ is asymptotically Gaussian with*

$$\sqrt{n}(\widehat{\beta}_S - \beta) \xrightarrow[n\to\infty]{d} \mathcal{N}(0, \Omega_S),$$

where $\Omega_S := \big(\Phi_S^\top \Delta(\mathsf{K})\big)^\dagger \Delta(\mathsf{W})\Sigma\Delta(\mathsf{W})\big(\Delta(\mathsf{K})\Phi_S\big)^\dagger$.

We then have

$$\widehat{w}_S = \mathrm{vec}(\widehat{\mathsf{W}}_S) = \arg\min_{a \in \mathcal{A}_S} \|\Delta(\widehat{\mathsf{K}})a\|^2 = a_0 - \Phi_S\widehat{\beta}_S\,. \tag{6}$$

The asymptotic distribution of $\widehat{w}_S$ follows directly from Theorem 10,

$$\sqrt{n}(\widehat{w}_s - w) \xrightarrow[n\to\infty]{d} \mathcal{N}\big(0, \Phi_S\Omega_S\Phi_S^\top\big). \tag{7}$$

The limit covariance matrix is unknown, but plugging the estimates $\widehat{\mathsf{W}}_S$, $\widehat{\mathsf{K}}$ and $\widehat{\Sigma}$ yields an estimator $\Phi_S\widehat{\Omega}_S\Phi_S^\top$, which is consistent under the F-identifiability assumption. In particular,

the diagonal entry of $\Phi_S\widehat{\Omega}_S\Phi_S^\top$ associated to the (i,j)-entry of W, which we denote $\widehat{\sigma}^2_{S,ij}$, provides a consistent estimator for the asymptotic variance of $\widehat{\mathsf{W}}_{S,ij}$. As a result, the statistic

$$\tau_{ij}(S) := \sqrt{n}\frac{\widehat{\mathsf{W}}_{S,ij}}{\widehat{\sigma}_{S,ij}} \tag{8}$$

can be used to measure the relative significance of the estimated entry $\widehat{\mathsf{W}}_{S,ij}$. The backward support selection procedure is then implemented by the recursive algorithm as follows.

Algorithm 1: Backward algorithm for support selection

Data: A set of forbidden entries F, a matrix $\widehat{\mathsf{K}}$.
Result: A sequence of estimators $\widehat{\mathsf{W}}_{S_1}, \widehat{\mathsf{W}}_{S_2}, \ldots$ with nested supports $S_1 \supset S_2 \supset \ldots$.

1: Start with the maximal acceptable support $S_1 = \overline{F}$,
2: At each step k, compute the statistics $\tau_{ij}(S_k)$ for all $(i,j) \in S_k$,
3: Remove the least significant edge (i,j) which minimizes $|\tau_{ij}(S_k)|$ for $(i,j) \in S_k$, and set $S_{k+1} = S_k \setminus \{(i,j),(j,i)\}$,
4: Stop when all edges have been removed.

The backward algorithm produces a sequence of nested supports that one can choose to stop once all the edges are judged significant, that is, when all the statistics $\tau_{ij}(S_k)$, $(i,j) \in S_k$ exceed in absolute value some fixed threshold τ_0. Owing to the asymptotic normality of $\widehat{\mathsf{W}}_{S,ij}$ shown in Eq. (7), the $(1 - \frac{\alpha}{2})$-quantile of the standard Gaussian distribution would appear as a reasonable choice for the threshold τ_0, as it boils down to performing an asymptotic significance test of level α. However, due to the slow convergence to the limit distribution and the tendency to overestimate the variance for small sample sizes (see Figure 2), a threshold based on the Gaussian quantile inevitably leads to an overly large estimated support. Nevertheless, we show that an adaptive calibration of the threshold can be achieved by considering the overall behavior of the commutator $\Delta(\widehat{\mathsf{K}})\widehat{w}_{S_m}$ computed over the nested sequence of active supports.

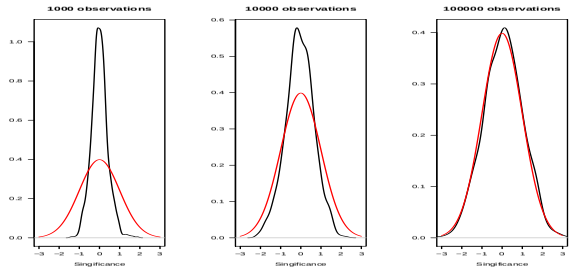


Figure 2: Estimated density of the statistic $\tau_{ij}(S)$ for an edge $(i,j) \in S \setminus S^\star$ compared to its theoretical Gaussian limit distribution, for samples of size $n = 1000$ (left), $n = 10000$ (center) and $n = 100000$ (right).

Calibration of the threshold by cross-validation

By removing the least significant edge at each step, the backward algorithm generates a sequence of nested active supports $S_1 \supset \cdots \supset S_\ell$, that we refer to as a "trajectory". Along this trajectory, we compute the empirical contrast defined by

$$\forall S \subseteq \overline{F}, \quad S \mapsto \mathrm{Crit}(\widehat{\mathsf{W}}_S, \widehat{\mathsf{K}}) := \frac{\|\widehat{\mathsf{W}}_S\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\widehat{\mathsf{W}}_S\|}{\|\widehat{\mathsf{W}}_S\|}\,. \tag{9}$$

Note that computing this criterion boils down to compute $\widehat{\mathsf{W}}_S$ which is a simple projection onto $\mathcal{A}_S$ as shown in (6).

When the true support $S^\star$ lies in the trajectory, one expects to observe a "gap" in the sequence $j \mapsto \mathrm{Crit}(\widehat{\mathsf{W}}_{S_j}, \widehat{\mathsf{K}})$ when S_j goes from $S^\star$ to a smaller support. Indeed:

- For $S^\star \subseteq S$, the target W is consistently estimated by $\widehat{\mathsf{W}}_S$ so that $\mathrm{Crit}(\widehat{\mathsf{W}}_S, \widehat{\mathsf{K}})$ tends to zero at rate $\sqrt{n}$,
- For $S \subsetneq S^\star$, the lower bound $\|\mathsf{A}\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\mathsf{A}\| \geq \|\mathsf{A}\mathsf{K} - \mathsf{K}\mathsf{A}\| - 2\|\widehat{\mathsf{K}} - \mathsf{K}\|\|\mathsf{A}\|$ yields

$$\mathrm{Crit}(\widehat{\mathsf{W}}_S, \widehat{\mathsf{K}}) = \frac{\|\widehat{\mathsf{W}}_S\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\widehat{\mathsf{W}}_S\|}{\|\widehat{\mathsf{W}}_S\|} \geq c(S) - 2\|\widehat{\mathsf{K}} - \mathsf{K}\| \tag{10}$$

  with $c(S) := \min_{A \in \mathcal{A}_S} \|\mathsf{A}\mathsf{K} - \mathsf{K}\mathsf{A}\|/\|\mathsf{A}\|$ a positive constant. In particular, one has

$$\min_{S \subsetneq S^\star} c(S) \geq \min_{\substack{S \neq S^\star \\ |S| \leq |S^\star|}} c(S) = c_0(S^\star) > 0$$

  where $c_0(S^\star)$ is defined in (4).

In some way, $c_0(S^\star)$ measures the amplitude of the signal: one expects to be able to recover the target W when the estimation error $\|\widehat{\mathsf{K}} - \mathsf{K}\|$ reaches at least the same order as $c_0(S^\star)$. The true support $S^\star$ then corresponds to a transitional gap in the contrast curve that can be captured by a suitably chosen threshold $t > 0$. Since $\widehat{\mathsf{K}}$ converges toward K in probability, any threshold $0 < t < c_0(S^\star)$ will work with probability one asymptotically.

Remark 11 *The condition that $S^\star$ lies in the trajectory of nested supports is crucial to detect the commutation gap, although seldom verified in practice due to the tremendous amount of testable supports. This issue is specifically targeted by the bagging version of the backward algorithm discussed in Section 4.4.*

An obstacle to the detection of the commutation gap is the increasing behavior of the commutator over the nested trajectory $S_1 \supset \cdots \supset S_\ell$. This phenomenon, indirectly caused by the dependence between the trajectory and $\widehat{\mathsf{K}}$, can be annihilated when considering the empirical contrast over a trajectory built from a training sample. In fact, the monotonicity can even be "reversed" before reaching the true support if the $\widehat{\mathsf{W}}_{S_j}$ are estimated independently from $\widehat{\mathsf{K}}$. This can be explained as follows. Consider the ideal scenario where estimators $\widetilde{\mathsf{W}}_{S_1}, ..., \widetilde{\mathsf{W}}_{S_\ell}$ are built from the backward algorithm using an estimator $\widetilde{\mathsf{K}}$ independent from $\widehat{\mathsf{K}}$. We assume moreover that the true support $S^\star$ lies in the trajectory $S_1 \supset ... \supset S_\ell$. The trick is to write

$$\Delta(\widehat{\mathsf{K}})\widetilde{w}_{s_j} = \Delta(\widetilde{\mathsf{K}})w + \Delta(\mathsf{K})\widetilde{w}_{S_j} + \Delta(\widehat{\mathsf{K}} - \mathsf{K})(\widetilde{w}_{S_j} - w),$$

and to analyze the three terms separately:

- The term $\Delta(\widehat{\mathsf{K}})w$ has no influence as it is common to all supports in the trajectory.

- The term $\Delta(\mathsf{K})\widetilde{w}_{S_j}$ approaches zero as $\widetilde{w}_{S_j}$ gets closer to w. Heuristically, the variance of $\widetilde{w}_{S_j}$, and incidentally that of $\Delta(\mathsf{K})\widetilde{w}_{S_j}$, is larger for over-fitting supports $S \supseteq S^\star$. This results in the sequence $j \mapsto \Delta(\mathsf{K})\widetilde{w}_{S_j}$ being stochastically decreasing as S_j approaches $S^\star$ from above. On the other hand, the bias of order $O(1)$ is expected to dominate once the trajectory passes through the true value $S^\star$, making the remaining of the sequence $\Delta(\mathsf{K})\widetilde{w}_{S_j}$increase stochastically.
- The term $\Delta(\widehat{\mathsf{K}} - \mathsf{K})(\widetilde{w}_{S_j} - w)$ is negligible for $S \supseteq S^\star$, as both $\widehat{\mathsf{K}} - \mathsf{K}$ and $\widetilde{w}_{S_j} - w$ tend to zero independently. We emphasize that this argument no longer holds without the independence of $\widetilde{w}_{S_j}$ and $\widehat{\mathsf{K}}$. This is precisely why we use a training sample.

Thus, the sequence $j \mapsto \mathrm{Crit}(\widetilde{\mathsf{W}}_{S_j}, \widehat{\mathsf{K}}) = \|\Delta(\widehat{\mathsf{K}})\widetilde{w}_{S_j}\|/\|\widetilde{w}_{S_j}\|$ is expected to achieve its minimum for the best estimator $\widetilde{w}_{S_j}$ in the trajectory, that is for $S_j = S^\star$. Furthermore, beyond the true support (for small active supports), $\widetilde{w}_{S_j}$ is not a consistent estimator of w so that the criterion no longer approaches zero, resulting in the so-called commutation gap.

The "reversed" monotonicity provides an easy way to calibrate the threshold in the backward algorithm. Indeed, since $S_j \mapsto \Delta(\widehat{\mathsf{K}})\widetilde{w}_{S_j}$ is expected to decrease when approaching the true support (coming from larger active supports along a trajectory), the estimated support can be heuristically chosen as the last time the criterion is below an adaptive threshold, see Figure 3. In particular, $\mathrm{Crit}(\widetilde{\mathsf{W}}_{S_1}, \widehat{\mathsf{K}})$ can be used as an adaptive threshold for the backward algorithm when the estimator $\widehat{\mathsf{K}}$ and the trajectory $S_1 \supset \cdots \supset S_\ell$ are obtained from independent samples.

Of course, to afford splitting the sample to build the $\widetilde{\mathsf{W}}_{S_j}$ independent from $\widehat{\mathsf{K}}$ may be unrealistic. Nevertheless, the numerical study suggests that the independence is well mimicked when $\widehat{\mathsf{K}}$ is built from the whole dataset but the backward algorithm sequence $\widetilde{\mathsf{W}}_{S_1}, ..., \widetilde{\mathsf{W}}_{S_\ell}$ is obtained from a learning sub-sample, as illustrated in Figure 3. Empirically, the optimal size of training samples could be calibrated in function of the number of observations using the robustness of the outputs of the algorithm. In this paper, we always draw training samples by taking each observation with probability 1/2, with no consideration regarding the size of the whole sample.

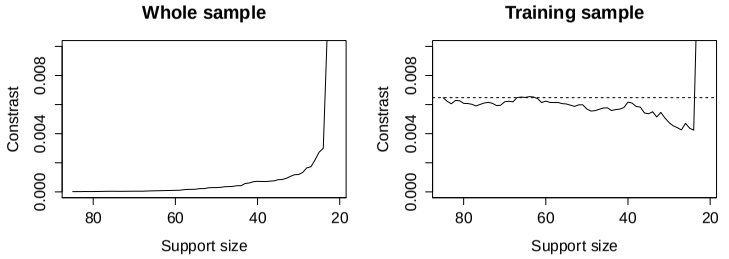


Figure 3: The contrast sequence $j \mapsto \mathrm{Crit}(\widetilde{\mathsf{W}}_{S_j}, \widehat{\mathsf{K}})$ computed in the example of Section 5.2. The nested support sequence and estimators $\widetilde{\mathsf{W}}_{S_j}$ are obtained from the backward algorithm implemented on the whole sample (left) and on a training sample of half size (right). In both cases, $\widehat{\mathsf{K}}$ is constructed from the whole sample. Using a training sample manages to reverse the monotonicity in the first part of the sequence, thus making the commutation gap easier to locate. The initial value of the sequence $t = \mathrm{Crit}(\widetilde{\mathsf{W}}_{S_1}, \widehat{\mathsf{K}})$ then provides a tractable adaptive choice for the threshold.

4.4 Improving the backward algorithm by bagging

The main weakness of the backward procedure remains that it requires the true support $S^\star$ to lie in the trajectory $S_1 \supset \cdots \supset S_\ell$ obtained from removing the least significant edge one at a time. In practice, this condition is rarely verified, especially with small datasets. A way to solve this issue is to replicate the backward algorithm over a collection of random sub-samples, a process commonly known to as Bootstrap Aggregating, or bagging. The description of this algorithm is given in Algorithm 2.

Algorithm 2: Bagging backward algorithm

Data: A set of forbidden entries F, a sample X.
Result: A collection of estimated supports $\widehat{S}_m, m = 1, ..., M$.

1: Build M bootstrapped samples without replacement.
2: For each sub-sample $m = 1, ..., M$, build an estimator $\widetilde{\mathsf{K}}_m$ of K.
3: For all m, run Algorithm 1 without stopping condition and return M trajectories $S_{1m} \supset \cdots \supset S_{\ell m}$ and the corresponding estimators $\widetilde{\mathsf{W}}_{S_{km}}$.
4: Evaluate the empirical contrast $\mathrm{Crit}(\widetilde{\mathsf{W}}_{S_{km}}, \widehat{\mathsf{K}})$ over each trajectory with the estimator $\widehat{\mathsf{K}}$ calculated from the whole sample.
5: For each trajectory, return the estimated support $\widehat{S}_m := S_{\hat{k}_m m}$ as the last support whose contrast lies below the initial value:

$$\hat{k}_m := \max\big\{k = 1, ..., \ell : \mathrm{Crit}(\widetilde{\mathsf{W}}_{S_{km}}, \widehat{\mathsf{K}}) \leq \mathrm{Crit}(\widetilde{\mathsf{W}}_{S_{1m}}, \widehat{\mathsf{K}})\big\}.$$

The bagging algorithm produces a collection of estimated supports in a way to make the final decision more robust. At this point, several solutions are possible: select the most represented support among the $\widehat{S}_m$'s, keep the edges present in the most supports etc... A preliminary detection of the outliers among the $\widehat{S}_m$'s, e.g. by removing beforehand the supports $\widehat{S}_m$'s that are either too big or too small, might also considerably improve the method, as we illustrate on actual examples in Section 5.

5. Numerical study

5.1 Toy example

In the previous section, we have introduced different algorithms. To emphasize the motivation of the bagging algorithm, we consider a simple example, and implement the different algorithms for support recovery. To check the performances of the ℓ_0 procedure, we need to consider a graph with a small number of vertices (since the ℓ_0 complexity grows with $2^{N(N-1)/2}$ where N denotes the number of vertices). Here, we consider the graph G_1 represented in Figure 4, the kite graph on 5 vertices.

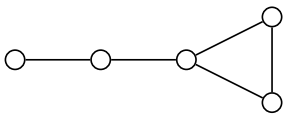


Figure 4: The kite graph $G_1 = \nabla_5$.

We choose W as the (normalized) adjacency matrix of G_1 then draw a sample of size $n = 500$ of centered Gaussian vectors $X_1, \cdots X_n$ of $\mathbb{R}^5$ with covariance matrix $\mathsf{K} = \exp(\mathsf{W})$. We assume

known that G_1 contains no self-loop so that we take $F = F_{diag}$ as the set of forbidden values. In this simple example, the constant $c_0(S^\star)$ (see Eq. (4)) can be calculated explicitly, yielding $c_0(S^\star) \approx 0.12$. In comparison, for $n = 500$, $\mathbb{E}\|\widehat{\mathsf{K}} - \mathsf{K}\|$ is evaluated to approximately 0.27 by Monte-Carlo. We expect to be able to recover the true support when the noise level drops below the signal amplitude. Based on the bound of Eq. (10), this occurs as soon as $\|\widehat{\mathsf{K}} - \mathsf{K}\| \leq c_0(S^\star)/2$. However, because this bound is not sharp, a lesser level of precision is required in practice.

We compare the following algorithms:

1. *Contrast penalized ℓ_0 minimization with optimal penalization constant.* We compute

$$\widehat{S} = \arg\min_{S \subseteq \overline{F}_{diag}} \left\{ \min_{\mathsf{A} \in \mathcal{E}(S)\setminus\{0\}} \frac{\|\mathsf{A}\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\mathsf{A}\|}{\|\mathsf{A}\|} + \lambda|S| \right\}.$$

   The constant λ is chosen as the best possible value, minimizing the oracle error $\delta(\widehat{S})$ measured by the symmetric difference between $\widehat{S}$ and $S^\star$, namely

$$\delta(\widehat{S}) := |\widehat{S} \cup S^\star \setminus \widehat{S} \cap S^\star|.$$

   Note that the calibration parameter λ is chosen optimally. Hence, the numerical performances of the method can be expected to be overestimated compared to a fully data-driven procedure.

2. *Thresholding contrast minimization with optimal threshold.* The target matrix W is estimated over the maximal acceptable support $\overline{F}_{diag}$. We then compute

$$\widehat{S} = \{(i, j) : |\widehat{\mathsf{W}}_{ij}| > t\},$$

   where the threshold t is chosen so as to minimize the oracle error $\delta(\widehat{S})$. Here again, the performances are expected to be overestimated compared to a data-driven threshold.

3. *Backward algorithm.* We generate a training sample by taking each observation with probability 1/2 independently, from the whole sample. The estimator of K in this sub-sample is denoted $\widetilde{\mathsf{K}}$. We implement Algorithm 1 on $\widetilde{\mathsf{K}}$, yielding a trajectory $S_1 \supset ... \supset S_\ell$ of nested supports whose sizes vary from $|S_1| = 20$ (the full off-diagonal support) to $|S_\ell| = 12$ (the minimal size required for diagonal identifiability), along with the associated estimators $\widetilde{\mathsf{W}}_{S_k}, k = 1, ..., \ell$. Remark that the supports are symmetric, hence two entries are removed at each step so that $\ell = 5$ in this case. We then compute the threshold $t = \mathrm{Crit}(\widetilde{\mathsf{W}}_{S_1}, \widehat{\mathsf{K}})$ corresponding to the initial value of the contrast. The estimated support $\widehat{S}$ is defined as the smallest support S in the trajectory such that $\mathrm{Crit}(\widehat{\mathsf{W}}_S, \widehat{\mathsf{K}}) \leq t$. Here, the choice of t is adaptive in a fully data driven manner.

4. *Bagging backward algorithm.* The previous algorithm is implemented over $M = 100$ training samples drawn keeping observations with probability 1/2. For each $m = 1, ..., M$, we retain

   - the threshold $t_m = \mathrm{Crit}(\widehat{\mathsf{W}}_{S_{1m}}, \widehat{\mathsf{K}})$ corresponding to the initial value of the contrast,
   - the estimated support, that is, the smallest support $\widehat{S}_m$ in the trajectory such that $\mathrm{Crit}(\widehat{\mathsf{W}}_{\widehat{S}_m}, \widehat{\mathsf{K}}) \leq t_m$.

   The estimated support $\widehat{S}$ is obtained as follows. Only a proportion q of the training samples m with a small initial contrast t_m are kept, they are expected to provide better results—in the whole paper, we chose $q = 2/\sqrt{M}$ empirically. Then, the smallest support among the remaining candidates is retained, breaking ties arbitrarily.

Remark 12 *Using a ℓ_1-penalized contrast*

$$\mathsf{A} \mapsto \|\widehat{\mathsf{K}}\mathsf{A} - \mathsf{A}\widehat{\mathsf{K}}\|^2 + \lambda\|\mathsf{A}\|_1$$

(subject to a normalizing condition so as to rule out the trivial solution $\mathsf{A} = 0$) tends to overestimate the support. In fact, any conservative choice of λ will lead to false positives in the estimated support (typically, a full support matrix may commute with $\widehat{\mathsf{K}}$ while still having a small ℓ_1 norm). Hence, when aiming for support recovery, the typical solution is to vanish the small entries of the minimizer, making it no more efficient than the thresholded ℓ_2 procedure considered in Algorithm 2. For this reason, the numerical performances of the Lasso procedure are not included in the study.

The next table compares the performances of the four algorithms. We calculated the Monte-Carlo estimated mean error $\mathbb{E}(\delta(\widehat{S}))$ and probability of exact recovery $\mathbb{P}\{\widehat{S} = S^\star\}$ for 1000 repetitions of the experiment. The average computational time (obtained with the function `timer` of Scilab) on a processor Intel Xeon @2.6 GHz are shown, using the oracle values of λ and t for the first two algorithms (the calibration of these parameters is thus not accounted for in the computation time).

Algorithm	ℓ_0	ℓ_2-thresholding	Backward	Bagging Backward
Mean Error	0.45	0.37	1.95	0.68
Exact recoverery	68%	75%	23%	61%
CPU time (s)	0.32	0.002	0.009	0.59

In this example, the first two algorithms are the more accurate. The percentage of successful recoveries for the bagging backward algorithm is nonetheless competitive given that the first two procedures have been calibrated optimally for each experiment, which would be highly infeasible in practice. Finally, we observe that although it is much more expensive computationally, the bagging version of the backward algorithm yields an undeniable improvement.

Upper bounds for the time and space complexity of the algorithms are given in the next table. The time complexity is calculated as the number of different supports S considered to lead to the solution in function of the size N of the graph and the number M of training samples. The spatial complexity measures the memory size needed to compute the solution. In this setting, it is the main limitation for applying the procedures to large graphs. The N^4 comes from the computation of $\Delta(\mathsf{K}) = \mathsf{K} \otimes \mathrm{I} - \mathrm{I} \otimes \mathsf{K}$ in the solver. Admittedly, the complexity could be improved by using sparse matrix encoding although this was not implemented.

Algorithm	ℓ_0	ℓ_2-thresholding	Backward	Bagging Backward
Space Complexity	$O(N^4)$	$O(N^4)$	$O(N^4)$	$O(N^4)$
Time Complexity	$O(2^{N(N-1)/2})$	$O(1)$	$O(N^2)$	$O(N^2.M)$

On the current version, the bagging backward algorithm contains scalability issues for big graphs due to its space complexity. Leads to reduce the spatial complexity include using sparse matrix encoding or the use of cheap approximations of the criterion. These shall be investigated in future works.

5.2 A diagonally identifiable matrix

The advantages of the bagging backward algorithm are highlighted for larger graphs. In the next example, we consider the graph G_2 on $N = 15$ vertices represented in Figure 5. The experimental conditions are similar to that of the previous example, a sample of size $n = 10000$ is drawn from

a centered Gaussian vector of variance $\mathsf{K} = \exp(\mathsf{W})$ where W is the normalized adjacency matrix of G_2, with normalizing constant chosen such that $\mathbf{1}^\top w = 1$. The implementation of the different algorithms follow the description of the previous example.

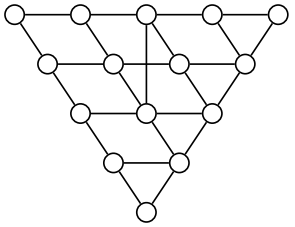


Figure 5: The graph G_2 is diagonally identifiable.

In this case, the number of possible supports is too large for the ℓ_0 method to be implementable while the accuracy of the thresholded ℓ_2 drops considerably compared to smaller cases. We summarize the results in the following table.

Algorithm	ℓ_2-thresholding	Backward	Bagging Backward
Mean Error	10	25	1
Exact recoverery	22%	26%	69%
CPU time (s)	0.04	2.5	256

A drawback of the bagging backward algorithm is the larger computational time: it takes around 4 minutes in average to estimate the support. Being essentially $M = 100$ repetitions of the backward algorithm, the numerical complexity of the bagging version is roughly M times that of the simple backward algorithm, although the improvement is, here again, clear.

To illustrate the influence of the unknown function f, we consider $f : t \mapsto (1-t)^{-2}$ and reproduce the numerical study for $\mathsf{K} = f(\mathsf{W})$. The results for various sample sizes are gathered in the next table, for $M = 100$ bagging runs.

n	10000	5000	2000	1000
Exact recovery	97%	87%	83%	13%
Mean error	0.05	0.33	0.9	8.5

The probability of recovering the true support appears to be greater than in the previous example (97% against 69% previously for $n = 10000$). This sheds lights on another important factor in the efficiency of the methods which is the separability of the spectrum of K. Indeed, in this framework, the information needed to recover W lies in its eigenspaces, which are estimated via $\widehat{\mathsf{K}}$. The accuracy of these estimates depends on the distance between the different eigenvalues (see e.g. Corollary 4.12 in Stewart and Sun (1990) and Wedin's $\sin(\theta)$ theorem in Stewart and Sun (1990)). Thus, for the spectrum $\lambda_1, ..., \lambda_N$ of W, the ability to recover W from $\mathsf{K} = f(\mathsf{W})$ essentially relies on how far the $f(\lambda_i)$'s are from each other. For the sake of comparison, the spectrum of W which lies in the interval $[-0.5, 0.5]$ is more "spread" by the function $t \mapsto (1-t)^{-2}$ than by the exponential, as we can see in Figure 6.

Remark 13 *We also implemented the procedure in a random setting where W is drawn from an Erdös-Rényi graph with binomial entries. The conclusions obtained in this case are similar to those already discussed and shall not be presented to avoid redundancy.*

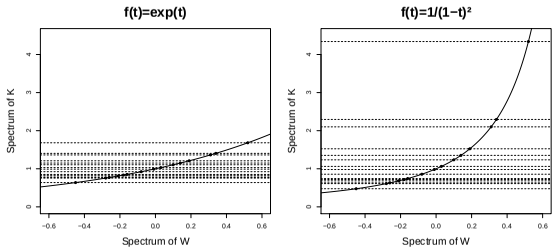


Figure 6: Separability of the spectrum of $\mathsf{K} = f(\mathsf{W})$ for $f : t \mapsto \exp(t)$ (left) and $f : t \mapsto (1-t)^{-2}$ (right). The eigenvalues of K are more separated in the second case, making it easier to approximate its eigenspaces from the estimator $\widehat{\mathsf{K}}$.

6. Real life application

We now implement the bagging backward algorithm on real life data provided by Météorage and Météo France. The data contain the daily number of lightnings during a 3 year period in 16 regions of France localized on a 4×4 grid. We expect to recover the spatial structure of the graph from the dependence of the lightning occurrences between the regions.

The data are refined as follows. We first eliminate the days without any lighting, leaving 950 vectors X_i of length 16 that contain the number of impacts at day i in each of the 16 regions. This numbers are highly non Gaussian, contain many zeros, and show a clear south-east/north-west tendency, with much more lightning in the south east. As a pre-processing, we apply the transformation $x \mapsto \log(1+x)$ to the data and subtract the spatial tendency estimated by linear regression. The process is then normalized in such manner that the conditional variance at each vertex conditionally to all the others is 1. This way, the precision matrix K^{-1}, with K the covariance matrix, has diagonal 1.

We model the resulting process as a spatial AutoRegressive process of order $p > 1$ on a 4×4 grid, as described in Section 3.6. In this setting, K^{-1} writes as a degree p polynomial of a matrix W supported on the 4×4 grid. Using only the information that W has zero diagonal, we aim to recover this dependency from the empirical estimator $\widehat{\mathsf{K}}$ of the covariance, using that K and W commute. Remark that, because the target graph is bipartite, the model is not diagonally identifiable—for instance W^3 also has zero diagonal. However, we still manage to recover the support from the fact that W is the *sparsest* matrix that commute with K. We run our algorithm 100 times and we show in Figure 7 the most frequent edges appearing in the output graph.

To the best of our knowledge, this exact framework has not been tackled in the literature. For this reason, it is difficult to compare the performances of our algorithm to other existing methods. For instance, while being a reference in Graphical Models, the package `GGMselect` (see Giraud et al. (2012)) fails to uncover the dependence structure in this case, as we see in Figure 8. The reason is simple: the package `GGMselect` aims to estimate the precision matrix, which is a polynomial of W. Thus, while GGM inference may be more stable, faster ($0.3s$ for `GGMselect` and $400s$ for our algorithm) and easily interpretable, it is only adapted to recover the dependence structure if $p = 1$, in which case K^{-1} is an affine function of W.

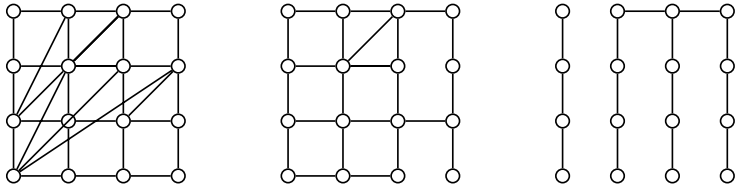


Figure 7: Edges that appear 30%, 50%, and 70% of the time when running the bagging backward algorithm 100 times. The spatial dependency becomes apparent around the 50% mark.

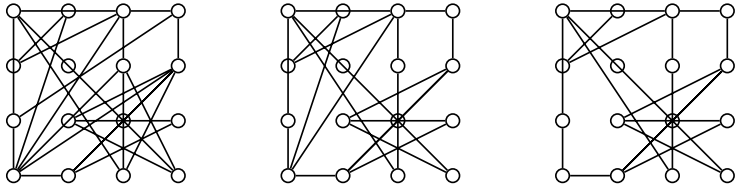


Figure 8: Edges that appear 30%, 50%, and 70% of the time using the `GGMselect` package with family `CO1` and maximal degree `dmax` $= 5$.

7. Discussion

In this paper, we develop a new method to recover hidden graphical structures in different models. We consider a general framework in which we have access to an approximation of the eigen-structure of an unknown graph, via an operator that commutes with its weighted adjacency matrix W. We are able to recover the support of W from an estimate of $\mathsf{K} = f(\mathsf{W})$ with f an unknown function, under the sole assumption that the location of some zeros of W are known. We tackle two situations where this condition may arise: Markov processes observed at random times and stationary signals on graphs. We focus on the particular case of a weighted adjacency matrix W with zero diagonal, indicating that the underlying graph has no self-loop.

A main limitation of our method lies in the large amount of data necessary to recover the true support with high probability. Arguably, this limitation is intrinsic to our model and is a matter of comparison between the estimation error $\|\widehat{\mathsf{K}} - \mathsf{K}\|$ and the signal strength, measured by the constant $c_0(S^\star)$ in (4) and (10). Even under the assumption that the estimation error has order $1/\sqrt{n}$, the difficulties stem from the constant $c_0(S^\star)$ being extremely small in some cases.

For practical issues, there remain three main challenges to be addressed. The first one concerns the symmetry of W, that once relaxed, would offer a wider range of applications. The second concerns the generalization and applications to identifiability conditions other than W having zero diagonal. Finally, our algorithm is greedy when the size of the graph increases. For large graphs, it remains to find a way to compute more efficiently the criterion and significance of the variables.

Acknowledgement

We would like to warmly thank Météo France and Météorage for providing us the data used in Section 6. We also thank Dieter Mitsche for fruitful discussions, as well as the Universidad de la Habana (Cuba) and the Centro de Modelamiento Matematico (Chile) for their hospitality.

Appendix A. Asserting the Diagonal Identifiability

A.1 Necessary and sufficient conditions

In this section, we focus on the F-identifiability in the special case where the set of forbidden entries is the diagonal $F_{\rm diag} := \{(i,i) : i \in [1,N]\}$. Recall that a support S is $F_{\rm diag}$-identifiable, or simply diagonally identifiable (DI), if for almost every matrix $\mathsf{A} \in \mathcal{E}(S)$,

$$\mathsf{BA} = \mathsf{AB}\ ,\ \mathrm{diag}(\mathsf{B}) = 0\ ,\ \mathsf{B} = \mathsf{B}^\top \implies \mathsf{B} = \lambda \mathsf{A}$$

for some $\lambda \in \mathbb{R}$. In other words, a support S is diagonally identifiable if almost every symmetric matrix A with support in S is uniquely determined, up to scaling, by its eigenspaces among symmetric matrices with zero diagonal. In this section, we provide both sufficient and necessary conditions on a support S to ensure the $F_{\rm diag}$-identifiability. For this, we consider a simple undirected graph $G_S = ([1,N], S)$ on N vertices with edge set S.

Definition 14 (Induced subgraph) *For $V \subseteq [1,N]$, the induced subgraph $G_S(V) = (V, S(V))$ is the graph on V with edge set $S(V) = S \cap V^2$.*

Proposition 15 *For all support $S \subseteq [1,N]^2$, the set of invertible matrices in $\mathcal{E}(S)$ is either empty or a dense open subset of $\mathcal{E}(S)$.*

The proof is straightforward when writing the determinant of $\mathsf{A} \in \mathcal{E}(S)$ as a polynomial in its entries. Observe that by this property, finding one invertible matrix A in $\mathcal{E}(S)$ guarantees that almost every matrix in $\mathcal{E}(S)$ is invertible. In this case, we say that the graph G_S is invertible. Similarly, we say that G_S is diagonally identifiable if S is diagonally identifiable.

Theorem 16 (Conditions for $F_{\rm diag}$-identifiability) *Let $S \subseteq \overline{F}_{\rm diag}$ and $G_S = ([1,N], S)$.*

1. ***Necessary condition:*** *If S is diagonally identifiable then there exists a sequence of subsets $V_3, ..., V_{N-1} \subset [1,N]$ such that $|V_k| = k$ and $G_S(V_k)$ is invertible for all $k = 3, ..., N-1$.*
2. ***Sufficient condition:*** *If there exists a nested sequence $V_3 \subset ... \subset V_{N-1} \subset [1,N]$ with $|V_k| = k$ such that $G_S(V_k)$ is invertible for all $k = 3, ..., N-1$, then S is diagonally identifiable.*

The gap between the sufficient and necessary conditions lies in the fact that the sequence $V_3, ..., V_{N-1}$ need to be nested for the sufficient condition.

Proof We proceed by contradiction. For the necessary condition, let $k \geq 3$ be such that $G_S(V_k)$ is not invertible, for all $V_k \subset [1,N]$ of size k. For $\mathsf{A} \in \mathcal{E}(S)$, denote by $\psi_0(\mathsf{A}), \psi_1(\mathsf{A}), \ldots, \psi_N(\mathsf{A})$ the coefficients of the characteristic polynomial

$$\det(z\,\mathrm{I} - \mathsf{A}) = \sum_{j=0}^{N} \psi_j(\mathsf{A})\ z^j, \quad z \in \mathbb{R}.$$

Consider the matrix $M_k(\mathsf{A}) := \sum_{j=0}^{k} \psi_j(\mathsf{A})\ \mathsf{A}^j$. By Eq. (14) in Espinasse and Rochet (2016), we see that the (i,i)-entry of $M_k(\mathsf{A})$ equals the sum of all minors of size k that do not contain the vertex i. Since for all subset V_k of size k, $G_S(V_k)$ is not invertible, this implies that $M_k(\mathsf{A})$ has zero diagonal. On the other hand, the non-zero entries of $M_k(\mathsf{A})$ are degree k polynomials in the variables $\mathsf{A}_{ij}, (i,j) \in \mathrm{Supp}(\mathsf{A})$. Therefore, the equality $M_k(\mathsf{A}) = \lambda \mathsf{A}$ for some $\lambda \in \mathbb{R}$ occurs for at most a countable number of $\mathsf{A} \in \mathcal{E}(S)$. Since $M_k(\mathsf{A})$ commutes with A, we deduce that S is not diagonally identifiable.

For the sufficient condition, we will need the following lemma.

Lemma 17 *If there exists a subset $V' \subset [1,N]$ of size $N-1$ such that $G_S(V')$ is both DI and invertible, then G_S is DI.*

Proof We may assume that $V' = [1, N-1]$ without loss of generality. Let M' denote a symmetric $(N-1)\times(N-1)$ matrix indexed on V' that is both invertible and diagonally identifiable, *i.e.*, for all non-zero matrix $\mathsf{A}' \neq \lambda \mathsf{M}'$,

$$\mathsf{M}'\mathsf{A}' = \mathsf{A}'\mathsf{M}' \implies \mathrm{diag}(\mathsf{A}') \neq 0.$$

To prove that G_S is DI, it suffices to find a symmetric matrix M with support S that is diagonally identifiable. Consider M defined by

$$\mathsf{M} = \begin{bmatrix} \mathsf{M}' & 0 \\ 0 & 0 \end{bmatrix}.$$

Let A be a matrix with zero diagonal that commutes with M and write

$$\mathsf{A} = \begin{bmatrix} \mathsf{A}' & a \\ a^\top & 0 \end{bmatrix}$$

for some $a \in \mathbb{R}^{N-1}$, with $\mathrm{diag}(\mathsf{A}') = 0$. The condition $\mathsf{MA} = \mathsf{AM}$ can be stated equivalently as

$$\begin{cases} \mathsf{M}'\mathsf{A}' = \mathsf{A}'\mathsf{M}' \\ \mathsf{M}'a = 0 \end{cases}$$

Since M' is invertible by assumption, $a = 0$ and the only matrix A with zero diagonal that commutes with M is the null matrix. Thus, M is diagonally identifiable. ■

We now go back to prove the sufficient condition in Theorem 16. Assume that G_S is not diagonally identifiable, then by Lemma 17, neither is $G_S(V_{N-1})$. By iterating the argument, we conclude that $G_S(V_3)$ is not diagonally identifiable. However, the only invertible graph on three vertices is the triangle graph, which is diagonally identifiable, leading to a contradiction. ■

Remark 18 *The proof of Theorem 16 combines the results of Lemma 2.1 in Barsotti et al. (2014) and Eq. (14) in Espinasse and Rochet (2016). The first one is of topological flavor proving that the set of identifiable matrices is either dense or empty in the set of matrices with prescribed support. The second ingredient is Eq. (14) in Espinasse and Rochet (2016) which contains a key combinatorial computation on the adjugate matrix of weighted graphs. Admittedly, its purpose was to provide the first step to prove the necessary condition for identifiability in the present article (precisely, that $M_k(\mathsf{A})$ has zero diagonal). The proof of the sufficient condition does not, however, involve this result.*

A.2 Proof of Proposition 4

From Claim (*ii*) in Theorem 16 and considering the nested sequence $V_{N-1} \supset ... \supset V_3$ obtained by removing the last vertex on the tail of the kite at each step, we deduce a simple and tractable sufficient condition for a graph G_S to be diagonally identifiable, namely that G_S contains the kite graph as a vertex covering (possibly not induced) subgraph.

A.3 Existence of kites

The condition on containing the kite graph ∇_N as a subgraph is mild in the sense that it is satisfied in the dense regime $\log n/n$ by random graphs, as depicted in the following proposition.

Proposition 19 *The existence of kite graphs in the Erdős-Rényi model occurs as follows. For any $\omega(N) \to \infty$ and for $G_N \sim G(N, p_N)$, if $p_N \geq (1/N)(\log N + \log\log N + \omega(N))$ then $\mathbb{P}\{G_N$ has a kite of length $N\}$ tends to 1 as N goes to infinity.*

The proof makes use of the existence of a hamiltonian cycle which is a standard result in Random Graph Theory, see Corollary 8.12 in Bollobás (1998) for instance. This results shows that in the regime $(\log N + \log\log N)/N$ an Erdős-Rényi graph is diagonally identifiable.

Proof We now present the proof of this fact. Let $\omega(n) \to \infty$ and set

$$p_1 := (1/n)(\log n + \log\log n + \omega(n)/2),$$
$$p_2 := \omega(n)/(2n)\,.$$

Let $G^{(1)}$ and $G^{(2)}$ be two independent Erdős-Rényi graphs such that

$$G_n^{(1)} \sim G(n, p_1) \quad \perp\!\!\!\perp \quad G_n^{(2)} \sim G(n, p_2)\,.$$

As shown in Corollary 8.12 in Bollobás (1998) for instance, $\mathbb{P}\{G_n^{(1)}$ is hamiltonian$\}$ tends to 1 as n goes to infinity. Given a hamiltonian cycle C_n of length n in $G^{(1)}$ one can construct a kite of length n using edges of $G^{(2)}$ to connect a pair of vertices at distance 2 on the cycle C_n. Invoke the independence of $G^{(1)}$ and $G^{(2)}$ to get that this latter probability is

$$\mathbb{P}\{\{k, k+2\} \text{ is an edge of } G^{(2)} \text{ for some } k\} = \mathbb{P}\{B(n, p_2) > 0\}\,,$$

where $B(n, p_2)$ denotes the binomial law. Using Poisson approximation one gets that this probability tends to 1 as n goes to infinity. We deduce that the probability that the graph $G = G_n^{(1)} + G_n^{(2)}$ has at least a kite tends to 1. Observe that G is an Erdős-Rényi graph of size n and parameter $p = p_1 + p_2 - p_1 p_2 \leq p_n$ which concludes the proof. ∎

A.4 Proof of Theorem 5

Combining Proposition 19 and Theorem 16, we deduce the first point. From the necessary condition in Theorem 16, we see that it is sufficient to find two isolated vertices to prove non-identifiability. Indeed, in this case, the kernel of the adjacency matrix has co-dimension at least 2 showing that all sub-graphs of size $N-1$ are not invertible. Furthermore, one knows (see Theorem 3.1 in Bollobás (1998) for instance) that the event "there is at least two isolated points" has sharp threshold function $\log n/n$. This proves the second point.

Appendix B. Support reconstruction

B.1 Proof of Theorem 7

Define $\mathcal{S}_1 := \{S \in \mathcal{S} : |S| \leq |S^\star|, S \neq S^\star\}$ and $\mathcal{S}_2 := \{S \in \mathcal{S} : |S| > |S^\star|\}$, clearly it holds $\mathcal{S} = \{S^\star\} \cup \mathcal{S}_1 \cup \mathcal{S}_2$. We want to control the terms $\mathbb{P}\{\widehat{S} \in \mathcal{S}_1\}$ and $\mathbb{P}\{\widehat{S} \in \mathcal{S}_2\}$ separately and conclude in view of

$$\mathbb{P}\{\widehat{S} \neq S^\star\} = \mathbb{P}\{\widehat{S} \in \mathcal{S}_1\} + \mathbb{P}\{\widehat{S} \in \mathcal{S}_2\}\,.$$

Since the Frobenius norm is sub-multiplicative, it holds, for all $\mathsf{A} \in \mathcal{E}(\overline{F})$,

$$\|\mathsf{A}(\widehat{\mathsf{K}} - \mathsf{K}) - (\widehat{\mathsf{K}} - \mathsf{K})\mathsf{A}\| \leq \|\mathsf{A}(\widehat{\mathsf{K}} - \mathsf{K})\|_2 + \|(\widehat{\mathsf{K}} - \mathsf{K})\mathsf{A}\| \leq 2\|\mathsf{A}\|\|\widehat{\mathsf{K}} - \mathsf{K}\|\,.$$

Thus, the quantity $\|\mathsf{A}\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\mathsf{A}\|$ for $\mathsf{A} \in \mathcal{E}(\overline{F})$ can be bounded from below and above by

$$\|\mathsf{AK} - \mathsf{KA}\| - 2\|\mathsf{A}\|\|\widehat{\mathsf{K}} - \mathsf{K}\| \leq \|\mathsf{A}\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\mathsf{A}\| \leq \|\mathsf{AK} - \mathsf{KA}\| + 2\|\mathsf{A}\|\|\widehat{\mathsf{K}} - \mathsf{K}\|. \tag{11}$$

To bound the term $\mathbb{P}\{\widehat{S} \in \mathcal{S}_1\}$, we use (11) to remark that for all $S \in \mathcal{S}_1$,

$$Q(S) = \min_{\mathsf{A} \in \mathcal{E}(S)\setminus\{0\}} \frac{\|\mathsf{A}\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\mathsf{A}\|}{\|\mathsf{A}\|} + \lambda_n|S| \geq \min_{\mathsf{A} \in \mathcal{E}(S)\setminus\{0\}} \frac{\|\mathsf{AK} - \mathsf{KA}\|}{\|\mathsf{A}\|} - 2\|\widehat{\mathsf{K}} - \mathsf{K}\|\,.$$

It follows

$$\min_{S \in \mathcal{S}_1} Q(S) \geq \min_{S \in \mathcal{S}_1} \min_{\mathsf{A} \in \mathcal{E}(S)\setminus\{0\}} \frac{\|\mathsf{AK} - \mathsf{KA}\|}{\|\mathsf{A}\|} - 2\|\widehat{\mathsf{K}} - \mathsf{K}\| = c_0(S^\star) - 2\|\widehat{\mathsf{K}} - \mathsf{K}\|. \tag{12}$$

The constant $c_0(S^\star)$ is positive by F-identifiability of W. Moreover, observe that

$$Q(S^\star) = \min_{\mathsf{A} \in \mathcal{E}(S^\star)\setminus\{0\}} \frac{\|\mathsf{A}\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\mathsf{A}\|}{\|\mathsf{A}\|} + \lambda_n|S^\star| \leq \frac{\|\mathsf{W}\widehat{\mathsf{K}} - \widehat{\mathsf{K}}\mathsf{W}\|}{\|\mathsf{W}\|} + \lambda_n|S^\star| \leq 2\|\widehat{\mathsf{K}} - \mathsf{K}\| + \lambda_n|S^\star|, \tag{13}$$

where we used both Eq. (11) and the fact that $\mathsf{WK} - \mathsf{KW} = 0$. Combining (12) and (13), we get

$$\mathbb{P}\{\widehat{S} \in \mathcal{S}_1\} \leq \mathbb{P}\Big\{\min_{S \in \mathcal{S}_1} Q(S) \leq Q(S^\star)\Big\} \leq \mathbb{P}\Big\{\|\widehat{\mathsf{K}} - \mathsf{K}\| \geq \frac{c_0(S^\star) - \lambda_n|S^\star|}{4}\Big\}\,.$$

To control the term $\mathbb{P}(\widehat{S} \in \mathcal{S}_2)$, we use that $\min_{S \in \mathcal{S}_2} Q(S) \geq \lambda_n \min_{S \in \mathcal{S}_2} |S| \geq \lambda_n(|S^\star| + 1)$. By Eq. (13), it follows

$$\begin{aligned}\mathbb{P}\big\{\widehat{S} \in \mathcal{S}_2\big\} &\leq \mathbb{P}\Big\{\min_{S \in \mathcal{S}_2} Q(S) \leq Q(S^\star)\Big\} \\ &\leq \mathbb{P}\Big\{\lambda_n(|S^\star| + 1) \leq 2\|\widehat{\mathsf{K}} - \mathsf{K}\| + \lambda_n|S^\star|\Big\} \\ &= \mathbb{P}\Big\{\|\widehat{\mathsf{K}} - \mathsf{K}\| \geq \frac{\lambda_n}{2}\Big\}\,.\end{aligned}$$

The proof of Theorem 7 follows directly by ($\mathbf{H}_2$). The corollary is a direct consequence using Borel-Cantelli's Lemma.

B.2 Proof of Theorem 10

Since $\Delta(\mathsf{K})\Phi_S$ is of full rank, the value $\widehat{\beta}_S = \big(\Delta(\widehat{\mathsf{K}})\Phi_S\big)^\dagger \Delta(\widehat{\mathsf{K}})a_0$ is the unique solution to Eq. (5) with probability tending to one asymptotically. Since the value of $\widehat{\beta}_S$ does not depend on $a_0 \in \mathcal{A}_S$, one can take $a_0 = w$ in view of $S^\star \subseteq S$. We obtain

$$\widehat{\beta}_S = \big(\Delta(\widehat{\mathsf{K}})\Phi_S\big)^\dagger \Delta(\widehat{\mathsf{K}})w = -\big(\Delta(\widehat{\mathsf{K}})\Phi_S\big)^\dagger \Delta(\mathsf{W})\widehat{k}.$$

The result follows from Slutsky's lemma, using that $(\Delta(\widehat{\mathsf{K}})\Phi_S)^\dagger$ converges in probability towards $(\Delta(K)\Phi_S)^\dagger$ and

$$\sqrt{n}\,\big(\Delta(\mathsf{W})\widehat{k} - \Delta(\mathsf{W})k\big) \xrightarrow[n\to\infty]{d} \mathcal{N}\big(0, \Delta(\mathsf{W})\Sigma\Delta(\mathsf{W})^\top\big).$$

References

David F Anderson and Thomas G Kurtz. Continuous time markov chain models for chemical reaction networks. In *Design and Analysis of Biomolecular Circuits*, pages 3–42. Springer, 2011.

Flavia Barsotti, Yohann De Castro, Thibault Espinasse, and Paul Rochet. Estimating the transition matrix of a Markov chain observed at random times. *Statistics & Probability Letters*, 94:98–105, 2014.

Flavia Barsotti, Anne Philippe, and Paul Rochet. Hypothesis testing for markovian models with random time observations. *Journal of Statistical Planning and Inference*, 173:87–98, 2016.

José Bento and Morteza Ibrahimi. Support recovery for the drift coefficient of high-dimensional diffusions. *IEEE Transactions on Information Theory*, 60(7):4026–4049, 2014.

José Bento, Morteza Ibrahimi, and Andrea Montanari. Learning networks of stochastic differential equations. In *Advances in Neural Information Processing Systems*, pages 172–180, 2010.

Béla Bollobás. *Random graphs*. Springer, 1998.

Sébastien Bubeck, Jian Ding, Ronen Eldan, and Miklós Z Rácz. Testing for high-dimensional geometry in random graphs. *Random Structures & Algorithms*, 2016.

Thibault Espinasse and Paul Rochet. Relations between connected and self-avoiding hikes in labelled complete digraphs. *Graphs and Combinatorics*, to appear, 2016.

Thibault Espinasse, Fabrice Gamboa, and Jean-Michel Loubes. Parametric estimation for gaussian fields indexed by graphs. *Probability Theory and Related Fields*, 159(1-2):117–155, 2014.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, 9(3):432–441, 2008.

Donald P Gaver Jr. Imbedded markov chain analysis of a waiting-line process in continuous time. *The Annals of Mathematical Statistics*, pages 698–720, 1959.

Christophe Giraud, Sylvie Huet, and Nicolas Verzelen. Graph selection with ggmselect. *Statistical applications in genetics and molecular biology*, 11(3), 2012.

Benjamin Girault. Stationary graph signals using an isometric graph translation. In *Signal Processing Conference (EUSIPCO), 2015 23rd European*, pages 1516–1520. IEEE, 2015.

Olivier Guédon and Roman Vershynin. Community detection in sparse networks via grothendieck's inequality. *Probability Theory and Related Fields*, pages 1–25, 2015.

Aapo Hyvärinen, Kun Zhang, Shohei Shimizu, and Patrik O Hoyer. Estimation of a structural vector autoregression model using non-gaussianity. *The Journal of Machine Learning Research*, 11:1709–1731, 2010.

Jing Jiang, Christo Wilson, Xiao Wang, Wenpeng Sha, Peng Huang, Yafei Dai, and Ben Y Zhao. Understanding latent interactions in online social networks. *ACM Transactions on the Web (TWEB)*, 7(4):18, 2013.

Olga Klopp, Alexandre B Tsybakov, and Nicolas Verzelen. Oracle inequalities for network models and sparse graphon estimation. *The Annals of Statistics*, 45(1):316–354, 2017.

Elizabeth Chase MacRae. Estimation of time-varying Markov processes with aggregate data. *Econometrica*, 45(1):183–198, 1977. ISSN 0012-9682.

Antonio G Marques, Santiago Segarra, Geert Leus, and Alejandro Ribeiro. Stationary graph processes and spectral estimation. *arXiv preprint arXiv:1603.04667*, 2016.

Catherine Matias, Tabea Rebafka, and Fanny Villers. A semiparametric extension of the stochastic block model for longitudinal networks. *arXiv preprint arXiv:1512.07075*, 2015.

Vincent Miele and Catherine Matias. Revealing the hidden structure of dynamic ecological networks. *arXiv preprint arXiv:1701.01355*, 2017.

Nathanaël Perraudin and Pierre Vandergheynst. Stationary signal processing on graphs. *arXiv preprint arXiv:1601.02522*, 2016.

Arthur O. Pittenger. Time changes of Markov chains. *Stochastic Process. Appl.*, 13(2):189–199, 1982. ISSN 0304-4149. doi: 10.1016/0304-4149(82)90034-5. URL http://dx.doi.org/10.1016/0304-4149(82)90034-5.

Fabrice Rossi and Pierre Latouche. Activity date estimation in timestamped interaction networks. In *21-th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, pages 113–118, Apr. 2013.

Gilbert W. Stewart and Ji-guang Sun. *Matrix Perturbation Theory*. Computer Science and Scientific Computing. Academic Press Boston, 1990.

Mathieu Thomas, Nicolas Verzelen, Pierre Barbillon, Oliver T Coomes, Sophie Caillon, Doyle McKey, Marianne Elias, Eric Garine, Christine Raimond, Edmond Dounias, et al. A network-based method to detect patterns of local crop biodiversity: Validation at the species and infra-species levels. *Advances in Ecological Research*, 53:259–320, 2015.

Nicolas Verzelen. *Gaussian graphical models and Model selection*. PhD thesis, Université Paris Sud-Paris XI, 2008.

Nicolas Verzelen, Ery Arias-Castro, et al. Community detection in sparse random networks. *The Annals of Applied Probability*, 25(6):3465–3510, 2015.

Journal of Machine Learning Research 18 (2017) 1-11 Submitted 12/16; Published 5/17

An Optimal Algorithm for Bandit and Zero-Order Convex Optimization with Two-Point Feedback

Ohad Shamir OHAD.SHAMIR@WEIZMANN.AC.IL
Department of Computer Science and Applied Mathematics
Weizmann Institute of Science
Rehovot 7610001, Israel

Editor: Alexander Rakhlin

Abstract

We consider the closely related problems of bandit convex optimization with two-point feedback, and zero-order stochastic convex optimization with two function evaluations per round. We provide a simple algorithm and analysis which is optimal for convex Lipschitz functions. This improves on Duchi et al. (2015), which only provides an optimal result for smooth functions; Moreover, the algorithm and analysis are simpler, and readily extend to non-Euclidean problems. The algorithm is based on a small but surprisingly powerful modification of the gradient estimator.

Keywords: zero-order optimization, bandit optimization, stochastic optimization, gradient estimator

1. Introduction

We consider the problem of bandit convex optimization with two-point feedback Agarwal et al. (2010). This problem can be defined as a repeated game between a learner and an adversary as follows: At each round t, the adversary picks a convex function f_t on $\mathbb{R}^d$, which is not revealed to the learner. The learner then chooses a point $\mathbf{w}_t$ from some known and closed convex set $\mathcal{W} \subseteq \mathbb{R}^d$, and suffers a loss $f_t(\mathbf{w}_t)$. As feedback, the learner may choose two points $\mathbf{w}'_t, \mathbf{w}''_t \in \mathcal{W}$ and receive[1] $f_t(\mathbf{w}'_t), f_t(\mathbf{w}''_t)$. The learner's goal is to minimize average regret, defined as

$$\frac{1}{T}\sum_{t=1}^{T} f_t(\mathbf{w}_t) - \min_{\mathbf{w}\in\mathcal{W}} \frac{1}{T}\sum_{t=1}^{T} f_t(\mathbf{w}).$$

In this paper, we focus on obtaining bounds on the expected average regret (with respect to the learner's randomness).

A closely-related and easier setting is zero-order stochastic convex optimization. In this setting, our goal is to approximately solve $F(\mathbf{w}) = \min_{\mathbf{w}\in\mathcal{W}} \mathbb{E}_\xi[f(\mathbf{w};\xi)]$, given limited access to $\{f(\cdot;\xi_t)\}_{t=1}^T$ where ξ_t are i.i.d. instantiations. Specifically, we assume that each $f(\cdot,\xi_t)$ is not directly observed, but rather can be queried at two points. This models situations where computing gradients directly is complicated or infeasible. It is well-known (Cesa-Bianchi et al., 2004) that given an algorithm with expected average regret R_T in the bandit optimization setting above, if we feed it with the functions $f_t(\mathbf{w}) = f(\mathbf{w};\xi_t)$, then the average $\bar{\mathbf{w}}_T = \frac{1}{T}\sum_{t=1}^T \mathbf{w}_t$ of the points generated satisfies the following bound on the expected optimization error:

$$\mathbb{E}[F(\bar{\mathbf{w}}_T)] - \min_{\mathbf{w}\in\mathcal{W}} F(\mathbf{w}) \leq R_T.$$

Thus, an algorithm for bandit optimization can be converted to an algorithm for zero-order stochastic optimization with similar guarantees.

The bandit optimization setting with two-point feedback was proposed and studied in Agarwal et al. (2010). Independently, Nesterov (2011) and considered two-point methods for stochastic optimization. Both papers are based on randomized gradient estimates which are then fed into standard first-order algorithms (e.g. gradient descent, or more generally mirror descent). However, the regret/error guarantees in both papers were suboptimal in terms of the dependence on the dimension. Recently, Duchi et al. (2015) considered a similar approach for the stochastic optimization setting, attaining an optimal error guarantee when $f(\cdot;\xi)$ is a smooth function (differentiable and with Lipschitz-continuous gradients). Related results in the smooth case were also obtained by Ghadimi and Lan (2013). However, to tackle the general case, where $f(\cdot;\xi)$ may be non-smooth, Duchi et al. (2015) resorted to a non-trivial smoothing scheme and a significantly more involved analysis. The resulting bounds have additional factors (logarithmic in the dimension) compared to the guarantees in the smooth case. Moreover, an analysis is only provided for Euclidean problems (where the domain $\mathcal{W}$ and Lipschitz parameter of f_t scale with the L_2 norm).

In this note, we present and analyze a simple algorithm with the following properties:

- For Euclidean problems, it is optimal up to constants for both smooth and non-smooth functions. This closes the gap between the smooth and non-smooth Euclidean problems in this setting.
- The algorithm and analysis are readily applicable to non-Euclidean problems. We give an example for the 1-norm, with the resulting bound optimal up to logarithmic factors.
- The algorithm and analysis are simpler than those proposed in Duchi et al. (2015). They apply equally to the bandit and zero-order optimization setting, and can be readily extended using standard techniques, e.g. improved bounds for strongly-convex functions; regret/error bounds holding with high-probability rather than just in expectation; and improved bounds if allowed $k > 2$ observations per round instead of just two (Hazan et al., 2007; Shalev-Shwartz, 2007; Agarwal et al., 2010).

Like previous algorithms, our algorithm is based on a random gradient estimator, which given a function f and point $\mathbf{w}$, queries f at two random locations close to $\mathbf{w}$, and computes a random vector whose expectation is a gradient of a smoothed version of f. The papers Nesterov (2011); Duchi et al. (2015); Ghadimi and Lan (2013) essentially use the estimator

1. This is slightly different than the model of Agarwal et al. (2010), where the learner only chooses $\mathbf{w}'_t, \mathbf{w}''_t$ and the loss is $\frac{1}{2}(f_t(\mathbf{w}'_t) + f_t(\mathbf{w}''_t))$. However, our results and analysis can be easily translated to their setting, and the model we discuss translates more directly to the zero-order stochastic optimization considered later.

©2017 Ohad Shamir.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-632.html.

which queries at $\mathbf{w}$ and $\mathbf{w}+\delta\mathbf{u}$ (where $\mathbf{u}$ is a random unit vector and $\delta>0$ is a small parameter), and returns

$$\frac{d}{\delta}\left(f(\mathbf{w}+\delta\mathbf{u})-f(\mathbf{w})\right)\mathbf{u}. \tag{1}$$

The intuition is readily seen in the one-dimensional ($d=1$) case, where the expectation of this expression equals

$$\frac{1}{2\delta}\left(f(w+\delta)-f(w-\delta)\right), \tag{2}$$

which indeed approximates the derivative of f (assuming f is differentiable) at w, if δ is small enough.

In contrast, our algorithm uses a slightly different estimator (also used in Agarwal et al., 2010), which queries at $\mathbf{w}-\delta\mathbf{u},\mathbf{w}+\delta\mathbf{u}$, and returns

$$\frac{d}{2\delta}\left(f(\mathbf{w}+\delta\mathbf{u})-f(\mathbf{w}-\delta\mathbf{u})\right)\mathbf{u}. \tag{3}$$

Again, the intuition is readily seen in the case $d=1$, where the expectation of this expression also equals Eq. (2).

When δ is sufficiently small and f is differentiable at $\mathbf{w}$, both estimators compute a good approximation of the true gradient $\nabla f(\mathbf{w})$. However, when f is not differentiable, the variance of the estimator in Eq. (1) can be quadratic in the dimension d, as pointed out by Duchi et al. (2015): For example, for $f(\mathbf{w})=\|\mathbf{w}\|_2$ and $\mathbf{w}=0$, the second moment equals

$$\mathbb{E}\left[\left\|\frac{d}{\delta}\left(f(\delta\mathbf{u})-f(\mathbf{0})\right)\mathbf{u}\right\|_2^2\right] \;=\; \mathbb{E}\left[d^2\|\mathbf{u}\|_2^4\right] \;=\; d^2.$$

Since the performance of the algorithm crucially depends on the second moment of the gradient estimate, this leads to a highly sub-optimal guarantee. In Duchi et al. (2015), this was handled by adding an additional random perturbation and using a more involved analysis. Surprisingly, it turns out that the slightly different estimator in Eq. (3) does not suffer from this problem, and its second moment is essentially *linear* in the dimension d.

We note that in this work, we assume that $\mathbf{u}$ is a random unit vector, similar to previous works. However, our results can be readily extended to other distributions, such as uniform in the Euclidean unit ball, or a Gaussian distribution.

2. Algorithm and Main Results

We consider the algorithm described in Figure 1, which performs standard mirror descent using a randomized gradient estimator $\tilde{\mathbf{g}}_t$ of a (smoothed) version of f_t at point $\mathbf{w}_t$. Following Duchi et al. (2015), we assume that one can indeed query f_t at any point $\mathbf{w}_t+\delta_t\mathbf{u}_t$ as specified in the algorithm[2].

The analysis of the algorithm is presented in the following theorem:

2. This may require us to query at a distance δ_t outside $\mathcal{W}$. If we must query within $\mathcal{W}$, then a standard technique (see Agarwal et al., 2010) is to simply run the algorithm on a slightly smaller set $(1-\epsilon)\mathcal{W}$, where $\epsilon>0$ is sufficiently large so that $\mathbf{w}_t+\delta_t\mathbf{u}_t$ must be in $\mathcal{W}$. Since the formal guarantee in Thm. 1 holds for arbitrarily small δ_t, and each f_t is Lipschitz, we can generally take δ_t (and hence ϵ) sufficiently small so that the additional regret/error incurred is arbitrarily small.

Algorithm 1 Two-Point Bandit Convex Optimization Algorithm

Input: Step size η, function $r:\mathcal{W}\mapsto\mathbb{R}$, exploration parameters $\delta_t>0$
Initialize $\boldsymbol{\theta}_1=\mathbf{0}$.
for $t=1,\ldots,T-1$ **do**
  Predict $\mathbf{w}_t=\arg\max_{\mathbf{w}\in\mathcal{W}}\langle\boldsymbol{\theta}_t,\mathbf{w}\rangle-r(\mathbf{w})$
  Sample $\mathbf{u}_t$ uniformly from the Euclidean unit sphere $\{\mathbf{w}:\|\mathbf{w}\|_2=1\}$
  Query $f_t(\mathbf{w}_t+\delta_t\mathbf{u}_t)$ and $f_t(\mathbf{w}_t-\delta_t\mathbf{u}_t)$
  Set $\tilde{\mathbf{g}}_t=\frac{d}{2\delta_t}\left(f_t(\mathbf{w}_t+\delta_t\mathbf{u}_t)-f_t(\mathbf{w}_t-\delta_t\mathbf{u}_t)\right)\mathbf{u}_t$
  Update $\boldsymbol{\theta}_{t+1}=\boldsymbol{\theta}_t-\eta\tilde{\mathbf{g}}_t$
end for

Theorem 1 *Assume the following conditions hold:*

1. *r is 1-strongly convex with respect to a norm $\|\cdot\|$, and $\sup_{\mathbf{w}\in\mathcal{W}}r(\mathbf{w})\leq R^2$ for some $R<\infty$.*
2. *f_t is convex and G_2-Lipschitz with respect to the 2-norm $\|\cdot\|_2$.*
3. *The dual norm $\|\cdot\|_*$ of $\|\cdot\|$ is such that $\sqrt[4]{\mathbb{E}_{\mathbf{u}_t}\|\mathbf{u}_t\|_*^4}\leq p_*$ for some $p_*<\infty$.*

If $\eta=\frac{R}{p_*G_2\sqrt{dT}}$, and δ_t chosen such that $\delta_t\leq p_*R\sqrt{\frac{d}{T}}$, then the sequence $\mathbf{w}_1,\ldots,\mathbf{w}_T$ generated by the algorithm satisfies the following for any T and $\mathbf{w}^*\in\mathcal{W}$:

$$\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^T f_t(\mathbf{w}_t)-\frac{1}{T}\sum_{t=1}^T f_t(\mathbf{w}^*)\right]\leq c\;p_*G_2R\sqrt{\frac{d}{T}},$$

where c is some numerical constant.

We note that condition 1 is standard in the analysis of the mirror-descent method (see the specific corollaries below), whereas conditions 2 and 3 are needed to ensure that the variance of our gradient estimator is controlled.

As mentioned earlier, the bound on the average regret which appears in Thm. 1 immediately implies a similar bound on the error in a stochastic optimization setting, for the average point $\bar{\mathbf{w}}_T=\frac{1}{T}\sum_{t=1}^T\mathbf{w}_t$. We note that the result is robust to the choice of η, and is the same up to constants as long as $\eta=\Theta(R/p_*G_2\sqrt{dT})$. Also, the constant c, while always strictly positive, shrinks as $\delta_t\to 0$ (see the proof below for details).

As a first application of the theorem, let us consider the case where $\|\cdot\|$ is the Euclidean norm $\|\cdot\|_2$. In this case, we can take $r(\mathbf{w})=\frac{1}{2}\|\mathbf{w}\|_2^2$, and the algorithm reduces to a standard variant of online gradient descent, defined as $\boldsymbol{\theta}_{t+1}=\boldsymbol{\theta}_t-\tilde{\mathbf{g}}_t$ and $\mathbf{w}_t=\arg\min_{\mathbf{w}\in\mathcal{W}}\|w-\boldsymbol{\theta}_t\|_2$. In this case, we get the following corollary:

Corollary 2 *Suppose f_t for all t is G_2-Lipschitz with respect to the Euclidean norm, and $\mathcal{W}\subseteq\{\mathbf{w}:\|\mathbf{w}\|_2\leq R\}$. Then using $\|\cdot\|=\|\cdot\|_2$ and $r(\mathbf{w})=\frac{1}{2}\|\mathbf{w}\|_2^2$, it holds for some constant c and any $\mathbf{w}^*\in\mathcal{W}$ that*

$$\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^T f_t(\mathbf{w}_t)-\frac{1}{T}\sum_{t=1}^T f_t(\mathbf{w}^*)\right]\leq c\;G_2R\sqrt{\frac{d}{T}},$$

The proof is immediately obtained from Thm. 1, noting that $p_* = 1$ in our case. This bound matches (up to constants) the lower bound in Duchi et al. (2015), hence closing the gap between upper and lower bounds in this setting.

As a second application, let us consider the case where $\|\cdot\|$ is the 1-norm, $\|\cdot\|_1$, the domain $\mathcal{W}$ is the simplex in $\mathbb{R}^d$, $d > 1$ (although our result easily extends to any subset of the 1-norm unit ball), and we use a standard entropic regularizer:

Corollary 3 *Suppose f_t for all t is G_1-Lipschitz with respect to the L_1 norm. Then using $\|\cdot\| = \|\cdot\|_1$ and $r(\mathbf{w}) = \sum_{i=1}^d w_i \log(dw_i)$, it holds for some constant c and any $\mathbf{w}^* \in \mathcal{W}$ that*

$$\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^T f_t(\mathbf{w}_t) - \frac{1}{T}\sum_{t=1}^T f_t(\mathbf{w}^*)\right] \leq c\, G_1 \sqrt{\frac{d\log^2(d)}{T}}.$$

This bound matches (this time up to a factor polylogarithmic in d) the lower bound in Duchi et al. (2015) for this setting.

Proof The function r is 1-strongly convex with respect to the 1-norm (see for instance Shalev-Shwartz, 2012, Example 2.5), and has value at most $\log(d)$ on the simplex. Also, if f_t is G_1-Lipschitz with respect to the 1-norm, then it must be $\sqrt{d}G_1$-Lipschitz with respect to the Euclidean norm. Finally, to satisfy condition 3 in Thm. 1, we upper bound $\sqrt[4]{\mathbb{E}[\|\mathbf{u}_t\|_\infty^4]}$ using the following lemma, whose proof is given in the appendix:

Lemma 4 *If $\mathbf{u}$ is uniformly distributed on the unit sphere in $\mathbb{R}^d$, $d > 1$, then $\sqrt[4]{\mathbb{E}[\|\mathbf{u}\|_\infty^4]} \leq c\sqrt{\frac{\log(d)}{d}}$ where c is a positive numerical constant independent of d.*

Plugging these observations into Thm. 1 leads to the desired result. ■

Finally, we make two additional remarks on possible extensions and improvements to Thm. 1.

Remark 5 (Querying at $k > 2$ points) *If the algorithm is allowed to query f_t at $k > 2$, then it can be modified to attain an improved regret bound, by computing $\lfloor k/2 \rfloor$ independent estimates of $\tilde{g}_t$ at every round (using a freshly sampled $\mathbf{u}_t$ each time), and using their average. This leads to a new gradient estimator $\tilde{\mathbf{g}}_t^k$, which satisfies $\mathbb{E}[\|\tilde{\mathbf{g}}_t^k\|^2] \leq \frac{1}{k}\mathbb{E}[\|\tilde{\mathbf{g}}_t\|^2] + \|\mathbb{E}[\tilde{\mathbf{g}}_t]\|^2$. Based on the proof of Thm. 1, it is easily verified that this leads to an average expected regret bound of $\frac{cG_2R}{\sqrt{T}}\left(1 + p_*\sqrt{d/k}\right)$ for some numerical constant c.*

Remark 6 (Non-Euclidean Geometries) *When considering norms other than the Euclidean norm, it is tempting to conjecture that our algorithm and analysis can be improved, by sampling $\mathbf{u}_t$ from a distribution adapted to the geometry of that norm (not necessarily the Euclidean ball), and assuming f_t is Lipschitz w.r.t. the dual norm. However, adapting the proof (and in particular getting appropriate versions of Lemma 8 and Lemma 9) does not appear straightforward, and the potential performance improvement is currently unclear.*

3. Proof of Theorem 1

As discussed in the introduction, the key to getting improved results compared to previous papers is the use of a slightly different random gradient estimator, which turns out to have significantly less variance. The formal proof relies on a few simple lemmas listed below. The key lemma is Lemma 10, which establishes the improved variance behavior.

Lemma 7 *For any $\mathbf{w}^* \in \mathcal{W}$, it holds that*

$$\sum_{t=1}^T \langle \tilde{\mathbf{g}}_t, \mathbf{w}_t - \mathbf{w}^* \rangle \leq \frac{1}{\eta}R^2 + \eta\sum_{t=1}^T \|\tilde{\mathbf{g}}_t\|_*^2.$$

This lemma is the canonical result on the convergence of online mirror descent, and the proof is standard (see e.g. Shalev-Shwartz, 2012).

Lemma 8 *Define the function*

$$\hat{f}_t(\mathbf{w}) = \mathbb{E}_{\mathbf{u}_t}\left[f_t(\mathbf{w} + \delta_t\mathbf{u}_t)\right],$$

over $\mathcal{W}$, where $\mathbf{u}_t$ is a vector picked uniformly at random from the Euclidean unit sphere. Then the function is convex, Lipschitz with constant G_2, satisfies

$$\sup_{\mathbf{w}\in\mathcal{W}} |\hat{f}_t(\mathbf{w}) - f_t(\mathbf{w})| \leq \delta_t G_2,$$

and is differentiable with the following gradient:

$$\nabla\hat{f}_t(\mathbf{w}) = \mathbb{E}_{\mathbf{u}_t}\left[\frac{d}{\delta_t}f_t(\mathbf{w} + \delta_t\mathbf{u}_t)\mathbf{u}_t\right].$$

Proof The fact that the function is convex and Lipschitz is immediate from its definition and the assumptions in the theorem. The inequality follows from $\mathbf{u}_t$ being a unit vector and that f_t is assumed to be G_2-Lipschitz with respect to the 2-norm. The differentiability property follows from Lemma 2.1 in Flaxman et al. (2005). ■

Lemma 9 *For any function g which is L-Lipschitz with respect to the 2-norm, it holds that if $\mathbf{u}$ is uniformly distributed on the Euclidean unit sphere, then*

$$\sqrt{\mathbb{E}\left[\left(g(\mathbf{u}) - \mathbb{E}[g(\mathbf{u})]\right)^4\right]} \leq c\frac{L^2}{d}.$$

for some numerical constant c.

Proof A standard result on the concentration of Lipschitz functions on the Euclidean unit sphere implies that

$$\Pr(|g(\mathbf{u}) - \mathbb{E}[g(\mathbf{u})]| > t) \leq 2\exp\left(-c'dt^2/L^2\right)$$

for some numerical constant $c' > 0$ (see the proof of Proposition 2.10 and Corollary 2.6 in Ledoux, 2005). Therefore,

$$\sqrt{\mathbb{E}\left[(g(\mathbf{u}) - \mathbb{E}[g(\mathbf{u})])^4\right]} = \sqrt{\int_{t=0}^{\infty} \Pr\left((g(\mathbf{u}) - \mathbb{E}[g(\mathbf{u})])^4 > t\right) dt}$$
$$= \sqrt{\int_{t=0}^{\infty} \Pr\left(|g(\mathbf{u}) - \mathbb{E}[g(\mathbf{u})]| > \sqrt[4]{t}\right) dt} \leq \sqrt{\int_{t=0}^{\infty} 2\exp\left(-\frac{c'd\sqrt{t}}{L^2}\right) dt} = \sqrt{2\frac{L^4}{(c'd)^2}},$$

where in the last step we used the fact that $\int_{x=0}^{\infty} \exp(-\sqrt{x})dx = 2$. The expression above equals cL^2/d for some numerical constant c. ■

Lemma 10 *It holds that $\mathbb{E}[\tilde{\mathbf{g}}_t|\mathbf{w}_t] = \nabla \hat{f}_t(\mathbf{w}_t)$ (where $\hat{f}_t(\cdot)$ is as defined in Lemma 8), and $\mathbb{E}[\|\tilde{\mathbf{g}}_t\|^2|\mathbf{w}_t] \leq cdp_*^2 G_2^2$ for some numerical constant c.*

Proof For simplicity of notation, we drop the t subscript. Since $\mathbf{u}$ has a symmetric distribution around the origin,

$$\begin{aligned}
\mathbb{E}[\tilde{\mathbf{g}}|\mathbf{w}] &= \mathbb{E}_{\mathbf{u}}\left[\frac{d}{2\delta}\left(f(\mathbf{w}+\delta\mathbf{u}) - f(\mathbf{w}-\delta\mathbf{u})\right)\mathbf{u}\right] \\
&= \mathbb{E}_{\mathbf{u}}\left[\frac{d}{2\delta}\left(f(\mathbf{w}+\delta\mathbf{u})\right)\mathbf{u}\right] + \mathbb{E}_{\mathbf{u}}\left[\frac{d}{2\delta}f(\mathbf{w}-\delta\mathbf{u})(-\mathbf{u})\right] \\
&= \mathbb{E}_{\mathbf{u}}\left[\frac{d}{2\delta}\left(f(\mathbf{w}+\delta\mathbf{u})\right)\mathbf{u}\right] + \mathbb{E}_{\mathbf{u}}\left[\frac{d}{2\delta}f(\mathbf{w}+\delta\mathbf{u})(\mathbf{u})\right] \\
&= \mathbb{E}_{\mathbf{u}}\left[\frac{d}{\delta}f(\mathbf{w}+\delta\mathbf{u})\mathbf{u}\right]
\end{aligned}$$

which equals $\nabla\hat{f}(\mathbf{w})$ by Lemma 8.

As to the second part of the lemma, we have the following, where α is an arbitrary parameter and where we use the elementary inequality $(a-b)^2 \leq 2(a^2+b^2)$.

$$\begin{aligned}
\mathbb{E}[\|\tilde{\mathbf{g}}\|_*^2|\mathbf{w}] &= \mathbb{E}_{\mathbf{u}}\left[\left\|\frac{d}{2\delta}\left(f(\mathbf{w}+\delta\mathbf{u}) - f(\mathbf{w}-\delta\mathbf{u})\right)\mathbf{u}\right\|_*^2\right] \\
&= \frac{d^2}{4\delta^2}\mathbb{E}_{\mathbf{u}}\left[\|\mathbf{u}\|_*^2\left(f(\mathbf{w}+\delta\mathbf{u}) - f(\mathbf{w}-\delta\mathbf{u})\right)^2\right] \\
&= \frac{d^2}{4\delta^2}\mathbb{E}_{\mathbf{u}}\left[\|\mathbf{u}\|_*^2\left(\left(f(\mathbf{w}+\delta\mathbf{u}) - \alpha\right) - \left(f(\mathbf{w}-\delta\mathbf{u}) - \alpha\right)\right)^2\right] \\
&\leq \frac{d^2}{2\delta^2}\mathbb{E}_{\mathbf{u}}\left[\|\mathbf{u}\|_*^2\left(\left(f(\mathbf{w}+\delta\mathbf{u}) - \alpha\right)^2 + \left(f(\mathbf{w}-\delta\mathbf{u}) - \alpha\right)^2\right)\right] \\
&= \frac{d^2}{2\delta^2}\left(\mathbb{E}_{\mathbf{u}}\left[\|\mathbf{u}\|_*^2\left(f(\mathbf{w}+\delta\mathbf{u}) - \alpha\right)^2\right] + \mathbb{E}_{\mathbf{u}}\left[\|\mathbf{u}\|_*^2\left(f(\mathbf{w}-\delta\mathbf{u}) - \alpha\right)^2\right]\right).
\end{aligned}$$

Again using the symmetric distribution of $\mathbf{u}$, this equals

$$\begin{aligned}
&\frac{d^2}{2\delta^2}\left(\mathbb{E}_{\mathbf{u}}\left[\|\mathbf{u}\|_*^2\left(f(\mathbf{w}+\delta\mathbf{u}) - \alpha\right)^2\right] + \mathbb{E}_{\mathbf{u}}\left[\|\mathbf{u}\|_*^2\left(f(\mathbf{w}+\delta\mathbf{u}) - \alpha\right)^2\right]\right) \\
&\quad = \frac{d^2}{\delta^2}\mathbb{E}_{\mathbf{u}}\left[\|\mathbf{u}\|_*^2\left(f(\mathbf{w}+\delta\mathbf{u}) - \alpha\right)^2\right].
\end{aligned}$$

Applying Cauchy-Schwartz and using the condition $\sqrt[4]{\mathbb{E}_{\mathbf{u}}\|\mathbf{u}\|_*^4} \leq p_*$ stated in the theorem, we get the upper bound

$$\frac{d^2}{\delta^2}\sqrt{\mathbb{E}_{\mathbf{u}}\left[\|\mathbf{u}\|_*^4\right]}\sqrt{\mathbb{E}_{\mathbf{u}}\left[\left(f(\mathbf{w}+\delta\mathbf{u}) - \alpha\right)^4\right]} = \frac{p_*^2 d^2}{\delta^2}\sqrt{\mathbb{E}_{\mathbf{u}}\left[\left(f(\mathbf{w}+\delta\mathbf{u}) - \alpha\right)^4\right]}.$$

In particular, taking $\alpha = \mathbb{E}_{\mathbf{u}}[f(\mathbf{w}+\delta\mathbf{u})]$ and using Lemma 9 (noting that $f(\mathbf{w}+\delta\mathbf{u})$ is $G_2\delta$-Lipschitz w.r.t. $\mathbf{u}$ in terms of the 2-norm), this is at most $\frac{p_*^2 d^2}{\delta^2} c\frac{(G_2\delta)^2}{d} = cdp_*^2 G_2^2$ as required. ■

We are now ready to prove the theorem. Taking expectations on both sides of the inequality in Lemma 7, we have

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle\tilde{\mathbf{g}}_t, \mathbf{w}_t - \mathbf{w}^*\rangle\right] \leq \frac{1}{\eta}R^2 + \eta\sum_{t=1}^{T}\mathbb{E}\left[\|\tilde{\mathbf{g}}_t\|_*^2\right] = \frac{1}{\eta}R^2 + \eta\sum_{t=1}^{T}\mathbb{E}\left[\mathbb{E}\left[\|\tilde{\mathbf{g}}_t\|_*^2|\mathbf{w}_t\right]\right]. \quad (4)$$

Using Lemma 10, the right hand side is at most

$$\frac{1}{\eta}R^2 + \eta cdp_*^2 G_2^2 T$$

The left hand side of Eq. (4), by Lemma 10 and convexity of $\hat{f}_t$, equals

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle\mathbb{E}[\tilde{\mathbf{g}}_t|\mathbf{w}_t], \mathbf{w}_t - \mathbf{w}^*\rangle\right] = \mathbb{E}\left[\sum_{t=1}^{T}\langle\nabla\hat{f}_t(\mathbf{w}_t), \mathbf{w}_t - \mathbf{w}^*\rangle\right] \geq \mathbb{E}\left[\sum_{t=1}^{T}\left(\hat{f}_t(\mathbf{w}_t) - \hat{f}_t(\mathbf{w}^*)\right)\right].$$

By Lemma 8, this is at least

$$\mathbb{E}\left[\sum_{t=1}^{T}\left(f_t(\mathbf{w}_t) - f_t(\mathbf{w}^*)\right)\right] - 2G_2\sum_{t=1}^{T}\delta_t.$$

Combining these inequalities and plugging back into Eq. (4), we get

$$\mathbb{E}\left[\sum_{t=1}^{T}\left(f_t(\mathbf{w}_t) - f_t(\mathbf{w}^*)\right)\right] \leq 2G_2\sum_{t=1}^{T}\delta_t + \frac{1}{\eta}R^2 + cdp_*^2 G_2^2\eta T.$$

Choosing $\eta = R/(p_* G_2\sqrt{dT})$, and any $\delta_t \leq p_* R\sqrt{d/T}$, we get

$$\mathbb{E}\left[\sum_{t=1}^{T}\left(f_t(\mathbf{w}_t) - f_t(\mathbf{w}^*)\right)\right] \leq (c+3)p_* G_2 R\sqrt{dT}.$$

Dividing both sides by T, the result follows.

Acknowledgments

This research was supported in part by an Israel Science Foundation grant 425/13 and an FP7 Marie Curie CIG grant. We thank the anonymous reviewers for several helpful comments.

Appendix A. Proof of Lemma 4

We note that the distribution of $\|\mathbf{u}\|_\infty^4$ is identical to that of $\frac{\|\mathbf{n}\|_\infty^4}{\|\mathbf{n}\|_2^4}$, where $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, I_d)$ is a standard Gaussian random vector. Moreover, by a standard concentration bound on the norm of Gaussian random vectors (e.g. Corollary 2.3 in Barvinok, 2005, with $\epsilon = 1/2$):

$$\max\left\{\Pr\left(\|\mathbf{n}\|_2 \leq \sqrt{\frac{d}{2}}\right), \Pr\left(\|\mathbf{n}\|_2 \geq \sqrt{2d}\right)\right\} \leq \exp\left(-\frac{d}{16}\right).$$

Finally, for any value of $\mathbf{n}$, we always have $\frac{\|\mathbf{n}\|_\infty}{\|\mathbf{n}\|_2} \leq 1$, since the Euclidean norm is always larger than the infinity norm. Combining these observations, and using $\mathbf{1}_A$ for the indicator function of the event A, we have

$$\begin{aligned}
\mathbb{E}[\|\mathbf{u}\|_\infty^4] &= \mathbb{E}\left[\frac{\|\mathbf{n}\|_\infty^4}{\|\mathbf{n}\|_2^4}\right] \\
&= \Pr\left(\|\mathbf{n}\|_2 \leq \sqrt{\frac{d}{2}}\right)\mathbb{E}\left[\frac{\|\mathbf{n}\|_\infty^4}{\|\mathbf{n}\|_2^4} \;\middle|\; \|\mathbf{n}\|_2 \leq \sqrt{\frac{d}{2}}\right] \\
&\quad + \Pr\left(\|\mathbf{n}\|_2 > \sqrt{\frac{d}{2}}\right)\mathbb{E}\left[\frac{\|\mathbf{n}\|_\infty^4}{\|\mathbf{n}\|_2^4} \;\middle|\; \|\mathbf{n}\|_2 > \sqrt{\frac{d}{2}}\right] \\
&\leq \exp\left(-\frac{d}{16}\right) * 1 + \Pr\left(\|\mathbf{n}\|_2 > \sqrt{\frac{d}{2}}\right)\mathbb{E}\left[\frac{\|\mathbf{n}\|_\infty^4}{\left(\sqrt{d/2}\right)^4} \;\middle|\; \|\mathbf{n}\|_2 > \sqrt{\frac{d}{2}}\right] \\
&= \exp\left(-\frac{d}{16}\right) + \left(\frac{2}{d}\right)^2 \mathbb{E}\left[\|\mathbf{n}\|_\infty^4 \mathbf{1}_{\|\mathbf{n}\|_2 > \sqrt{d/2}}\right] \\
&\leq \exp\left(-\frac{d}{16}\right) + \frac{4}{d^2}\mathbb{E}\left[\|\mathbf{n}\|_\infty^4\right]. \qquad (5)
\end{aligned}$$

Thus, it remains to upper bound $\mathbb{E}\left[\|\mathbf{n}\|_\infty^4\right]$ where $\mathbf{n}$ is a standard Gaussian random variable. Letting $\mathbf{n} = (n_1, \ldots, n_d)$, and noting that $n_1, \ldots, n_d$ are independent and identically

distributed standard Gaussian random variables, we have for any scalar $z \geq 1$ that

$$\begin{aligned}
\Pr(\|\mathbf{n}\|_\infty \leq z) = \prod_{i=1}^n \Pr(|n_i| \leq z) &= (\Pr(|n_1| \leq z))^d \\
&= (1 - \Pr(|n_1| > z))^d \overset{(1)}{\geq} 1 - d\Pr(|n_1| > z) \\
&= 1 - 2d\Pr(n_1 > z) \overset{(2)}{\geq} 1 - d\exp(-z^2/2),
\end{aligned}$$

where (1) is Bernoulli's inequality, and (2) is using a standard tail bound for a Gaussian random variable. In particular, the above implies that

$$\Pr\left(\|\mathbf{n}\|_\infty > z\right) \leq d\exp(-z^2/2).$$

Therefore, for an arbitrary positive scalar $r \geq 1$,

$$\begin{aligned}
\mathbb{E}\left[\|\mathbf{n}\|_\infty^4\right] &= \int_{z=0}^\infty \Pr\left(\|\mathbf{n}\|_\infty^4 > z\right) dz \\
&\leq \int_{z=0}^r 1 dz + \int_{z=r}^\infty \Pr\left(\|\mathbf{n}\|_\infty > \sqrt[4]{z}\right) dz \\
&\leq r + \int_{z=r}^\infty d\exp\left(-\frac{\sqrt{z}}{2}\right) dz \\
&= r + 4d(2 + \sqrt{r})\exp\left(-\frac{\sqrt{r}}{2}\right).
\end{aligned}$$

In particular, plugging $r = 4\log^2(d)$ (which is larger than 1, since we assume $d > 1$), we get $4(2 + 2\log(d) + \log^2(d))$. Plugging this back into Eq. (5), we get that

$$\mathbb{E}[\|\mathbf{u}\|_\infty^4] \;\leq\; \exp\left(-\frac{d}{16}\right) + 16\frac{2 + 2\log(d) + \log^2(d)}{d^2},$$

which can be shown to be at most $c'\left(\frac{\log(d)}{d}\right)^2$ for all $d > 1$, where $c' < 150$ is a numerical constant. In particular, this means that $\sqrt[4]{\mathbb{E}[\|\mathbf{u}\|_\infty^4]} \leq \sqrt[4]{c'}\sqrt{\frac{\log(d)}{d}}$ as required.

References

A. Agarwal, O. Dekel, and L. Xiao. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *Conference on Learning Theory (COLT)*, 2010.

A. Barvinok. Measure concentration lecture notes. `http://www.math.lsa.umich.edu/~barvinok/total710.pdf`, 2005.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *Information Theory, IEEE Transactions on*, 50(9):2050–2057, 2004.

J. Duchi, M. Jordan, M. Wainwright, and A. Wibisono. Optimal rates for zero-order optimization: the power of two function evaluations. *Information Theory, IEEE Transactions on*, 61(5):2788–2806, May 2015.

A. Flaxman, A. Kalai, and B. McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2005.

S. Ghadimi and G. Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

E. Hazan, A. Agarwal, and S. Kale. Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2):169–192, 2007.

M. Ledoux. *The concentration of measure phenomenon*, volume 89. American Mathematical Soc., 2005.

Y. Nesterov. Random gradient-free minimization of convex functions. Technical Report 2011/16, ECORE, 2011.

S. Shalev-Shwartz. *Online learning: Theory, algorithms, and applications*. PhD thesis, The Hebrew University, 2007.

S. Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2), 2012.

Journal of Machine Learning Research 18 (2017) 1-31 Submitted 2/17; Revised 4/17; Published 5/17

Perishability of Data: Dynamic Pricing under Varying-Coefficient Models

Adel Javanmard AJAVANMA@USC.EDU
Department of Data Sciences and Operations
Marshall School of Business
University of Southern California
Los Angeles, CA 90089 , USA

Editor: Qiang Liu

Abstract

We consider a firm that sells a large number of products to its customers in an online fashion. Each product is described by a high dimensional feature vector, and the market value of a product is assumed to be linear in the values of its features. Parameters of the valuation model are unknown and can change over time. The firm sequentially observes a product's features and can use the historical sales data (binary sale/no sale feedbacks) to set the price of current product, with the objective of maximizing the collected revenue. We measure the performance of a dynamic pricing policy via regret, which is the expected revenue loss compared to a clairvoyant that knows the sequence of model parameters in advance.

We propose a pricing policy based on projected stochastic gradient descent (PSGD) and characterize its regret in terms of time T, features dimension d, and the temporal variability in the model parameters, δ_t. We consider two settings. In the first one, feature vectors are chosen antagonistically by nature and we prove that the regret of PSGD pricing policy is of order $O(\sqrt{T}+\sum_{t=1}^{T}\sqrt{t}\delta_t)$. In the second setting (referred to as stochastic features model), the feature vectors are drawn independently from an unknown distribution. We show that in this case, the regret of PSGD pricing policy is of order $O(d^2\log T+\sum_{t=1}^{T}t\delta_t/d)$.

Keywords: Dynamic Pricing, Revenue Management, Varying-Coefficient Models, Regret, Stochastic Gradient Descent, Hypothesis Testing

1. Introduction

Motivated by the prevalence of online marketplaces, we consider the problem of a firm selling a large number of products, that are significantly differentiated from each other, to customers that arrive over time. The firm needs to price the products in a dynamic manner, with the objective of maximizing the expected revenue.

The majority of work in dynamic pricing assume that a retailer sells *identical* items to its customers (Besbes and Zeevi, 2009; Farias and Van Roy, 2010; Broder and Rusmevichientong, 2012; den Boer and Zwart, 2013; Wang et al., 2014). Recently, feature-based models have been used to model the products differentiation by assuming that each product is described by vectors of high-dimensional features. These models are suitable for business settings where there are an enormous number of distinct products. One important example is online ad markets. In this context, products are the impressions (user view) that are sold by the web publisher to advertisers. Due to the ever-growing amount of data that is available on the Internet, for each impression there is large number of associated features, including demographic information, browsing history of the user, and context of the webpage. Many other online markets, such as Airbnb, eBay and Etsy also have a similar setting in which products to be sold are highly differentiated. For example, in the case of Aribnb, the products are "stays" and each is characterized by a large number of features including space properties, location, amenities, house rules, as well as arrival dates, events in the area, availability of near-by hotels, etc (Airbnb Documentation, 2015).

Here, we consider a feature-based model that postulates a linear relation between the market value of each product and its feature values. Further, from the firm's perspective, we treat distinct buyers independently, and hereafter focus on a single buyer. Put it formally, we start with the following model for the buyer's valuation:

$$v(x_t) = \langle x_t, \theta\rangle + z_t\,, \tag{1}$$

where $x_t \in \mathbb{R}^d$ denotes the product feature vector, θ represents the buyer's preferences and z_t, $t \geq 1$ are idiosyncratic shocks, referred to as noise, which are drawn independently and identically from a zero mean distribution. For two vectors a, b, we write $\langle a, b\rangle$ to refer to their inner product. Feature vectors x_t are observable, while model parameter θ is a-priori unknown to the firm (seller). Therefore, the buyer's valuation $v(x_t)$ is also hidden from the firm.

Parameters of the above model represents how different features are weighted by the buyer in assessing the product. Considering such model, a firm can use historical sales data to estimate parameters of the valuation model, while concurrently collecting revenue from new sales. In practice, though, the buyer's valuation of a product will change over time and this raises the concern of *perishability* of sales data.

In order to capture this point, we consider a richer model with varying coefficients:

$$v_t(x_t) = \langle x_t, \theta_t\rangle + z_t\,. \tag{2}$$

Model parameters θ_t may change over time and as a result, valuation of a product depends on both the product feature vector and the time index.

We study a dynamic pricing problem, where at each time period t, the firm has a product to sell and after observing the product feature vector x_t, posts a price p_t. If the buyer's valuation is above the posted price, $v_t(x_t) \geq p_t$, a sale occurs and the firm collects a revenue of p_t. If the posted price exceeds the buyer's valuation, $p_t > v_t(x_t)$, no sale occurs. Note that at each step, the firm has access to the previous feedbacks (sale/no sale) from the buyer and can use this information in setting the current price.

In this paper, we will analyze the varying-coefficient model (2) and answer two fundamental questions:

> First, what is the value of knowing the sequence of model parameters θ_t; in other words, what is the expected revenue loss (regret) compared to the clairvoyant policy that knows the parameters of the valuation model in advance? Second, what is a good pricing policy?

The answer to the first question intrinsically depends on the temporal variability in the sequence θ_t. If this variation is very large, then there is not much that can be learnt

©2017 Adel Javanmard.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/17-061.html.

from previous feedback on the buyer's behavior and the problem turns into a random price experimentation. On the other hand, if all of the parameters θ_t are the same, then this feedback information can be used to learn the model parameters, which in turn helps in setting the future prices. In this case, an algorithm that performs a good balance between price exploration and best-guess pricing (exploitation) can lead to a small regret. In this work, we study this trade-off through a projected stochastic gradient descent algorithm and investigate the effect of variations of the sequence of θ_t on the regret bounds.

Feature-based models have recently attracted interest in dynamic pricing. (Amin et al., 2014) studied a similar model to (1) (without the noise terms z_t), where the features x_t are drawn from an unknown i.i.d distribution. A pricing strategy was proposed based on stochastic gradient descent, which results in a regret of the form $O(T^{2/3}\sqrt{\log T})$. This work also studied the problem of dynamic incentive compatibility in repeated posted-price auctions. Subsequently, (Cohen et al., 2016) studied model (1), wherein the feature vectors x_t are chosen antagonistically by nature and not sampled i.i.d. This work proposes a pricing policy based on the ellipsoid method from convex optimization (Boyd and Vandenberghe, 2004) with a regret bound of $O(d^2\log(T/d))$, under a low-noise setting. More accurately, the regret scales as $O(d^2\log(\min\{T/d, 1/\delta\}) + d\delta T)$, where δ measures the noise magnitude: in case of bounded noise, δ represents the uniform bound on noise and in case of gaussian noise with variance σ^2, it is defined as $\delta = 2\sigma\sqrt{\log(T)}$. In (Lobel et al., 2016), the regret bound of this policy was improved to $O(d\log T)$, under the noiseless setting. In (Javanmard and Nazerzadeh, 2016), authors study and highlight the role of the structure of demand curve in dynamic pricing. They introduce model (1), and assume that the feature vectors x_t are drawn i.i.d. from an unknown distribution. Further, motivated by real-world applications, it is assumed that the parameter vector θ is sparse in the sense that only a few of its entries are nonzero. A regularized log-likelihood approach is taken to get an improved regret bound of order $s_0(\log(d)+\log(T))$. We add to this body of work by considering feature-based models for valuation of products whose parameters vary over time.

Time-varying demand environments have also been studied recently by (Keskin and Zeevi, 2016). Explicitly, they consider a firm that sells one type of product to customers that arrive over a time horizon. After setting price p_t, the firm observes demand D_t given by $D_t = \alpha_t + \beta_t p_t + \epsilon_t$, where $\alpha_t, \beta_t \in \mathbb{R}$ are the unknown parameters of the demand model and ϵ_t are the unobserved demand shocks (noise). By contrast, in this work we consider different products, each characterized by a high-dimensional feature vector. Further, the seller only receives a binary feedback (sale/no sale) of the customer's behavior at each step, rather than observing the customer's valuation.

1.1 Organization of paper and our main contributions

The remainder of this paper is structured as follows. In Section 2, we formally define the model and formulate the problem. Technical assumptions and the notion of regret will be discussed in this section. We next propose a pricing policy based on projected stochastic gradient descent (PSGD) applied to the log-likelihood function. At each time period t, it returns an estimate $\widehat{\theta}_t$. The price p_t is then set to the optimal price as if $\widehat{\theta}_t$ was the actual parameter θ_t. We next analyze the regret of our PSGD algorithm. Let $\delta_t = \|\theta_{t+1} - \theta_t\|$ be the variation in model parameters at time period t. In Section 3.1, we consider the setting where the product feature vectors x_t are chosen antagonistically by nature and show that the regret of PSGD algorithm is of order $O(\sqrt{T} + \sum_{t=1}^{T}\sqrt{t}\delta_t)$. Interestingly, this bound is independent of the dimension d, which is a desirable property of our policy for high-dimensional applications. We next, in Section 4, consider a stochastic features model, where the feature vectors x_t are drawn independently from an unknown distribution (cf. Assumption 6). Under this setting, we show that the regret of PSGD is of order $O(d^2\log T + \sum_{t=1}^{T} t\delta_t/d)$. Note that setting $\delta_t = 0$ corresponds to model (1) and our PSGD pricing obtains a logarithmic regret in T. Section 7 is devoted to the proof of main theorems and the main lemmas are proved in Section 8. Finally, proof of several technical steps are deferred to Appendices.

1.2 Related literature

Our works is at the intersection of dynamic pricing, online optimization and high-dimensional statistics. In the following, we briefly discuss the work most related to ours from these contexts.

Feature-based dynamic pricing. Recent papers on dynamic pricing consider models with features/covariates, motivated in part by new advances in big data technology that allow firms to collect large amount of fine-grained information. In the introduction, we discussed the work (Amin et al., 2014; Javanmard and Nazerzadeh, 2016; Cohen et al., 2016) which are closely related to our setting. Another recent work on feature-based dynamic pricing is (Qiang and Bayati, 2016). In this work, authors consider a model where the seller observes the demand entirely, rather than a binary feedback as in our setting. A greedy iterative least squares (GILS) algorithm is proposed that at each time period estimates the demand as a linear function of price by applying least squares to the set of prior prices and realized demands. The work underscores the role of feature-based approaches and show that they create enough price dispersion to achieve a regret of $O(\log(T))$. This is closely related to the work of (den Boer and Zwart, 2013) and (Keskin and Zeevi, 2014) in dynamic pricing (without demand covariates) that demonstrate the GILS is suboptimal and propose methods to integrate forced price-dispersion with GILS to achieve optimal regret.

Online optimization. This field offers a variety of tools for sequential prediction, where an agent measures its predictive performance according to a series of convex functions. Specifically, there is a sequence of a priori unknown reward functions $f_1, f_2, f_3, \dots$ and an agent must make a sequence of decisions: at each time period t, he selects a point z_t and a loss $f_t(z_t)$ is incurred. Note that the function f_t is not known to agent at step t, but he has access to all previous functions $f_1, \dots, f_{t-1}$. First order methods, like online gradient descent (OGD) or online mirror descent (OMD) only use the gradient of previous function at the selected points, i.e., $\partial f_t(z_t)$. The notion of regret here is defined by comparing the agent with the best fixed comparator (Shalev-Shwartz, 2011).

(Hall and Willett, 2015) proposed dynamic mirror descent that is capable of adapting adapts to a possibly non-stationary environment. In contrast to OMD (Beck and Teboulle, 2003; Shalev-Shwartz, 2011), the notion of regret is defined more generally with respect to the best comparator "sequence".

It is worth noting that the general framework of online learning does not directly apply to our problem. To see this, we define the the loss f_t to be the negative of the revenue obtained

in time period t, i.e., $f_t = -p_t \mathbb{I}(p_t \geq v_t)$. Then (i) the loss functions are not convex; (ii) the (first order information) of previous loss functions depend on the corresponding valuations $v_1, \ldots, v_{t-1}$ which are never revealed to the seller. That said, we borrow some of the techniques from online optimization in proving our results. (See proof of Lemma 3.)

High-dimensional statistics. Among the work in this area, perhaps the most related one to our setting is the problem of 1-bit compressed sensing (Plan and Vershynin, 2013a,b; Ai et al., 2014; Bhaskar and Javanmard, 2015). In this problem, a set of linear measurements are taken from an unknown vector and the goal is to recover this vector having access to the sign of these measurements (1-bit information). This is related to the dynamic pricing problem on model (1), as the seller observes 1-bit feedback (sale/no sale from previous time periods). However, there are a few important differences between these two problem that are worth noting: 1) In dynamic pricing, the crux of the matter is the decisions (prices) made by the firm. Of course this task entails learning the model parameters and therefore the firm gets into the realm of exploration (learning) and exploitation (earning revenue). By contrast, 1-bit compressed sensing is only a learning task; 2) In dynamic pricing, the prices are set based on the previous (sale/no sale) feedbacks. Therefore, the feedbacks are inherently correlated and this makes the learning task challenging. However, in 1-bit compressed sensing it is assumed that the measurements (and therefore the observed signs) are independent; 3) The majority of work on 1-bit compressed sensing consider an offline setting, while in the dynamic pricing, decision are made in an online manner.

2. Model

We consider a pricing problem faced by a firm that sells products in a sequential manner. At each time period $t = 1, 2, \cdots, T$ the firm has a product to sell and the product is represented by an *observable* vector of features (covariates) $x_t \in \mathcal{X} \subseteq \mathbb{R}^d$. The length of the time horizon, denoted by T, is *unknown* the to the firm and the set $\mathcal{X}$ is bounded.

The product at time t has a market value $v_t = v_t(x_t)$, depending on both t and x_t, which is *unobservable*. At each period t, the firm (seller) posts a price p_t. If $p_t \leq v_t$, a sale occurs, and the firm collects revenue p_t. If the price is set higher than the market value, $p_t > v_t$, no sale occurs and no revenue is generated. The goal of the firm is to design a pricing policy that maximizes the collected revenue.

We assume that the market value of a product is a linear function of its covariates, namely

$$v_t(x_t) = \langle \theta_t, x_t \rangle + z_t \,. \tag{3}$$

Here, θ_t and x_t are d-dimensional and $\{z_t\}_{t\geq 1}$ are idiosyncratic shocks, referred to as noise, which are drawn independently and identically from a zero-mean distribution over $\mathbb{R}$. We denote its cumulative distribution function by F, and the corresponding density by $f(x) = F'(x)$. Note that the noise can account for the features that are not measured. We refer to (Keskin and Zeevi, 2014; den Boer and Zwart, 2014; Qiang and Bayati, 2016) for a similar notion of demand shocks.

The sequence of parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots)$ is *unknown* to the firm and it may vary across time. This paper focuses on arbitrary sequences $\boldsymbol{\theta}$ and propose an efficient algorithm whose regret scale gracefully in time and the temporal variability in the sequences of θ_t.

The regret is measured with respect to the clairvoyant policy that knows the sequence $\boldsymbol{\theta}$ in advance. We will formally define the regret in Section 2.2.

Let y_t be the response variable that indicates whether a sale has occurred at period t:

$$y_t = \begin{cases} +1 & \text{if } v_t \geq p_t \,, \\ -1 & \text{if } v_t < p_t \,. \end{cases} \tag{4}$$

Note that the above model for y_t can be represented as the following probabilistic model:

$$y_t = \begin{cases} +1 & \text{with probability } \; 1 - F\left(p_t - \langle \theta_t, x_t \rangle\right) \\ -1 & \text{with probability } \; F\left(p_t - \langle \theta_t, x_t \rangle\right) \end{cases} \tag{5}$$

2.1 Technical assumptions and notations

For a vector v, we write $\|v\|_p$ for the standard ℓ_p norm of a vector v, i.e., $\|v\|_p = (\sum_i |v_i|^p)^{1/p}$. Whenever the subscript p is not mentioned it is deemed as the ℓ_2 norm. For a matrix A, $\|A\|$ denotes its ℓ_2 operator norm. For two vectors a, b, we use the notation $\langle a, b \rangle$ to refer to their inner product.

To simplify the presentation, we assume that $\|x_t\| \leq 1$, for all $x_t \in \mathcal{X}$, and $\|\theta_t\| \leq W$ for a known constant W. We denote by Θ the d-dimensional ℓ_2 ball of radius W (In fact, we can take Θ to be any convex set that contains parameters θ_t. The size of Θ effects our regret bounds up to a constant factor.)

We also make the following assumption on the distribution of noise F.

Assumption 1 *The function $F(v)$ is strictly increasing. Further, $F(v)$ and $1 - F(v)$ are log-concave in v.*

Log-concavity is a widely-used assumption in the economics literature (Bagnoli and Bergstrom, 2005). Note that if the density f is symmetric and the distribution F is log-concave, then $1 - F$ is also log-concave. Assumption 1 is satisfied by several common probability distributions including normal, uniform, Laplace, exponential, and logistic. Note that the cumulative distribution function of all log-concave densities is also log-concave (Boyd and Vandenberghe, 2004).

We use the standard big-O notation. In particular $f(n) = O(g(n))$ if there exists a constant $C > 0$ such that $|f(n)| \leq Cg(n)$ for all n large enough. We also use $\mathbb{R}_{\geq 0}$ to refer to the set of non-negative real-valued numbers.

2.2 Benchmark policy and regret minimization

For a pricing policy, we measures its performance via the notion of regret, which is the expected revenue loss compared to an oracle that knows the sequence of model parameters in advance (but not the realizations of $\{z_t\}_{t\geq 1}$).We first characterize this benchmark policy.

Using Eq. (3), the expected revenue from a posted price p is equal to $p \times \mathbb{P}(v_t \geq p) = p(1 - F(p - \theta_t \cdot x_t))$. First order condition for the optimal price $p^*(x_t, \theta_t)$ reads

$$p^*(x_t, \theta_t) = \frac{1 - F\left(p^*(x_t, \theta_t) - \langle \theta_t, x_t \rangle\right)}{f\left(p^*(x_t, \theta_t) - \langle \theta_t, x_t \rangle\right)} \,. \tag{6}$$

To lighten the notation, we drop the arguments x_t, θ_t and denote by p_t^* the optimal price at time t.

We next recall the *virtual valuation* function, commonly used in mechanism design (Myerson, 1981):

$$\varphi(v) \equiv v - \frac{1 - F(v)}{f(v)}\,.$$

Writing Eq. (6) in terms of function φ, we get

$$\langle \theta_t, x_t \rangle + \varphi\left(p_t^* - \langle \theta_t, x_t \rangle\right) = 0\,.$$

In order to solve for p_t^*, we define the pricing function g as follows:

$$g(v) \equiv v + \varphi^{-1}(-v)\,. \tag{7}$$

By Assumption 1, φ is injective and hence g is well-defined. Further, it is easy to verify that g is non-negative. Using the definition of g and rearranging the terms we obtain

$$p_t^* = g(\langle \theta_t, x_t \rangle)\,. \tag{8}$$

The performance metric we use in this paper is the worst-case regret with respect to a clairvoyant policy that knows the sequence $\boldsymbol{\theta}$ in advance. Formally, for a policy π to be the seller's policy that sets price p_t at period t, the worst-case regret is defined over T periods is defined as:

$$\mathsf{Regret}^{\pi}(T) \equiv \sup\left\{\Delta^{\pi}_{\boldsymbol{\theta},\boldsymbol{x}} : \theta_t \in \Theta,\, x_t \in \mathcal{X}\right\}\,, \tag{9}$$

where for $T \geq 1$, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_T)$ and $\boldsymbol{x} = (x_1, x_2, \ldots, x_T)$,

$$\Delta^{\pi}_{\boldsymbol{\theta},\boldsymbol{x}}(T) = \mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}}\left[\sum_{t=1}^{T}\left(p_t^*\mathbb{I}(v_t \geq p_t^*) - p_t\mathbb{I}(v_t \geq p_t)\right)\right]\,. \tag{10}$$

Here the expectation $\mathbb{E}_{\boldsymbol{\theta},\boldsymbol{x}}$ is with respect to the distributions of idiosyncratic noise, z_t. Note that v_t, p_t, and p_t^* depend on $\boldsymbol{\theta}$ and $\boldsymbol{x}$.

3. Pricing policy

Our dynamic pricing policy consists of a projected gradient descent algorithm to predict parameters $\widehat{\theta}_t$. With each new product, it computes the negative gradient of the loss and shirts its prediction in that direction. The result is projected onto set Θ to produce the next prediction. The policy then sets the prices as $p_t = g(\langle x_t, \widehat{\theta}_t \rangle)$. Note that by Eq. (7), p_t is the optimal price if $\widehat{\theta}_t$ was the true parameter θ_t. Also, by log-concavity assumption on F and $1 - F$, the function $\ell_t(\theta)$ is convex.

In projected gradient descent, the sequence of step sizes $\{\eta_t\}_{t\geq 1}$ is an arbitrary sequence of non-increasing values. In Sections 3.1 and 4, we analyze the regret of our pricing policy and provide guidelines for choosing step sizes.

PSGD (Projected stochastic gradient descent) pricing policy

Input: **(at time** 0**)** function g, set Θ,
Input: **(arrives over time)** covariate vectors $\{x_t\}_{t\in\mathbb{N}}$
Output: prices $\{p_t\}_{t\in\mathbb{N}}$

1: $p_1 \leftarrow 0$ and initialize $\widehat{\theta}_1 \in \Theta$
2: **for** $t = 1, 2, 3, \ldots$ **do**
3: Set $\widehat{\theta}_{t+1}$ according to the following rule:

$$\widehat{\theta}_{t+1} = \Pi_{\Theta}(\widehat{\theta}_t - \eta_t \nabla \ell_t(\widehat{\theta}_t)) \tag{11}$$

with

$$\ell_t(\theta) = -\mathbb{I}(y_t = 1)\log(1 - F(p_t - \langle x_t, \theta\rangle)) - \mathbb{I}(y_t = -1)\log(F(p_t - \langle x_t, \theta\rangle)) \tag{12}$$

4: Set price p_{t+1} as

$$p_{t+1} \leftarrow g(\langle x_{t+1}, \widehat{\theta}_{t+1}\rangle) \tag{13}$$

3.1 Regret analysis

We first define a few useful quantities that appear in our regret bounds. Define

$$M \equiv W + \varphi^{-1}(0)\,, \tag{14}$$

$$u_M \equiv \sup_{|x|\leq M}\left\{\max\left\{-\frac{\mathrm{d}}{\mathrm{d}x}\log F(x), -\frac{\mathrm{d}}{\mathrm{d}x}\log(1 - F(x))\right\}\right\}\,, \tag{15}$$

$$\ell_M \equiv \inf_{|x|\leq M}\left\{\min\left\{-\frac{\mathrm{d}^2}{\mathrm{d}x^2}\log F(x), -\frac{\mathrm{d}^2}{\mathrm{d}x^2}\log(1 - F(x))\right\}\right\}\,, \tag{16}$$

where the derivatives are with respect to x. We note that M is an upper-bound on the maximum price offered and also, by the log-concavity property of F and $1 - F$, we have

$$u_M = \max\left\{-\frac{\mathrm{d}}{\mathrm{d}x}\log F(-M), -\frac{\mathrm{d}}{\mathrm{d}x}\log(1 - F(M))\right\}.$$

Further, by log-concavity property of F and $1 - F$, we have $\ell_M > 0$.

We also let $B = \max_v f(v)$ and $B' = \max_v f'(v)$, respectively denote the maximum value of the density function f and the its derivative f'.

The following theorem bounds the regret of our PSGD policy.

Theorem 2 *Consider model* (3) *for the product market values and let Assumption 1 hold. Set* $M = 2W + \varphi^{-1}(0)$*, with* φ *being the virtual valuation function w.r.t distribution* F*. Then, the regret of PSGD pricing policy using a non-increasing sequence of step sizes* $\{\eta_t\}_{t\geq 1}$ *is bounded as follows:*

$$\mathsf{Regret}(T) \leq \frac{2(2B + MB')}{\ell_M}\max\left\{\frac{16}{\ell_M}\log T,\, \frac{2W^2}{\eta_{T+1}} + \frac{u_M^2}{2}\sum_{t=1}^{T}\eta_t + 2W\sum_{t=1}^{T}\frac{\delta_t}{\eta_t}\right\} + \frac{M}{T}\,, \tag{17}$$

where $\delta_t \equiv \|\theta_{t+1} - \theta_t\|$.

In particular, if $\eta_t \propto \frac{1}{\sqrt{t}}$, then there exists a constant $C = C(B, M, W, \ell_M, u_M) > 0$, independent of T, such that

$$\mathsf{Regret}(T) \leq C\Big(\sqrt{T} + \sum_{t=1}^{T} \sqrt{t}\delta_t\Big)\,. \tag{18}$$

At the core of our regret analysis (proof of Theorem 2) is the following Lemma that provides a prediction error bound for the customer's valuations.

Lemma 3 *Consider model* (3) *for the product market values and let Assumption 1 hold. Set* $M = 2W + \varphi^{-1}(0)$*, with* φ *being the virtual valuation function w.r.t distribution* F*. Let* $\{\widehat{\theta}_t\}_{t\geq 1}$ *be generated by PSGC pricing policy, using a non-increasing positive series* $\eta_{t+1} \leq \eta_t$*. Then, with probability at least* $1 - \frac{1}{T^2}$ *the following holds true:*

$$\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 \leq \frac{4}{\ell_M}\max\Bigg\{\frac{16}{\ell_M}\log T, \frac{2W^2}{\eta_1} + \sum_{t=1}^{T}\Big(\frac{1}{2\eta_{t+1}} - \frac{1}{2\eta_t}\Big)\|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2 + \frac{u_M^2}{2}\sum_{t=1}^{T}\eta_t + 2W\sum_{t=1}^{T}\frac{\delta_t}{\eta_t}\Bigg\}\,, \tag{19}$$

where u_M, ℓ_M *are given by Equations* (15), (16)*, respectively.*

Lemma 3 is presented in a form that can also be used in proving our next results under the stochastic features model. For proving Theorem 2, we simplify bound (19) as follows. Given that $\theta_{t+1}, \widehat{\theta}_{t+1} \in \Theta$, we have $\|\theta_{t+1} - \widehat{\theta}_{t+1}\| \leq 2W$. Using the non-increasing property of sequence η_t, we write

$$\frac{2W^2}{\eta_1} + \sum_{t=1}^{T}\left(\frac{1}{2\eta_{t+1}} - \frac{1}{2\eta_t}\right)\|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2 \leq \frac{2W^2}{\eta_1} + \sum_{t=1}^{T}\left(\frac{2W^2}{\eta_{t+1}} - \frac{2W^2}{\eta_t}\right) \leq \frac{2W^2}{\eta_{T+1}}\,.$$

Therefore, bound (19) simplifies to:

$$\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 \leq \frac{4}{\ell_M}\max\Big\{\frac{16}{\ell_M}\log T, \frac{2W^2}{\eta_{T+1}} + \frac{u_M^2}{2}\sum_{t=1}^{T}\eta_t + 2W\sum_{t=1}^{T}\frac{\delta_t}{\eta_t}\Big\}, \tag{20}$$

The regret bound (17) is derived by relating regret at each time period to the prediction error at that time. We refer to Section 7 for the proof of Theorem 2.

Remark 4 *The regret bound* (17) *does not depend on the dimension* d*, which makes our pricing policy desirable for high-dimensional applications. Also, note that the temporal variation* δ_t *appears in our bound with coefficient* $\sqrt{t}$*. Therefore, variations at later times are more impactful on the regret of PSGD pricing policy. This is expected because at later times, the pricing policy is more relied on the accumulated information about the valuation model and an abrupt change in the model parameters can make this information worthless. On the other side, temporal changes at the beginning steps are not that effective since the policy is still experimenting different prices to learn the customer's behavior.*

Remark 5 *While the regret bound is dimension-free, the computational complexity of PSGD pricing policy scales with dimension* d*. Specifically, the complexity of each step is* $O(d)$*. To see this, we note that the gradient* $\nabla\ell_t(\theta)$ *can be computed in* $O(d)$ *by Equations* (70) *and* (71)*. Projection onto set* Θ *(*ℓ_2 *projection) is also* $O(d)$*.*

4. Stochastic features model

In Theorem 2, we showed that our PSGD pricing policy achieves regret of order $O(\sqrt{T} + \sum_{t=1}^{T}\sqrt{t}\delta_t)$. Let us point out that in Theorem 2 the arrivals (feature vectors x_t) are modeled as adversarial. In this section, we assume that features x_t are independent and identically distributed according to a probability distribution on $\mathbb{R}^d$. Under such stochastic model, we show that the regret of PSGD pricing scales at most of order $O(d^2\log T + \sum_{t=1}^{T} t\delta_t/d)$.

We proceed by formally defining the stochastic features model.

Assumption 6 *(*Stochastic features model*). Feature vectors* x_t *are generated independently according to a probability distribution* $\mathbb{P}_{\boldsymbol{x}}$*, with a bounded support in* $\mathbb{R}^d$*. We denote by* Σ *the covariance matrix of distribution* $\mathbb{P}_{\boldsymbol{x}}$ *and assume that* Σ *has bounded eigenvalues. Specifically, there exist constants* $C_{\min}$ *and* $C_{\max}$ *such that for every eigenvalue* σ *of* Σ*, we have* $0 < \frac{1}{d}C_{\min} \leq \sigma < \frac{1}{d}C_{\max}$*.*

Without loss of generality and to simplify the presentation, we assume that $\mathbb{P}_{\boldsymbol{x}}$ is supported on the unit ℓ_2 ball in $\mathbb{R}^d$. The rationale behind the above assumption on the scaling of eigenvalues is that $\mathrm{Trace}(\Sigma) = \mathbb{E}(\|x_t\|^2) \leq 1$. Therefore, the assumption above on the eigenvalues of Σ states that all the eigenvalues are of the same order.

Under the stochastic features model, we define the notion of worst-case regret as follows. For a policy π be the seller's policy that sets price p_t at period t, the T-period regret is defined as:

$$\mathsf{Regret}^{\pi}(T) \equiv \sup\left\{\Delta^{\pi}_{\boldsymbol{\theta},\mathbb{P}_{\boldsymbol{x}}} : \theta_t \in \Theta, \mathbb{P}_{\boldsymbol{x}} \in Q\right\}\,, \tag{21}$$

where Q denotes the set of probability distribution supported on ℓ_2 unit ball satisfying Assumption 6 (bounded eigenvalues). Further, for $T \geq 1$, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_T)$ and probability measure $\mathbb{P}_{\boldsymbol{x}}$, we define

$$\Delta^{\pi}_{\boldsymbol{\theta},\mathbb{P}_{\boldsymbol{x}}}(T) = \mathbb{E}_{\boldsymbol{\theta},\mathbb{P}_{\boldsymbol{x}}}\left[\sum_{t=1}^{T}\Big(p_t^*\mathbb{I}(v_t \geq p_t^*) - p_t\mathbb{I}(v_t \geq p_t)\Big)\right]\,. \tag{22}$$

where the expectation is with respect to the distributions of idiosyncratic noise, z_t, and $\mathbb{P}_{\boldsymbol{x}}$, the distribution of feature vectors. Note the subtle difference with definition (9), in that the worst case is computed over Q rather than $\mathcal{X}$.

We propose a similar PSGD pricing policy for this setting, with a specific choice of the step sizes. Ideally, we want to set $\eta_t = 6/(\ell_M C t)$, where C is an arbitrary fixed constant such that $0 < C < \sigma_{\min}$, with $\sigma_{\min}$ being the minimum eigenvalue of population covariance Σ. Of course, Σ is unknown and therefore we proceed as follows. We let $Q_t = (1/t)\sum_{\ell=1}^{t} x_\ell x_\ell^{\mathsf{T}}$ be the empirical covariance based on the first t features. Denote by σ_t the minimum eigenvalue of Q_t. We then use the sequence σ_t, and set the step size η_t as

$$\eta_t = \frac{1}{\lambda_t \cdot t}\,, \qquad \lambda_t = \frac{\ell_M}{6}\left\{\frac{1}{t}\Big(1 + \sum_{\ell=1}^{t}\sigma_\ell\Big)\right\}\,.$$

PSGD pricing policy for stochastic features model

Input: (at time 0) function g, set Θ,
Input: (arrives over time) covariate vectors $\{x_t\}_{t\in\mathbb{N}}$
Output: prices $\{p_t\}_{t\in\mathbb{N}}$

1: $p_1 \leftarrow 0$ and initialize $\widehat{\theta}_1 \in \Theta$
2: $Q_1 \leftarrow x_1 x_1^{\mathsf{T}}$
3: **for** $t = 1, 2, 3, \ldots$ **do**
4: Define σ_t as the minimum eigenvalue of Q_t.
5: Set

$$\lambda_t = \frac{\ell_M}{6t}(1 + \sum_{\ell=1}^{t} \sigma_\ell)\,. \tag{23}$$

6: Set

$$\eta_t = \frac{1}{\lambda_t \cdot t} \tag{24}$$

7: Set $\widehat{\theta}_{t+1}$ according to the following rule:

$$\widehat{\theta}_{t+1} = \Pi_\Theta(\widehat{\theta}_t - \eta_t \nabla \ell_t(\widehat{\theta}_t)) \tag{25}$$

with

$$\ell_t(\theta) = -\mathbb{I}(y_t = 1)\log(1 - F(p_t - \langle x_t, \theta\rangle)) - \mathbb{I}(y_t = -1)\log(F(p_t - \langle x_t, \theta\rangle)) \tag{26}$$

8: $Q_{t+1} \leftarrow (\frac{t}{t+1})Q_t + (\frac{1}{t+1})x_{t+1}x_{t+1}^{\mathsf{T}}$
9: Set price p_{t+1} as

$$p_{t+1} \leftarrow g(\langle x_{t+1}, \widehat{\theta}_{t+1}\rangle) \tag{27}$$

Description of the PSGD pricing policy is given in Table above.

4.1 Logarithmic regret bound

The following theorem bounds the regret of our dynamics pricing policy.

Theorem 7 *Consider model* (3) *for the product market values and suppose Assumption 1 holds. Let* $M = 2W + \varphi^{-1}(0)$*, with* φ *being the virtual valuation function w.r.t distribution* F*. Under the stochastic features model (Assumption 6), the regret of PSGD pricing policy is bounded as follows:*

$$\mathsf{Regret}(T) \le C_1 d^2 \log T + C_2 \sum_{t=1}^{T} \frac{t}{d}\delta_t\,, \tag{28}$$

where $\delta_t \equiv \|\theta_{t+1} - \theta_t\|$ *and* C_1, C_2 *are constants that depend on* $C_{\max}, C_{\min}, u_M, \ell_M, M, B, W$ *but are independent of dimension* d*.*

Proof of Theorem 7 relies on the following lemma that is analogous to Lemma 3 and establishes a prediction error bound for the customer's valuations.

Lemma 8 *Consider model* (3) *for the product market values and the stochastic features model (Assumption 6). Suppose that Assumption 1 holds and set* $M = 2W + \varphi^{-1}(0)$*, with* φ *being the virtual valuation function w.r.t distribution* F*. Let* $\{\widehat{\theta}_t\}_{t\ge 1}$ *be generated by PSGD pricing policy. Then,*

$$C_{\min}\sum_{t=1}^{T}\mathbb{E}(\|\theta_t - \widehat{\theta}_t\|^2) \le \left[\frac{128}{\ell_M^2} + \frac{24u_M^2}{\ell_M^2}\left(\tilde{c} + \frac{4}{C_{\min}d}\right)\right]\cdot d^3 \log T + 8W^2 d\left(\frac{1}{T} + \frac{12}{\ell_M^2} + \frac{1}{c_2 d}\right) + 4W\sum_{t=1}^{T} t\delta_t\,.$$

Here $\sigma_{\min}$ *denotes the minimum eigenvalue of covariance* Σ*. (See Assumption 6.)*

4.2 A lower bound on regret

In this section, we provide a theoretical lower bound on the minimum achievable regret of any pricing policy under the stochastic features model. Prior to that, we need to adopt a few notations.

For a given time horizon T and a sequence of valuations parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_T)$, let

$$V_{\boldsymbol{\theta}}(T) \equiv \sum_{t=1}^{T} t\|\theta_{t+1} - \theta_t\|\,. \tag{29}$$

We also define, for $\nu \in [1/2, 2]$,

$$\mathcal{V}(T, B, \nu) \equiv \{\boldsymbol{\theta} :\ \theta_t \in \Theta,\, V_{\boldsymbol{\theta}}(T) \le BdT^\nu\}\,. \tag{30}$$

By assuming $\boldsymbol{\theta} \in \mathcal{V}(T, B, \nu)$ for all T, we are assuming that nature has a finite temporal variation budget to use in changing the valuation parameters throughout the time horizon. Of course, different variation metrics can be considered such as total variation $\sum_{t=1}^{T}\delta_t$ or the maximum temporal variation $\sup_{1\le t\le T}\delta_t$ and the performance of a pricing policy can be studied under different variation budget constraints. The specific choice of (29) is putting higher weights at later variations in the sequence $\boldsymbol{\theta}$ and is reasonable for applications where one expects the buyer's preferences (valuation parameters) become stable over time. Note that designing favorable pricing policy for applications with gradual changes in buyer's preferences is more challenging than that for environments with bursty changes. This might look counterintuitive at first glance because at any time, the accumulated information about valuations can become useless by an abrupt change in the valuation model. However, as noticed and analyzed in (Keskin and Zeevi, 2016), this is not that case because, intuitively, gradual changes can be undetectable and lead to significant revenue loss, while for bursty

changes, the policy can be designed in a way to detect the changes and reset its estimate of the valuation model after each change to avoid large estimation error and revenue loss. For a pricing policy π, consider the T-period regret, defined as

$$\mathsf{Regret}^{\pi}(T, B, \nu) \equiv \max \left\{ \Delta^{\pi}_{\boldsymbol{\theta}, \mathbb{P}_{\boldsymbol{x}}}(T) : \boldsymbol{\theta} \in \mathcal{V}(T, B, \nu),\, \mathbb{P}_{\boldsymbol{x}} \in Q \right\} \tag{31}$$

where we recall that

$$\Delta^{\pi}_{\boldsymbol{\theta}, \mathbb{P}_{\boldsymbol{x}}}(T) \equiv \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{\theta}, \mathbb{P}_{\boldsymbol{x}}} \Big(p_t^* \mathbb{I}(v_t \geq p_t^*) - p_t \mathbb{I}(v_t \geq p_t) \Big) \,. \tag{32}$$

Note that this is the same regret notion defined in (21), where we just make the variation budget constraint explicit in the notation.

Rephrasing the statement of Theorem 7, for PSGD pricing policy we have $\mathsf{Regret}^{\pi}(T, B, \nu) \leq C_1 d^2 \log T + C_2 B T^{\nu}$. We next provide a lower bound on the regret of any pricing policy. Indeed this lower bound applies to a powerful clairvoyant who fully observes the market values after the price is either accepted or rejected.

Theorem 9 *Consider linear model* (3) *where the market values $v_t(x_t)$, $1 \leq t \leq T$, are fully observed. We further assume that market value noises are generated as $z_t \sim \mathsf{N}(0, \sigma^2)$. There exists a constant c, depending on σ, $C_{\max}$, such that*

$$\mathsf{Regret}^{\pi}(T, B, \nu) \geq c \min \Big(\big(B^2 d T^{2\nu-1}\big)^{1/3}, T/d \Big) \,,$$

for any pricing policy π and time horizon T.

The high-level intuition behind this result is that the nature can change the valuation parameters in a gradual manner such that the seller should pay a revenue loss in order to detect the changes and learn the new valuation parameter after a change. To be more specific, we divide the time horizon into cycles of length N periods, where N is of order $(T^{4-2\nu}/d)^{1/3}$ and consider a setting where the value of θ_t can change to one of two options θ^0, θ^1, only in the first period of a cycle. We choose the parameter change $\delta = \|\theta^1 - \theta^0\|$ of order $\sqrt{d/N}$ to ensure that (i) no policy can identify the change without incurring a revenue loss of order $N\delta^2/d$ (ii) The variation metric $V_{\boldsymbol{\theta}}(T)$ remains below the allowable limit of BdT^{ν}. Therefore, the total regret over T periods works out at $T\delta^2/d$. In particular, for proving point (i) we quantify the likelihood of valuations under the probability measures corresponding to θ^0 and θ^1, using Kullback-Leibler divergence. We use Pinsker inequality form probability theory and hypothesis testing results from information theory to show that there is a significant probability of not detecting the (potential) change, which consequently yields a revenue loss of order $N\delta^2/d$, over each cycle.

We refer to Section 7.3 for the proof of Theorem 9.

5. Numerical experiments

We numerically study the performance of our PSGD pricing policy on synthetic data. In our experiments, we set $W = 5$ and set $\theta_1 = (W/2)(Z/\|Z\|)$, with $Z \sim \mathsf{N}(0, \mathrm{I}_d)$ a multivariate normal variable. We then generate a sequence of parameters θ_t as follows:

$$\theta_{t+1} = \theta_t + r_t \,,$$

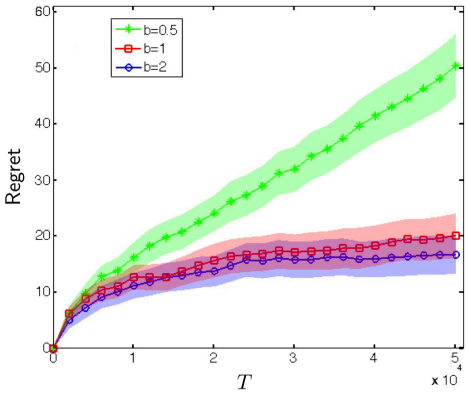


Figure 1: Cumulative regret of PSGD pricing policy for the synthetic data in Section 5. Temporal variations are $\delta_t = t^{-b}$ and the curves are obtained by averaging across 80 trials. Shaded region around each curve is the 95% confidence interval.

where $r_t = t^{-b}(\tilde{Z}/\|\tilde{Z}\|)$, with $\tilde{Z} \sim \mathsf{N}(0, \mathrm{I}_d)$. Note that $\delta_t = \|\theta_{t+1} - \theta_t\| = \|r_t\| = t^{-b}$.

Next, at each time t, product covariates x_t are independently sampled from a Gaussian distribution $\mathsf{N}(0, \mathrm{I}_d)$ and normalized so that $\|x_t\| = 1$. Further, the market shocks are generated as $z_t \sim \mathsf{N}(0, \sigma^2)$, with $\sigma = 1$. We run the PSGD pricing policy for stochastic features model.

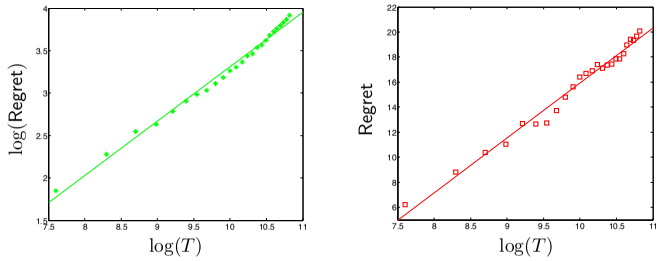
Results. Figure 1 compares the cumulative regret (averaged over 80 trials) of the PSGD policy, for $b = 0.5, 1, 2$, on the aforementioned synthetic data for $T = 50,000$ steps. The shaded region around each curve correspond to the 95% confidence interval across the 80 trials. As expected, increase in b results in larger temporal variations and larger regret.

To better understand the behavior of regret for different values of b, we plotted the regret bounds in various scales in Figure 2. For $b = 0.5$, we have $\mathsf{Regret}(T) \sim T^{2/3}$, and for $b = 1, 2$, we have $\mathsf{Regret} \sim \log(T)$. Comparing with Theorem 7, we see that the empirical regret in case of $b = 0.5$, 1, is smaller than the upper bound given by Equation (28), order-wise. However, it is worth noting that bound given in Theorem 7 applies to any adversarial choice of temporal variations r_t, while in our experiments we generated these terms independently at random.

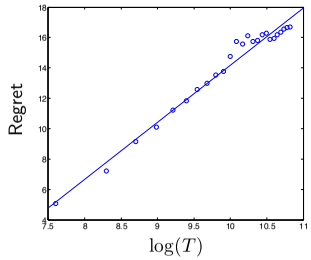
6. Extension to nonlinear model

Throughout the paper, we exclusively focused on linear models for buyer's valuation with varying coefficients. In order to generalize our results to nonlinear models, we consider a setting where the market value of a product with feature vector x_t is given by

$$v_t(x_t) = \psi(\langle x_t, \theta_t \rangle + z_t) \,. \tag{33}$$



(a) $b = 0.5$ (b) $b = 1$



(c) $b = 2$

Figure 2: Cumulative regrets of PSGD for different values of b. For $b = 0.5$, $\mathsf{Regret}(T) \sim T^{2/3}$; for $b = 1, 2$, $\mathsf{Regret}(T) \sim \log(T)$.

This model is often referred to as generalized linear model and captures nonlinear dependencies on features to some extent. We assume that the link function $\psi : \mathbb{R} \mapsto \mathbb{R}$ is a general log-concave function and is strictly increasing.

We next compute the pricing function. Since ψ is strictly increasing, the expected revenue at a price p amounts to $p\left(1 - F\left(\psi^{-1}(p) - \langle x_t, \theta_t\rangle\right)\right)$. First order condition for the optimal price $p_t^*(x_t)$ reads as

$$\psi'(\psi^{-1}(p_t^*)) = \frac{pf\left(\psi^{-1}(p_t^*) - \langle x_t, \theta_t\rangle\right)}{1 - F\left(\psi^{-1}(p_t^*) - \langle x_t, \theta_t\rangle\right)}\,. \tag{34}$$

Define $\lambda(v) = f(v)/(1 - F(v))$ the hazard rate function for distribution F, and let $\tilde{p} = \psi^{-1}(p)$. Writing (34) in terms of λ function, we get

$$\langle x_t, \theta_t\rangle = \tilde{p}_t^* - \lambda^{-1}\left(\frac{\psi'(\tilde{p}_t^*)}{\psi(\tilde{p}_t^*)}\right)\,. \tag{35}$$

For real-valued v, define

$$g_\psi^{-1}(v) \equiv v - \lambda^{-1}\left(\frac{\psi'(v)}{\psi(v)}\right)\,. \tag{36}$$

Note that by log-concavity of $1 - F$, the hazard function λ is increasing. Also, by log-concavity of ψ, the term $\frac{\mathrm{d}}{\mathrm{d}v}\log\psi(v) = \psi'(v)/\psi(v)$ is decreasing. Putting these together, we obtain that $-\lambda^{-1}(\psi'(v)/\psi(v))$ is increasing. Therefore, the right-hand side of (36) is strictly increasing and the function g_ψ is well-defined. Invoking Equation (35), we derive the optimal price as

$$p_t^* = \psi\left(g_\psi(\langle x_t, \theta_t\rangle)\right)\,. \tag{37}$$

As noted before, since ψ is increasing, at each period t, a sale happens if $z_t \geq \psi^{-1}(p_t) - \langle x_t, \theta_t\rangle$. Hence, the log-likelihood function reads as

$$\ell_t(\theta) = -\mathbb{I}(y_t = 1)\log\left(1 - F\left(\psi^{-1}(p_t) - \langle x_t, \theta\rangle\right)\right) - \mathbb{I}(y_t = -1)\log\left(F\left(\psi^{-1}(p_t) - \langle x_t, \theta\rangle\right)\right)\,. \tag{38}$$

In PSGD pricing policy, we run gradient step with this log-likelihood function and then set price p_{t+1} at next step as $p_{t+1} = \psi\left(g_\psi(\langle x_{t+1}, \theta_{t+1}\rangle)\right)$.

The results on the regret of PSGD pricing policy carries over to the generalized linear model as well. The analysis goes along the same lines and is omitted.

7. Proof of main theorems

7.1 Proof of Theorem 2

Lemma 10 *Set $M = 2W + \varphi^{-1}(0)$, and for $\theta \in \Theta$ define $u_t(\theta) = p_t - \langle x_t, \theta\rangle$, where $p_t = g(\langle x_t, \widehat{\theta}_t\rangle)$ is the posted price at time t. Then $|u_t(\theta)| \leq M$ for all $t \geq 1$.*

Define function $h(\,;u)$ from $\mathbb{R}_{\geq 0}$ to $\mathbb{R}_{\geq 0}$ as

$$h(p;u) = p(1 - F(p - u))$$

This is the expected revenue at price p when the noiseless valuation is u, i.e., $\langle x_t, \theta_t\rangle = u$. We let

$$R_t \equiv p_t^* \mathbb{I}(v_t \geq p_t^*) - p_t \mathbb{I}(v_t \geq p_t) \tag{39}$$

be the regret incurred at time t, and define $\mathcal{F}_t$ as the history up to time t (Formally, $\mathcal{F}_t$ is the σ-algebra generated by market noise $\{z_\ell\}_{\ell=1}^t$.) Then,

$$\mathbb{E}(R_t|\mathcal{F}_{t-1}) = p_t^* \mathbb{P}(v_t \geq p_t^*) - p_t \mathbb{P}(v_t \geq p_t) = h(p_t^*; \langle x_t, \theta_t\rangle) - h(p_t; \langle x_t, \widehat{\theta}_t\rangle)\,. \tag{40}$$

The optimal price p_t^* is the maximizer of $h(p; \langle x_t, \theta_t\rangle)$ and thus $h'(p_t^*; \langle x_t, \theta_t\rangle) = 0$. By Taylor expansion of function h, there exists a value p between p_t and p_t^*, such that,

$$h(p_t; \langle x_t, \theta_t\rangle) - h(p_t^*; \langle x_t, \theta_t\rangle) = \frac{1}{2} h''(p; \langle x_t, \theta_t\rangle)(p_t - p_t^*)^2\,. \tag{41}$$

We next show that $|h''(p; \langle x_t, \theta_t\rangle)| \leq C$ with $C = 2B + MB'$. Recall that $B = \max_v f(v)$ and $B' = \max_v f'(v)$. To see this, we write

$$|h''(p; \langle x_t, \theta_t\rangle)| = \Big|2f(p - \langle x_t, \theta_t\rangle) + pf'(p - \langle x_t, \theta_t\rangle)\Big| \leq 2B + MB'\,. \tag{42}$$

Putting Equations (39), (41), (42) and using the 1-Lipschitz property of price function g, we conclude:

$$\begin{aligned}\mathbb{E}[R_t|\mathcal{F}_{t-1}] &= h(p_t^*; \langle x_t, \theta_t\rangle) - h(p_t; \langle x_t, \widehat{\theta}_t\rangle) \leq \frac{2B + MB'}{2}(p_t - p_t^*)^2 \\ &= \frac{2B + MB'}{2}\Big(g(\langle x_t, \widehat{\theta}_t\rangle) - g(\langle x_t, \theta_t\rangle)\Big)^2 \leq \frac{2B + MB'}{2}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 \end{aligned} \tag{43}$$

To ease the presentation, define the shorthand

$$A(T) \equiv \frac{4}{\ell_M} \max\left\{\frac{16}{\ell_M}\log T,\, \frac{2W^2}{\eta_{T+1}} + \frac{u_M^2}{2}\sum_{t=1}^T \eta_t + 2W\sum_{t=1}^T \frac{\delta_t}{\eta_t}\right\}.$$

We further let $\mathcal{G}$ be the probabilistic event that $\sum_{t=1}^T \langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 \leq A(T)$. Employing Lemma 3 and using the fact that $\|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2 \leq 4W^2$, we obtain that $\mathbb{P}(\mathcal{G}) \geq 1 - \frac{1}{T^2}$.

We continue by bounding $E(R_t)$ as follows:

$$\begin{aligned}\mathbb{E}[R_t] &= \mathbb{E}[\mathbb{E}[R_t|\mathcal{F}_{t-1}]] = \mathbb{E}\Big[\mathbb{E}[R_t|\mathcal{F}_{t-1}]\cdot\Big(\mathbb{I}(\mathcal{G}) + \mathbb{I}(\mathcal{G}^c)\Big)\Big] \\ &= \frac{2B + MB'}{2}\mathbb{E}\Big[\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 \cdot \mathbb{I}(\mathcal{G})\Big] + M\mathbb{P}(\mathcal{G}^c)\,.\end{aligned}$$

Consequently,

$$\mathsf{Regret}(T) \leq \sum_{t=1}^T \mathbb{E}[R_t] \leq \frac{2B + MB'}{2}\mathbb{E}\Big[\sum_{t=1}^T \langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 \cdot \mathbb{I}(\mathcal{G})\Big] + MT\,\mathbb{P}(\mathcal{G}^c) \leq \frac{2B + MB'}{2}A(T) + \frac{M}{T}\,.$$

The proof is complete.

7.2 Proof of Theorem 7

Proof of Theorem 7 follows along the same lines as proof of Theorem 2. Let $\tilde{\mathcal{F}}_t$ be the σ-algebra generated by market noises $\{z_\ell\}_{\ell=1}^t$ and feature vectors $\{x_\ell\}_{\ell=1}^t$. Further, let $\mathcal{F}_t$ be the σ-algebra generated by $\tilde{\mathcal{F}}_t \cup \{x_{t+1}\}$. For term R_t defined by (39) and following the chain of inequalities as in (43),

$$\mathbb{E}[R_t|\mathcal{F}_{t-1}] \leq \frac{2B + MB'}{2}\,\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\,. \tag{44}$$

For brevity in notation, let $\bar{B} = (2B + MB')/2$. Since, $\mathcal{F}_t \supseteq \tilde{\mathcal{F}}_t$, by iterated law of iteration,

$$\mathbb{E}(R_t|\tilde{\mathcal{F}}_{t-1}) = \mathbb{E}(\mathbb{E}(R_t|\mathcal{F}_{t-1})|\tilde{\mathcal{F}}_{t-1}) \leq \bar{B}\langle \theta_t - \widehat{\theta}_t, \Sigma(\theta_t - \widehat{\theta}_t)\rangle \leq \frac{1}{d}\bar{B}C_{\max}\|\theta_t - \widehat{\theta}_t\|^2 \tag{45}$$

Applying Lemma 8, we get

$$\begin{aligned}\mathsf{Regret}(T) &\leq \sum_{t=1}^T \mathbb{E}[R_t] \leq \frac{1}{d}\bar{B}C_{\max}\sum_{t=1}^T \mathbb{E}(\|\theta_t - \widehat{\theta}_t\|^2) \\ &\leq \bar{B}\frac{C_{\max}}{C_{\min}}\left[\frac{128}{\ell_M^2} + \frac{24u_M^2}{\ell_M^2}\left(\tilde{c} + \frac{4}{C_{\min}d}\right)\right]\cdot d^2\log T \\ &\quad + 8W^2\bar{B}\frac{C_{\max}}{C_{\min}}\left(\frac{1}{T} + \frac{12}{\ell_M^2} + \frac{1}{c_2 d}\right) + \bar{B}\frac{C_{\max}}{C_{\min}}\Big(\frac{4W}{d}\Big)\sum_{t=1}^T t\delta_t\,.\end{aligned}$$

The result follows by taking

$$\begin{aligned}C_1 &= \bar{B}\frac{C_{\max}}{C_{\min}}\left[8W^2\left(\frac{1}{T} + \frac{12}{\ell_M^2} + \frac{1}{c_2 d}\right) + \frac{128}{\ell_M^2} + \frac{24u_M^2}{\ell_M^2}\left(\tilde{c} + \frac{4}{C_{\min}d}\right)\right]\,, \\ C_2 &= 4W\bar{B}\frac{C_{\max}}{C_{\min}}\,.\end{aligned}$$

7.3 Proof of Theorem 9

The proof methodology is similar to the proof of (Keskin and Zeevi, 2016, Theorem 1).

We first propose a setting for constructing the sequence of valuation parameters $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_T)$. Divide the time horizon into cycles of length $N = \lceil m_0 T^{(4-2\nu)/3}\rceil$, where $m_0 = (\frac{\sigma^2}{C_{\max}B^2 d})^{1/3}$. Consider a setting wherein the noise markets are generated as $z_t \sim \mathsf{N}(0, \sigma^2)$ and the value of θ_t can change only in the first period of a cycle, taking one of the two values $\{\theta^0, \theta^1\}$. Here, $\theta^0, \theta^1 \in \mathbb{R}^d$ are two arbitrary vectors such that $\|\theta^0 - \theta^1\| = \delta$, with $\delta = \min(\sigma\sqrt{d/(C_{\max}N)}, \sqrt{c_2})$. Note that for this sequence of $\boldsymbol{\theta}$, we have

$$V_{\boldsymbol{\theta}}(T) \leq \sum_{k=1}^{\lceil T/N\rceil}(kN)\delta \leq \frac{T^2}{N}\delta \leq BdT^\nu \tag{46}$$

We consider a clairvoyant who fully observes the market values $v_t(x_t)$. Focus on a single cycle and let $\mathbb{P}_0^\pi$ (resp. $\mathbb{P}_1^\pi$) denote the probability distribution of valuations $(v_1, v_2, \ldots, v_N)$

when all the parameters θ_t are equal to θ^0 (resp. θ^1), for $1 \le t \le N$. The KL divergence between $\mathbb{P}_0^\pi$ and $\mathbb{P}_1^\pi$ amounts to

$$D_{\mathrm{KL}}(\mathbb{P}_0^\pi, \mathbb{P}_1^\pi) \equiv \mathbb{E}_0^\pi \log \left(\frac{\prod_{t=1}^N \phi\left(\frac{v_t - \langle x_t, \theta_0 \rangle}{\sigma}\right)}{\prod_{t=1}^N \phi\left(\frac{v_t - \langle x_t, \theta_1 \rangle}{\sigma}\right)} \right) . \tag{47}$$

where $\mathbb{E}_0^\pi$ denotes expectation w.r.t $\mathbb{P}_0^\pi$ and $\phi(s) = 1/(\sqrt{2\pi})e^{-s^2/2}$ is the standard Gaussian density. After simple algebraic manipulation, we obtain

$$\begin{aligned} D_{\mathrm{KL}}(\mathbb{P}_0^\pi, \mathbb{P}_1^\pi) &= -\frac{1}{2\sigma^2}\mathbb{E}_0^\pi \Big\{ \sum_{t=1}^N (2z_t - \langle x_t, \theta^1 - \theta^0 \rangle) \langle x_t, \theta^1 - \theta^0 \rangle \Big\} \\ &= \frac{1}{2\sigma^2} \sum_{t=1}^N \mathbb{E}_0^\pi (\langle x_t, \theta^1 - \theta^0 \rangle^2) \le \frac{1}{2\sigma^2 d} \sum_{t=1}^N C_{\max} \|\theta^1 - \theta^0\|^2 \\ &= \frac{1}{2\sigma^2} C_{\max} \frac{\delta^2 N}{d} . \end{aligned}$$

We next relate the expected regret to the KL divergence between $\mathbb{P}_0^\pi$ and $\mathbb{P}_1^\pi$.

Lemma 11 *Let R_t be the regret incurred at time t, defined as $R_t \equiv p_t^* \mathbb{I}(v_t \ge p_t^*) - p_t \mathbb{I}(v_t \ge p_t)$. Then, there exist constants c_1, c_2 depending on σ, W, and $C_{\min}$, such that*

$$\mathbb{E}(R_t) \ge \frac{c_1}{d} \mathbb{E}\Big\{ \min\Big(\|\widehat{\theta}_t - \theta_t\|_2^2, c_2 \Big) \Big\} . \tag{48}$$

Proof of Lemma 11 goes along the proof of (Javanmard and Nazerzadeh, 2016, Equation (55)) and is omitted.

By applying Lemma 11, we have

$$\Delta_{\boldsymbol{\theta}, \mathbb{P}_{\boldsymbol{x}}}^\pi(N) = \sum_{t=1}^N \mathbb{E}_{\boldsymbol{\theta}}(R_t) \ge \frac{c_1}{d} \sum_{t=1}^N \mathbb{E}\Big\{ \min\Big(\|\widehat{\theta}_t - \theta_t\|_2^2, c_2 \Big) \Big\} . \tag{49}$$

For brevity in notations, for the sequence $\boldsymbol{\theta} = (\theta_1, \dots, \theta_N)$, we define $\mathsf{d}_a(\boldsymbol{\theta}) = c_1 \sum_{t=1}^N \min(\|\theta_t - \theta^a\|_2^2, c_2)$, for $a = 1, 2$. Define two sets J_a, for $a = 1, 2$ as follows:

$$J_a = \Big\{ \boldsymbol{\theta} = (\theta_1, \dots, \theta_N) : \theta_i \in \mathbb{R}^d , \mathsf{d}_a(\boldsymbol{\theta}) < \frac{1}{4} N \delta^2 \Big\} . \tag{50}$$

Then,

$$\begin{aligned} \max\big(\Delta_{0,\mathbb{P}_{\boldsymbol{x}}}^\pi(N), \Delta_{1,\mathbb{P}_{\boldsymbol{x}}}^\pi(N)\big) &\ge \frac{1}{d} \max\Big(\mathbb{E}_0^\pi(\mathsf{d}_0(\boldsymbol{\theta})), \mathbb{E}_1^\pi(\mathsf{d}_1(\boldsymbol{\theta})) \Big) \\ &\ge \frac{N}{4d}\delta^2 \max\Big(\mathbb{P}_0^\pi(\boldsymbol{\theta} \notin J_0), \mathbb{P}_1^\pi(\boldsymbol{\theta} \notin J_1) \Big) \\ &\overset{(a)}{\ge} \frac{N}{4d}\delta^2 \max\Big(\mathbb{P}_0^\pi(\boldsymbol{\theta} \notin J_0), \mathbb{P}_1^\pi(\boldsymbol{\theta} \in J_0) \Big) \\ &\ge \frac{N}{8d}\delta^2 \Big(\mathbb{P}_0^\pi(\boldsymbol{\theta} \notin J_0) + \mathbb{P}_1^\pi(\boldsymbol{\theta} \in J_0) \Big) \\ &\ge \frac{N}{8d}\delta^2 \Big(1 - \mathbb{P}_0^\pi(\boldsymbol{\theta} \in J_0) + \mathbb{P}_1^\pi(\boldsymbol{\theta} \in J_0) \Big) \\ &\ge \frac{N}{8d}\delta^2 \Big(1 - \sqrt{\frac{1}{2} D_{\mathrm{KL}}(\mathbb{P}_0^\pi, \mathbb{P}_1^\pi)} \Big) \quad \text{(By Pinsker inequality)} \\ &\ge \frac{N}{8d}\delta^2 \left(1 - \frac{1}{2\sigma}\delta \sqrt{C_{\max}\frac{N}{d}} \right) \ge \frac{N\delta^2}{16d} . \end{aligned}$$

Here (a) holds because $\boldsymbol{\theta} \in J_0$ implies $\boldsymbol{\theta} \notin J_1$. Otherwise, $\mathsf{d}_0(\boldsymbol{\theta}) < N\delta^2/4$ and $\mathsf{d}_1(\boldsymbol{\theta}) < N\delta^2/4$. Using the inequality $\min(a+b, c) \le \min(a, c) + \min(b, c)$ for $a, b, c \ge 0$, and applying triangle inequality, we get

$$N \min(\|\theta^0 - \theta^1\|^2, c_2) \le 2\mathsf{d}_0(\boldsymbol{\theta}) + 2\mathsf{d}_1(\boldsymbol{\theta}) < N\delta^2 , \tag{51}$$

which is a contradiction because $\delta^2 = \|\theta^0 - \theta^1\|^2 \le c_2$. Therefore, we conclude that

$$\begin{aligned} \mathsf{Regret}^\pi(T, B, \nu) &\ge \Big\lfloor \frac{T}{N} \Big\rfloor \max\big(\Delta_{0,\mathbb{P}_{\boldsymbol{x}}}^\pi(N), \Delta_{1,\mathbb{P}_{\boldsymbol{x}}}^\pi(N)\big) \\ &\ge \frac{T\delta^2}{16d} = \frac{T}{16} \min\Big(\frac{\sigma^2}{C_{\max} N}, \frac{c_2}{d} \Big) \\ &= \frac{1}{16} \min\left\{ \Big(\frac{\sigma^2}{C_{\max}} \Big)^{2/3} (B^2 d T^{2\nu-1})^{1/3}, \frac{c_2 T}{d} \right\} . \end{aligned} \tag{52}$$

The result follows.

8. Proof of main lemmas

8.1 Proof of Lemma 3

We prove Lemma 3 by developing an upper bound and a lower bound for the quantity $\sum_{t=1}^T \ell_t(\widehat{\theta}_t) - \sum_{t=1}^T \ell_t(\theta_t)$. The result follows by combining these two bounds.

Lemma 12 (*Upper bound*) *Suppose $\{\theta_t\}_{t \ge 1}$ is an arbitrary sequence in Θ, and $\|\theta\| \le W$ for all $\theta \in \Theta$. Set $M = 2W + \varphi^{-1}(0)$, with φ being the virtual valuation function w.r.t distribution F. Further, let $\{\widehat{\theta}_t\}_{t \ge 1}$ be generated by PSGD policy using a non-increasing*

positive series $\eta_{t+1} \le \eta_t$. *Then*

$$\sum_{t=1}^{T} \ell_t(\widehat{\theta}_t) - \sum_{t=1}^{T} \ell_t(\theta_t) \le \frac{2W^2}{\eta_1} + \sum_{t=1}^{T} \Big(\frac{1}{2\eta_{t+1}} - \frac{1}{2\eta_t}\Big)\|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2 + \frac{u_M^2}{2}\sum_{t=1}^{T}\eta_t + 2W\sum_{t=1}^{T}\frac{\delta_t}{\eta_t} - \frac{\ell_M}{2}\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\,, \tag{53}$$

where $\delta_t \equiv \|\theta_{t+1} - \theta_t\|$ *and we recall* u_M *from Equation* (15).

The proof of Lemma 12 uses similar ideas to the regret bounds established in (Hall and Willett, 2015), but uses the log-concavity of F and $1-F$ and also definition of u_M and ℓ_M as per Equations (15) and (16) to get a more refined bound including quadratic terms $\langle x_t, \widehat{\theta}_t - \theta_t\rangle^2$. We refer to Appendix B for the proof of Lemma 12.

Our next Lemma provides a probabilistic lower bound on $\sum_{t=1}^{T} \ell_t(\widehat{\theta}_t) - \sum_{t=1}^{T} \ell_t(\theta_t)$.

Lemma 13 (*Lower bound*) *Consider model* (3) *for the product market values and suppose Assumption 1 holds. Let* $\{\widehat{\theta}_t\}_{t\ge 1}$ *be an arbitrary sequence in* Θ. *Then with probability at least* $1 - \frac{1}{T^2}$ *the following holds true*

$$\sum_{t=1}^{T} \ell_t(\widehat{\theta}_t) - \sum_{t=1}^{T} \ell_t(\theta_t) \ge -2\sqrt{\log T}\Big\{\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\Big\}^{1/2}. \tag{54}$$

Proof of Lemma 13 is given in Appendix C. It uses convexity of $\ell_t(\widehat{\theta})$ and an application of a concentration bound on martingale difference sequences.

Combining Equations (53) and (54) we obtain that with probability at least $1 - \frac{1}{T^2}$ the following holds true

$$-2\sqrt{\log T}\Big\{\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\Big\}^{1/2} \le \frac{2W^2}{\eta_1} + \sum_{t=1}^{T} \Big(\frac{1}{2\eta_{t+1}} - \frac{1}{2\eta_t}\Big)\|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2 + \frac{u_M^2}{2}\sum_{t=1}^{T}\eta_t + 2W\sum_{t=1}^{T}\frac{\delta_t}{\eta_t} - \frac{\ell_M}{2}\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 \tag{55}$$

Rearranging the terms, we get

$$\frac{\ell_M}{2}\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 - 2\sqrt{\log T}\Big\{\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\Big\}^{1/2} \le \frac{2W^2}{\eta_1} + \sum_{t=1}^{T} \Big(\frac{1}{2\eta_{t+1}} - \frac{1}{2\eta_t}\Big)\|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2 + \frac{u_M^2}{2}\sum_{t=1}^{T}\eta_t + 2W\sum_{t=1}^{T}\frac{\delta_t}{\eta_t} \tag{56}$$

Define $A \equiv \sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2$ and denote by B the right-hand side of Equation (56).

Writing in terms of A and B, we have

$$A - \frac{4}{\ell_M}\sqrt{A\log T} \le \frac{2B}{\ell_M}\,. \tag{57}$$

We next upper bound A as follows. Consider two cases:

Case 1: Assume that

$$\sqrt{A\log T} \le \frac{\ell_M}{8}A\,.$$

Using this in Equation (57), we get $A \le 4B/\ell_M$.

Case 2: Assume that

$$\sqrt{A\log T} > \frac{\ell_M}{8}A\,.$$

Then, $A < (64/\ell_M^2)\log T$.

Combining the above two cases, we obtain

$$A \le \frac{4}{\ell_M}\max\Big(\frac{16}{\ell_M}\log T, B\Big)\,.$$

Substituting for A and B, we have

$$\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 \le \frac{4}{\ell_M}\max\Big\{\frac{16}{\ell_M}\log T, \frac{2W^2}{\eta_1} + \sum_{t=1}^{T}\Big(\frac{1}{2\eta_{t+1}} - \frac{1}{2\eta_t}\Big)\|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2 + \frac{u_M^2}{2}\sum_{t=1}^{T}\eta_t + 2W\sum_{t=1}^{T}\frac{\delta_t}{\eta_t}\Big\}$$

The proof is complete.

8.2 Proof of Lemma 8

theorem 8.1 *Let* σ_t *denote the minimum eigenvalue of* $Q_t \equiv (1/t)\sum_{\ell=1}^{t} x_\ell x_\ell^{\mathsf{T}}$. *Further, let* $\sigma_{\min}$ *be the minimum eigenvalue of* Σ, *where* Σ *is the population covariance of feature vectors as in Assumption 6. Then, there exist constants* $c_1, c_2 > 0$, *such that*

$$\forall t \ge c_1 d:\ \mathbb{P}\Big(\frac{1}{2}\sigma_{\min} \le \sigma_t \le \frac{3}{2}\sigma_{\min}\Big) \ge 1 - 2e^{-c_2 t/d}\,. \tag{58}$$

Further, $\sigma_t \le 1$, *for all* $t \ge 1$.

Let $\mathcal{F}_t$ be the σ algebra generated by market shocks $\{z_\ell\}_{\ell=1}^{t}$ and features $\{x_\ell\}_{\ell=1}^{t}$. We further define $D_t = \langle x_t, \widehat{\theta}_t - \theta_t\rangle^2 - \|\Sigma^{1/2}(\widehat{\theta}_t - \theta_t)\|^2$. Note that $\widehat{\theta}_t$ is $\mathcal{F}_{t-1}$ measurable and x_t is independent of $\mathcal{F}_{t-1}$, which implies $\mathbb{E}(D_t|\mathcal{F}_{t-1}) = 0$. Hence, $\mathbb{E}(D_t) = 0$ by iterated law of expectation and therefore $\sum_{t=1}^{T}\mathbb{E}(D_t) = 0$. Equivalently,

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle x_t, \widehat{\theta}_t - \theta_t\rangle^2\right] = \sum_{t=1}^{T}\mathbb{E}\left[\|\Sigma^{1/2}(\widehat{\theta}_t - \theta_t)\|^2\right] \ge \sigma_{\min}\,\mathbb{E}\left[\sum_{t=1}^{T}\|\widehat{\theta}_t - \theta_t\|^2\right] \tag{59}$$

Define $\mathcal{G}_T$ the event that bound (19) holds true. Then,

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle x_t,\widehat{\theta}_t-\theta_t\rangle^2\right]=\mathbb{E}\left[\sum_{t=1}^{T}\langle x_t,\widehat{\theta}_t-\theta_t\rangle^2\cdot(\mathbb{I}_{\mathcal{G}}+\mathbb{I}_{\mathcal{G}^c})\right]$$
$$\le\mathbb{E}\left[\sum_{t=1}^{T}\langle x_t,\widehat{\theta}_t-\theta_t\rangle^2\cdot\mathbb{I}_{\mathcal{G}}\right]+4W^2T\,\mathbb{P}(\mathcal{G}^c)$$
$$\le\mathbb{E}\left[\sum_{t=1}^{T}\langle x_t,\widehat{\theta}_t-\theta_t\rangle^2\cdot\mathbb{I}_{\mathcal{G}}\right]+\frac{4W^2}{T}\,. \tag{60}$$

Further, using inequality $\max(a,b)\le|a|+|b|$, we get

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle x_t,\widehat{\theta}_t-\theta_t\rangle^2\cdot\mathbb{I}_{\mathcal{G}}\right]\le\frac{4}{\ell_M}\Big\{\frac{16}{\ell_M}\log T+\frac{12W^2}{\ell_M}+\frac{1}{2}\sum_{t=1}^{T}\mathbb{E}\Big[\Big((t+1)\lambda_{t+1}-t\lambda_t\Big)\cdot\|\theta_{t+1}-\widehat{\theta}_{t+1}\|^2\Big]$$
$$+\frac{u_M^2}{2}\sum_{t=1}^{T}\mathbb{E}\left[\frac{1}{t\lambda_t}\right]+2W\sum_{t=1}^{T}\mathbb{E}[t\lambda_t]\delta_t\Big\}\,. \tag{61}$$

We next bound the terms on the right-hand side individually.

$$\sum_{t=1}^{T}\mathbb{E}\Big[\Big((t+1)\lambda_{t+1}-t\lambda_t\Big)\cdot\|\theta_{t+1}-\widehat{\theta}_{t+1}\|^2\Big]\le\frac{\ell_M}{6}\sum_{t=1}^{T}\mathbb{E}\Big[\sigma_{t+1}\cdot\|\theta_{t+1}-\widehat{\theta}_{t+1}\|^2\Big]$$
$$\le\frac{\ell_M}{6}\sum_{t=1}^{T}\mathbb{E}\Big[\sigma_{t+1}\|\theta_{t+1}-\widehat{\theta}_{t+1}\|^2\,\mathbb{I}(\sigma_{t+1}<3\sigma_{\min}/2)\Big]+\frac{\ell_M}{6}\sum_{t=1}^{T}\mathbb{E}\Big[\sigma_{t+1}\,\|\theta_{t+1}-\widehat{\theta}_{t+1}\|^2\,\mathbb{I}(\sigma_{t+1}>3\sigma_{\min}/2)\Big]$$
$$\le\frac{\ell_M}{4}\sigma_{\min}\sum_{t=1}^{T}\mathbb{E}\Big(\|\theta_{t+1}-\widehat{\theta}_{t+1}\|^2\Big)+\sum_{t=1}^{T}2\ell_MW^2e^{-c_2t/d}$$
$$\le\frac{\ell_M}{4}\sigma_{\min}\sum_{t=1}^{T}\mathbb{E}\Big(\|\theta_{t+1}-\widehat{\theta}_{t+1}\|^2\Big)+\frac{2\ell_M}{c_2d}W^2\,, \tag{62}$$

where in the last inequality, we used $\mathbb{P}(\sigma_{t+1}>3\sigma_{\min}/2)\le 2e^{-c_2dt}$, $\sigma_t\le 1$ and $\|\widehat{\theta}_t-\theta_t\|\le 2W$, according to Proposition 8.1.

The next term on the right-hand side of (61) is bounded in the following proposition.

theorem 8.2 *Using rule* (23) *for* λ_t*, we have*

$$\mathbb{E}\left[\frac{1}{t\lambda_t}\right]\le\frac{6}{\ell_M}\left(\tilde{c}d^2\log T+\frac{4d}{C_{\min}}\log T\right)\,, \tag{63}$$

where $\tilde{c}=\max(c_1,1/c_2)$ *and constants* c_1 *and* c_2 *are defined in Proposition 8.1 .*

Finally, for the last term, we note that Q_t is rank deficient for $t\le d$ and hence $\sigma_t=0$, for $1\le t\le d$. Further, the minimum eigenvalue of a matrix is a concave function over PSD matrices. By Jensen inequality, we have

$$\mathbb{E}(\lambda_t)=\frac{\ell_M}{6t}(1+\sum_{\ell=1}^{t}\mathbb{E}(\sigma_\ell))=\frac{\ell_M}{6t}\Big(1+\sum_{\ell=d+1}^{t}\mathbb{E}(\sigma_\ell)\Big)$$
$$\le\frac{\ell_M}{6t}\Big(1+\sum_{\ell=d+1}^{t}\sigma_{\min}\Big)\le\frac{\ell_M}{6t}\Big(1+\frac{t-d}{d}\Big)=\frac{\ell_M}{6d}\,. \tag{64}$$

In the last inequality, we used the fact that $\mathrm{Trace}(\Sigma)=\mathbb{E}(\|x_t\|^2)=1$, and thus $\sigma_{\min}\le 1/d$. Hence,

$$\sum_{t=1}^{T}\mathbb{E}[t\lambda_t]\delta_t\le\frac{\ell_M}{6d}\sum_{t=1}^{T}t\delta_t\,, \tag{65}$$

Using Equations (62), (63), (65) to bound the right-hand side of (61), we get

$$\mathbb{E}\left[\sum_{t=1}^{T}\langle x_t,\widehat{\theta}_t-\theta_t\rangle^2\cdot\mathbb{I}_{\mathcal{G}}\right]\le\left[\frac{64}{\ell_M^2}+\frac{12u_M^2}{\ell_M^2}\left(\tilde{c}+\frac{4}{C_{\min}d}\right)\right]\cdot d^2\log T$$
$$+\frac{48W^2}{\ell_M^2}+\frac{4W^2}{c_2d}+\frac{2W}{d}\sum_{t=1}^{T}t\delta_t+\frac{\sigma_{\min}}{2}\sum_{t=1}^{T}\mathbb{E}(\|\theta_t-\widehat{\theta}_t\|^2)\,. \tag{66}$$

Combining bounds (59),(60) and (65), we obtain

$$\frac{\sigma_{\min}}{2}\sum_{t=1}^{T}\mathbb{E}(\|\theta_t-\widehat{\theta}_t\|^2)\le\left[\frac{64}{\ell_M^2}+\frac{12u_M^2}{\ell_M^2}\left(\tilde{c}+\frac{4}{C_{\min}d}\right)\right]\cdot d^2\log T$$
$$+4W^2\left(\frac{1}{T}+\frac{12}{\ell_M^2}+\frac{1}{c_2d}\right)+\frac{2W}{d}\sum_{t=1}^{T}t\delta_t\,.$$

The result follows by recalling that $\sigma_{\min}\ge C_{\min}/d$ as stated by Assumption 6.

Acknowledgments

The author was partially supported by a grant from Google Faculty Research Award.

Appendix A. Proof of Lemma 10

We first state some properties of the the virtual valuation function φ and the price function g, given by Equation (7).

theorem A.1 *If $1-F$ is log-concave, then the virtual valuation function φ is strictly monotone increasing and the price function g satisfies $0 < g'(v) < 1$, for all values of $v \in \mathbb{R}$.*

We refer to (Javanmard and Nazerzadeh, 2016) (Lemmas 1 and 2 in Appendix A therein) for a proof of Proposition A.1.

For $\theta \in \Theta$ we have $\|\theta\| \le W$ and hence $|\langle x_t, \theta\rangle| \le \|x_t\|\|\theta\| \le W$ for all t. Applying Proposition A.1 (1-Lipschitz property of g),

$$p_t = g(\langle x_t, \theta_t\rangle) \le g(0) + |\langle x_t, \theta_t\rangle| \le \varphi^{-1}(0) + W\,.$$

Therefore,

$$|u_t(\theta)| \le |p_t| + |\langle x_t, \theta\rangle| \le \varphi^{-1}(0) + 2W\,. \tag{67}$$

Appendix B. Proof of Lemma 12

We note that the update rule (11) can be recast as $\widehat{\theta}_{t+1} = \arg\min_{\theta\in\Theta} \mathcal{C}_t(\theta)$, where

$$\mathcal{C}_t(\theta) = \eta_t \langle \nabla \ell_t(\widehat{\theta}_t), \theta\rangle + \frac{1}{2}\|\theta - \widehat{\theta}_t\|^2\,.$$

By convexity of $\mathcal{C}_t$ and optimality of $\widehat{\theta}_{t+1}$, we have $\langle \theta - \widehat{\theta}_{t+1}, \nabla \mathcal{C}_t(\widehat{\theta}_{t+1})\rangle \ge 0$ for all $\theta \in \Theta$. Setting $\theta = \theta_t$,

$$\langle \theta_t - \widehat{\theta}_{t+1}, \eta_t \nabla \ell_t(\widehat{\theta}_t) + \widehat{\theta}_{t+1} - \widehat{\theta}_t\rangle \ge 0\,. \tag{68}$$

Expanding $\ell_t(\theta)$ around $\widehat{\theta}_t$, we have

$$\ell_t(\widehat{\theta}_t) - \ell(\theta_t) = \langle \nabla \ell_t(\widehat{\theta}_t), \widehat{\theta}_t - \theta_t\rangle - \frac{1}{2}\langle \theta_t - \widehat{\theta}_t, \nabla^2 \ell_t(\tilde{\theta})(\theta_t - \widehat{\theta}_t)\rangle\,, \tag{69}$$

for some $\tilde{\theta}$ on the line segment between $\widetilde{\theta}_t$ and $\widehat{\theta}_t$. Recalling (12), the gradient and the hessian of ℓ_t read as

$$\nabla \ell_t(\theta) = \mu_t(\theta) x_t\,, \quad \nabla^2 \ell_t(\theta) = \eta_t(\theta) x_t x_t^{\mathsf{T}}\,, \tag{70}$$

with,

$$\begin{aligned}\mu_t(\theta) &= -\frac{f(u_t(\theta))}{F(u_t(\theta))}\mathbb{I}(y_t = -1) + \frac{f(u_t(\theta))}{1 - F(u_t(\theta))}\mathbb{I}(y_t = +1)\\ &= -\frac{\mathrm{d}}{\mathrm{d}x}\log F(u_t(\theta))\mathbb{I}(y_t = -1) - \frac{\mathrm{d}}{\mathrm{d}x}\log(1 - F(u_t(\theta)))\mathbb{I}(y_t = +1)\end{aligned} \tag{71}$$

$$\begin{aligned}\eta_t(\theta) &= \left(\frac{f(u_t(\theta))^2}{F(u_t(\theta))^2} - \frac{f'(u_t(\theta))}{F(u_t(\theta))}\right)\mathbb{I}(y_t = -1) + \left(\frac{f(u_t(\theta))^2}{(1 - F(u_t(\theta)))^2} + \frac{f'(u_t(\theta))}{1 - F(u_t(\theta))}\right)\mathbb{I}(y_t = +1)\\ &= -\frac{\mathrm{d}^2}{\mathrm{d}x^2}\log F(u_t(\theta))\mathbb{I}(y_t = -1) - \frac{\mathrm{d}^2}{\mathrm{d}x^2}\log(1 - F(u_t(\theta)))\mathbb{I}(y_t = +1)\,.\end{aligned} \tag{72}$$

Here, $u_t(\theta) = p_t - \langle x_t, \theta\rangle$, and $\frac{\mathrm{d}}{\mathrm{d}x}\log F(x)$ and $\frac{\mathrm{d}^2}{\mathrm{d}x^2}\log F(x)$ represent first and second derivative w.r.t x, respectively. In addition, using Equation (73)

$$|u_t(\theta)| \le \varphi^{-1}(0) + 2W = M\,, \quad \forall \theta \in \Theta\,. \tag{73}$$

Hence, invoking the definition of ℓ_M, as per Equation (16), we get that $\eta_t(\theta) \ge \ell_M$ and hence $\nabla^2 \ell_t(\tilde{\theta}) \succeq \ell_M x_t x_t^{\mathsf{T}}$.

Continuing from Equation (69), we get

$$\begin{aligned}\ell_t(\widehat{\theta}_t) - \ell(\theta_t) &\le \langle \nabla \ell_t(\widehat{\theta}_t), \widehat{\theta}_t - \theta_t\rangle - \frac{\ell_M}{2}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\\ &= \langle \nabla \ell_t(\widehat{\theta}_t), \widehat{\theta}_{t+1} - \theta_t\rangle + \langle \nabla \ell_t(\widehat{\theta}_t), \widehat{\theta}_t - \widehat{\theta}_{t+1}\rangle - \frac{\ell_M}{2}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\\ &\le \frac{1}{\eta_t}\langle \theta_t - \widehat{\theta}_{t+1}, \widehat{\theta}_{t+1} - \widehat{\theta}_t\rangle + \langle \nabla \ell_t(\widehat{\theta}_t), \widehat{\theta}_t - \widehat{\theta}_{t+1}\rangle - \frac{\ell_M}{2}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\\ &= \frac{1}{2\eta_t}\Big\{\|\theta_t - \widehat{\theta}_t\|^2 - \|\theta_t - \widehat{\theta}_{t+1}\|^2 - \|\widehat{\theta}_{t+1} - \widehat{\theta}_t\|^2\Big\}\\ &\quad + \langle \nabla \ell_t(\widehat{\theta}_t), \widehat{\theta}_t - \widehat{\theta}_{t+1}\rangle - \frac{\ell_M}{2}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\\ &= \frac{1}{2\eta_t}\Big\{\|\theta_t - \widehat{\theta}_t\|^2 - \|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2\Big\} + \frac{1}{2\eta_t}\Big\{\|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2 - \|\theta_t - \widehat{\theta}_{t+1}\|^2\Big\}\\ &\quad - \frac{1}{2\eta_t}\|\widehat{\theta}_{t+1} - \widehat{\theta}_t\|^2 + \langle \nabla \ell_t(\widehat{\theta}_t), \widehat{\theta}_t - \widehat{\theta}_{t+1}\rangle - \frac{\ell_M}{2}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\end{aligned} \tag{74}$$

We next note that the second term above can be bounded as

$$\frac{1}{2\eta_t}\Big\{\|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2 - \|\theta_t - \widehat{\theta}_{t+1}\|^2\Big\} = \frac{1}{\eta_t}\langle \theta_{t+1} - \widehat{\theta}_{t+1}, \theta_{t+1} - \theta_t\rangle \le \frac{2}{\eta_t} W\delta_t\,, \tag{75}$$

because $\theta_{t+1}, \widehat{\theta}_{t+1} \in \Theta$ and hence $\|\theta_{t+1} - \widehat{\theta}_{t+1}\| \le 2W$ by triangle inequality.

Further,

$$\begin{aligned}\langle \nabla \ell_t(\widehat{\theta}_t), \widehat{\theta}_t - \widehat{\theta}_{t+1}\rangle &\le \frac{1}{2\eta_t}\|\widehat{\theta}_{t+1} - \widehat{\theta}_t\|^2 + \frac{\eta_t}{2}\|\nabla \ell_t(\widehat{\theta}_t)\|^2\\ &\le \frac{1}{2\eta_t}\|\widehat{\theta}_{t+1} - \widehat{\theta}_t\|^2 + \frac{\eta_t}{2}|\mu(\widehat{\theta}_t)|^2\|x_t\|^2 \le \frac{1}{2\eta_t}\|\widehat{\theta}_{t+1} - \widehat{\theta}_t\|^2 + \frac{\eta_t}{2}u_M^2\,,\end{aligned} \tag{76}$$

where we used the inequality $2ab \le a^2 + b^2$ and the characterization of gradient (70). Note that by (73), $|u_t(\widehat{\theta})| \le M$ and by definition (15), $|\mu_t(\widehat{\theta}_t)| \le u_M$. Plugging in bounds from (75) and (76) in Equation (74), we arrive at

$$\ell_t(\widehat{\theta}_t) - \ell(\theta_t) \le \frac{1}{2\eta_t}\Big\{\|\theta_t - \widehat{\theta}_t\|^2 - \|\theta_{t+1} - \widehat{\theta}_{t+1}\|^2\Big\} + \frac{2}{\eta_t}W\delta_t + \frac{\eta_t}{2}u_M^2 - \frac{\ell_M}{2}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 \tag{77}$$

We use the shorthand $D_t = \frac{1}{2}\|\theta_t - \widehat{\theta}_t\|^2$. The result follows by summing the above bound over time:

$$\sum_{t=1}^{T} \ell_t(\widehat{\theta}_t) - \sum_{t=1}^{T} \ell_t(\theta_t) = \sum_{t=1}^{T} \Big(\frac{D_t}{\eta_t} - \frac{D_{t+1}}{\eta_{t+1}}\Big) + \sum_{t=1}^{T} D_{t+1}\Big(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t}\Big) + \frac{u_M^2}{2}\sum_{t=1}^{T}\eta_t + 2W\sum_{t=1}^{T}\frac{\delta_t}{\eta_t} - \frac{\ell_M}{2}\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2 .$$

The proof is concluded because $D_1 \le 2W^2$ as $\widehat{\theta}_1, \theta_1 \in \Theta$; therefore

$$\sum_{t=1}^{T} \Big(\frac{D_t}{\eta_t} - \frac{D_{t+1}}{\eta_{t+1}}\Big) = \frac{D_1}{\eta_1} - \frac{D_{T+1}}{\eta_{T+1}} \le \frac{D_1}{\eta_1} \le \frac{2W^2}{\eta_1} .$$

Appendix C. Proof of Lemma 13

By convexity of $\ell_t(\theta)$, we have

$$\ell_t(\theta_t) - \ell_t(\widehat{\theta}_t) \le \langle \nabla \ell_t(\theta_t), \widehat{\theta}_t - \theta_t\rangle = \mu_t(\theta_t)\langle x_t, \theta_t - \widehat{\theta}_t\rangle . \tag{78}$$

We denote $D_t = \mu_t(\theta_t)\langle x_t, \theta_t - \widehat{\theta}_t\rangle$ and let $\mathcal{F}_t$ be the σ-algebra generated by $\{z_t\}_{t=1}^T$. Since $\widehat{\theta}_t$ is $\mathcal{F}_{t-1}$ measurable, we have

$$\mathbb{E}(D_t|\mathcal{F}_{t-1}) = \mathbb{E}(\mu_t(\theta_t)|\mathcal{F}_{t-1})\langle x_t, \theta_t - \widehat{\theta}_t\rangle = 0 , \tag{79}$$

where $\mathbb{E}(\mu_t(\theta_t)|\mathcal{F}_{t-1}) = 0$ follows readily from Equation (71). Therefore, $D(T) \equiv \sum_{t=1}^T D_t$ is a martingale adapted to the filtration $\mathcal{F}_t$.

We next bound $\mathbb{E}[e^{\lambda D_t}|\mathcal{F}_{t-1}]$ for any $\lambda \in \mathbb{R}$. Conditional on $\mathcal{F}_{t-1}$, we have $|D_t| \le \beta_t$, with $\beta_t \equiv u_M|\langle x_t, \theta_t - \widehat{\theta}_t\rangle|$. Since $e^{\lambda z}$ is convex,

$$\begin{aligned}\mathbb{E}[e^{\lambda D_t}|\mathcal{F}_{t-1}] &\le \mathbb{E}\left[\frac{\beta_t - D_t}{2\beta_t}e^{-\lambda\beta_t} + \frac{\beta_t + D_t}{2\beta_t}e^{\lambda\beta_t}\middle|\mathcal{F}_{t-1}\right] \\ &= \mathbb{E}\left[\frac{e^{-\lambda\beta_t} + e^{\lambda\beta_t}}{2}\right] + \mathbb{E}[D_t|\mathcal{F}_{t-1}]\left(\frac{e^{-\lambda\beta_t} + e^{\lambda\beta_t}}{2\beta_t}\right) = \cosh(\lambda\beta_t) \le e^{\lambda^2\beta_t^2/2} .\end{aligned} \tag{80}$$

We are now ready to apply the following Bernstein-type concentration bound for martingale difference sequences, whose proof is given in Appendix D for the reader's convenience.

theorem C.1 *Consider a martingale difference sequence D_t adapted to a filtration $\mathcal{F}_t$, such that for any $\lambda \ge 0$, $\mathbb{E}[e^{\lambda D_t}|\mathcal{F}_{t-1}] \le e^{\lambda^2\sigma_t^2/2}$. Then, for $D(T) = \sum_{t=1}^T D_t$, the following holds true:*

$$\mathbb{P}(D(T) \ge \xi) \le e^{-\xi^2/(2\sum_{t=1}^T \sigma_t^2)} . \tag{81}$$

Combining Equation (78) and the result of Proposition C.1 we obtain

$$\mathbb{P}\Bigg(\sum_{t=1}^{T}\ell_t(\widehat{\theta}_t) - \sum_{t=1}^{T}\ell_t(\theta_t) \le -2\sqrt{\log T}\Big\{\sum_{t=1}^{T}\langle x_t, \theta_t - \widehat{\theta}_t\rangle^2\Big\}^{1/2}\Bigg) \le \frac{1}{T^2} . \tag{82}$$

The result follows.

Appendix D. Proof of Proposition C.1

We follow the standard approach of controlling the moment generating function of $D(T)$.Conditioning on $\mathcal{F}_{t-1}$ and applying iterated expectation yields

$$\mathbb{E}[e^{\lambda D(T)}] = \mathbb{E}\left[e^{\lambda\sum_{t=1}^{T-1}D_t}\cdot\mathbb{E}[e^{\lambda D_T}|\mathcal{F}_{T-1}]\right] \le \mathbb{E}\left[e^{\lambda\sum_{t=1}^{T-1}D_t}\right]e^{\lambda^2\sigma_T^2/2} . \tag{83}$$

Iterating this procedure gives the bound $\mathbb{E}[e^{\lambda\sum_{t=1}^T D_t}] \le e^{\lambda^2\sum_{t=1}^T\sigma_t^2/2}$, for all $\lambda \ge 0$.

Now by applying the exponential Markov inequality we get

$$\mathbb{P}(D(T) \ge \xi) = \mathbb{P}(e^{\lambda D(T)} \ge e^{\lambda\xi}) \le e^{-\lambda\xi}\mathbb{E}[e^{\lambda\sum_{t=1}^T D_t}] \le e^{-\lambda\xi}e^{\lambda^2(\sum_{t=1}^T\sigma_t^2)/2} . \tag{84}$$

Choosing $\lambda = \xi/(\sum_{t=1}^T\sigma_t^2)$ gives the desired result.

Appendix E. Proof of Proposition 8.1

We prove the result in a more general case, namely when the features are independent random vectors with bounded subgaussian norms.

Definition 14 *For a random variable z, its subgaussian norm, denoted by $\|z\|_{\psi_2}$ is defined as*

$$\|z\|_{\psi_2} = \sup_{p\ge 1} p^{-1/2}(\mathbb{E}|z|^p)^{1/p} . \tag{85}$$

Further, for a random vector z its subgaussian norm is defined as

$$\|z\|_{\psi_2} = \sup_{\|u\|\ge 1} \|\langle z, u\rangle\|_{\psi_2} . \tag{86}$$

We next recall the following result from (Vershynin, 2012) about random matrices with independent rows.

theorem E.1 *Suppose $x_\ell \in \mathbb{R}^d$ are independent random vectors generated from a distribution with covariance Σ and their subgaussian norms are bounded by K. Further, let $Q_t = (1/t)\sum_{\ell=1}^t x_\ell x_\ell^\mathsf{T}$. Then for every $s \ge 0$, the following inequality holds with probability at least $1 - 2\exp(-cs^2)$:*

$$\Big\|Q_t - \Sigma\Big\| \le \max(\delta, \delta^2) \qquad \text{where } \delta = C\sqrt{\frac{d}{t}} + \frac{s}{\sqrt{t}} . \tag{87}$$

Here C and $c > 0$ are constants that depend solely on K.

We next show that the feature vectors in our problem have bounded subgaussian norm. Given that $\|x_\ell\| \le 1$, for $\|u\| \le 1$, we have

$$\|\langle x_\ell, u\rangle\|_{\psi_2} = \sup_{p\ge1} p^{-1/2}(\mathbb{E}|\langle x_\ell, u\rangle|^p)^{1/p} \le \sup_{p\ge1} p^{-1/2}(\mathbb{E}[\|x_\ell\|\|u\|]^p)^{1/p} \le 1 .$$

Applying Proposition (E.1) with $K = 1$, there exist constants c_1, c_2 (depending on $C_{\min}$), such that for $t \geq c_1 d^2$, we have

$$\|Q_t - \Sigma\| \leq \frac{1}{2d} C_{\min} \leq \frac{1}{2}\sigma_{\min}\,, \tag{88}$$

with probability at least $1 - 2e^{-c_2 t/d}$. Weyl's inequality then implies that $|\sigma_t - \sigma_{\min}| \leq \sigma_{\min}/2$.

Also note that for $t \geq 1$,

$$\sigma_t \leq \|Q_t\| \leq \frac{1}{t}\sum_{\ell=1}^{t} \|x_\ell x_\ell^{\mathsf{T}}\| = \frac{1}{t}\sum_{\ell=1}^{t} \|x_\ell\|^2 = 1\,.$$

The proof is complete.

Appendix F. Proof of Lemma 8.2

The way we set λ_t (see Equation (23)), we have

$$\frac{1}{t\lambda_t} = \left(\frac{6}{\ell_M}\right) \frac{1}{1 + \sigma_1 + \sigma_2 + \ldots + \sigma_t}$$

Clearly, for $t \geq 1$, $1/(t\lambda_t) \leq 6/\ell_M$. Let $t_0 = \tilde{c} d^2 \log T$, with $\tilde{c} = \max(c_1, 1/c_2)$. For $T \geq t_0$, define the event $\mathcal{E}_T$ as follows

$$\mathcal{E}_T = \{\sigma_t \geq \sigma_{\min}/2, \text{ for } t_0 \leq t \leq T\}\,. \tag{89}$$

By applying Proposition 8.1 and union bounding over t, we get

$$\mathbb{P}(\mathcal{E}_T) \geq 1 - \sum_{t=t_0}^{T} 2e^{-c_2 t/d} \geq 1 - \frac{2d}{c_2} e^{-c_2 t_0/d} \tag{90}$$

Therefore,

$$\begin{aligned}
\sum_{t=t_0}^{T} \mathbb{E}\left[\frac{1}{t\lambda}\right] &\leq \mathbb{E}\left[\left(\sum_{t=t_0}^{T} \frac{1}{t\lambda}\right)\mathbb{I}(\mathcal{E}_T)\right] + \frac{6T}{\ell_M}\mathbb{P}(\mathcal{E}_T^c) \\
&= \frac{6}{\ell_M}\mathbb{E}\left[\left(\sum_{t=t_0}^{T} \frac{1}{1 + \sigma_1 + \ldots + \sigma_t}\right)\cdot \mathbb{I}(\mathcal{E}_T)\right] + \frac{6T}{\ell_M}\mathbb{P}(\mathcal{E}_T^c) \\
&\leq \frac{6}{\ell_M}\left(\sum_{t=1}^{T} \frac{1}{1 + \frac{t}{2}\sigma_{\min}} + \frac{2d}{c_2} T^{1 - c_2\tilde{c}d}\right) \\
&\leq \frac{12}{\ell_M}\left(\frac{1}{\sigma_{\min}}\log T + \frac{d}{c_2} T^{1-d}\right) \leq \frac{24d}{\ell_M C_{\min}}\log T\,.
\end{aligned} \tag{91}$$

For $t \geq 1$, we use the bound $1/(t\lambda_t) \leq 6/\ell_M$. Hence,

$$\sum_{t=1}^{T} \mathbb{E}\left[\frac{1}{t\lambda}\right] \leq \frac{6}{\ell_M}\left(t_0 + \frac{4d}{C_{\min}}\log T\right) \leq \frac{6}{\ell_M}\left(\tilde{c}d^2 \log T + \frac{4d}{C_{\min}}\log T\right) \tag{92}$$

The proof is complete.

References

Albert Ai, Alex Lapanowski, Yaniv Plan, and Roman Vershynin. One-bit compressed sensing with non-gaussian measurements. *Linear Algebra and its Applications*, 441:222–239, 2014.

Airbnb Documentation. Smart pricing: Set prices based on demand. `https://www.airbnb.com/help/article/1168/smart-pricing--set-prices-based-on-demand`, 2015.

Kareem Amin, Afshin Rostamizadeh, and Umar Syed. Repeated contextual auctions with strategic buyers. In *Advances in Neural Information Processing Systems*, pages 622–630, 2014.

Mark Bagnoli and Ted Bergstrom. Log-concave probability and its applications. *Economic theory*, 26(2):445–469, 2005.

Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

Omar Besbes and Assaf Zeevi. Dynamic pricing without knowing the demand function: risk bounds and near-optimal algorithms. *Operations Research*, 57:1407–1420, 2009.

Sonia A Bhaskar and Adel Javanmard. 1-bit matrix completion under exact low-rank constraint. In *Information Sciences and Systems (CISS), 2015 49th Annual Conference on*, pages 1–6. IEEE, 2015.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Josef Broder and Paat Rusmevichientong. Dynamic pricing under a general parametric choice model. *Operations Research*, 60(4):965–980, 2012.

Maxime C Cohen, Ilan Lobel, and Renato Paes Leme. Feature-based dynamic pricing. *ACM Conference on Economics and Computation*, 2016.

Arnoud V. den Boer and Bert Zwart. Simultaneously learning and optimizing using controlled variance pricing. *Management Science*, 60(3):770–783, 2013.

Arnoud V. den Boer and Bert Zwart. Mean square convergence rates for maximum quasi-likelihood estimators. *Stochastic Systems*, 4(2):375–403, 2014.

Vivek F Farias and Benjamin Van Roy. Dynamic pricing with a prior on market response. *Operations Research*, 58(1):16–29, 2010.

Eric C. Hall and Rebecca M. Willett. Online convex optimization in dynamic environments. *IEEE Journal of Selected Topics in Signal Processing*, 9(4):647–662, June 2015.

Adel Javanmard and Hamid Nazerzadeh. Dynamic pricing in high-dimensions. arXiv:1609.07574, 2016.

Bora Keskin and Assaf Zeevi. Dynamic pricing with an unknown demand model: Asymptotically optimal semi-myopic policies. *Operations Research*, 62(5):1142–1167, 2014.

Bora Keskin and Assaf Zeevi. Chasing demand: Learning and earning in a changing environment. *To appear in Mathematics of Operations Research*, 2016. doi: 10.1287/moor.2016.0807.

Ilan Lobel, Renato Paes Leme, and Adrian Vladu. Multidimensional binary search for contextual decision-making. *arXiv preprint arXiv:1611.00829*, 2016.

Roger B. Myerson. Optimal auction design. *Mathematics of Operations Research*, 6(1): 58–73, 1981.

Yaniv Plan and Roman Vershynin. One-bit compressed sensing by linear programming. *Communications on Pure and Applied Mathematics*, 66(8):1275–1297, 2013a.

Yaniv Plan and Roman Vershynin. Robust 1-bit compressed sensing and sparse logistic regression: A convex programming approach. *IEEE Transactions on Information Theory*, 59(1):482–494, 2013b.

Sheng Qiang and Mohsen Bayati. Dynamic pricing with demand covariates. Working Paper, 2016.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. In *Compressed sensing*, pages 210–268. Cambridge Univ. Press, 2012.

Zizhuo Wang, Shiming Deng, and Yinyu Ye. Close the gaps: A learning-while-doing algorithm for single-product revenue management problems. *Operations Research*, 62(2): 318–331, 2014.

Journal of Machine Learning Research 18 (2017) 1-24 Submitted 10/14; Revised 2/17; Published 7/17

Two New Approaches to Compressed Sensing Exhibiting Both Robust Sparse Recovery and the Grouping Effect

Mehmet Eren Ahsen MEHMET.ERENAHSEN@MSSM.EDU
Icahn School of Medicine at Mount Sinai
1468 Madison Avenue
New York, NY 10029, USA

Niharika Challapalli NIHARIKA15C@GMAIL.COM
Department of Electrical Engineering
University of Texas at Dallas
Richardson, TX 75080, USA

Mathukumalli Vidyasagar M.VIDYASAGAR@UTDALLAS.EDU, M.VIDYASAGAR@IITH.AC.IN
Department of Systems Engineering
University of Texas at Dallas
Richardson, TX 75080, USA, and
Department of Electrical Engineering
Indian Institute of Technology Hyderabad
Kandi, Telangana, India 502285

Editor: Saharon Rosset

Abstract

In this paper we introduce a new optimization formulation for sparse regression and compressed sensing, called CLOT (Combined L-One and Two), wherein the regularizer is a convex combination of the ℓ_1- and ℓ_2-norms. This formulation differs from the Elastic Net (EN) formulation, in which the regularizer is a convex combination of the ℓ_1- and ℓ_2-norm *squared*. It is shown that, in the context of compressed sensing, the EN formulation *does not achieve* robust recovery of sparse vectors, whereas the new CLOT formulation achieves robust recovery. Also, like EN but unlike LASSO, the CLOT formulation achieves the grouping effect, wherein coefficients of highly correlated columns of the measurement (or design) matrix are assigned roughly comparable values. It is already known LASSO does not have the grouping effect. Therefore the CLOT formulation combines the best features of both LASSO (robust sparse recovery) and EN (grouping effect).

The CLOT formulation is a special case of another one called SGL (Sparse Group LASSO) which was introduced into the literature previously, but without any analysis of either the grouping effect or robust sparse recovery. It is shown here that SGL achieves robust sparse recovery, and also achieves a version of the grouping effect in that coefficients of highly correlated columns belonging to the same group of the measurement (or design) matrix are assigned roughly comparable values.

Keywords: Sparse regression, compressed sensing, LASSO, Sparse Group LASSO, Elastic Net

©2017 Mehmet Eren Ahsen, Niharika Challapalli and Mathukumalli Vidyasagar.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/14-453.html.

1. Introduction

The LASSO and the Elastic Net (EN) formulations are among the most popular approaches for sparse regression and compressed sensing. In this section, we briefly review these two problems and their current status, so as to provide the background for the remainder of the paper.

1.1 Sparse Regression

In sparse regression, one is given a measurement matrix (also called a design matrix in statistics) $A \in \mathbb{R}^{m \times n}$ where $m \ll n$, together with a measurement or measured vector $y \in \mathbb{R}^m$. The objective is to choose a vector $x \in \mathbb{R}^n$ such that x is rather sparse, and Ax is either exactly or approximately equal to y. The problem of finding the most sparse x that satisfies $Ax = y$ is known to be NP-hard (Natarajan, 1995); therefore it is necessary to find alternate approaches.

For the sparse regression problem, the general approach is to determine the estimate $\hat{x}$ by solving the minimization problem

$$\hat{x} = \underset{z}{\operatorname{argmin}} \|y - Az\|_2^2 \text{ s.t. } \mathcal{R}(z) \leq \gamma, \tag{1}$$

or in Lagrangian form,

$$\hat{x} = \underset{z}{\operatorname{argmin}} \|y - Az\|_2^2 + \lambda \mathcal{R}(z), \tag{2}$$

where $\mathcal{R} : \mathbb{R}^n \to \mathbb{R}$ is known as a "regularizer," and γ, λ are adjustable parameters. Different choices of the regularizer lead to different approaches. With the choice $\mathcal{R}_{\text{ridge}}(z) = \|z\|_2^2$, the approach is known as ridge regression (Hoerl and Kennard, 1970), which builds on earlier work (Tikhonov, 1943). The LASSO approach (Tibshirani, 1996) results from choosing $\mathcal{R}_{\text{LASSO}}(z) = \|z\|_1$, while the Elastic Net (EN) approach (Zou and Hastie, 2005) results from choosing

$$\mathcal{R}_{\text{EN}}(z) = \alpha_1 \|z\|_1 + \alpha_2 \|z\|_2^2, \tag{3}$$

where α_1, α_2 are adjustable parameters. For later use, we redefine the EN regularizer as

$$\mathcal{R}_{\text{EN}}(z) = (1 - \mu)\|z\|_1 + \mu \|z\|_2^2, \tag{4}$$

where

$$\mu = \frac{\alpha_2}{\alpha_1 + \alpha_2} \in [0, 1]$$

is an adjustable parameter, and the constant $\alpha_1 + \alpha_2$ can be absorbed into the Lagrange multiplier λ in (2). Note that the EN regularizer function interpolates ridge regression and LASSO, in the sense that EN reduces to LASSO if $\mu = 0$ and to ridge regression if $\mu = 1$. A very general approach to regression using a convex regularizer is given in (Negabhan et al., 2012).

The LASSO approach can be shown to return a solution $\hat{x}$ with no more than m nonzero components, under mild regularity conditions; see (Osborne et al., 2000). There is no such bound on the number of components of $\hat{x}$ when EN is used. However, when the columns of the matrix A are highly correlated, then LASSO chooses just one of these columns and ignores the rest. Measurement matrices with highly correlated columns occur in many

practical situations, for example, in microarray measurements of messenger RNA, otherwise known as gene expression data. The EN approach was proposed at least in part to overcome this undesirable behavior of the LASSO formulation. It is shown in (Zou and Hastie, 2005, Theorem 1) that if two columns (say i and j) of the matrix A are highly correlated, then the corresponding components $\hat{x}_i$ and $\hat{x}_j$ of the EN solution are nearly equal. This is known as the "grouping effect", and the point is that EN demonstrates the grouping effect whereas LASSO does not.

1.2 Compressed Sensing

In compressed sensing, the objective is to *choose* the measurement matrix A (which is part of the data in sparse regression), such that whenever the vector x is nearly sparse, it is possible to nearly recover x from noise-corrupted measurements of the form $y = Ax + \eta$. Let us make the problem formulation precise. For this purpose we begin by introducing some notation.

Throughout, the symbol $[n]$ denotes the index set $\{1, \ldots, n\}$. The **support** of a vector $x \in \mathbb{R}^n$ is denoted by $\text{supp}(x)$ and is defined as

$$\text{supp}(x) := \{i \in [n] : x_i \neq 0\}.$$

A vector $x \in \mathbb{R}^n$ is said to be k-**sparse** if $|\text{supp}(x)| \leq k$. The set of all k-sparse vectors is denoted by Σ_k. The k-**sparsity index** of a vector x with respect to a given norm $\|\cdot\|$ is defined as

$$\sigma_k(x, \|\cdot\|) := \min_{z \in \Sigma_k} \|x - z\|. \tag{5}$$

It is obvious that $x \in \Sigma_K$ if and only if $\sigma_k(x, \|\cdot\|) = 0$ for every norm.

The general formulation of the compressed sensing problem given below is essentially taken from (Cohen et al., 2009). Suppose that $A \in \mathbb{R}^{m \times n}$ is the "measurement matrix", and $\Delta : \mathbb{R}^m \to \mathbb{R}^n$ is the "decoder map", where $m \ll n$. Suppose $x \in \mathbb{R}^n$ is an unknown vector that is to be recovered. The input to the decoder consists of $y = Ax + \eta$ where η denotes the measurement noise, and a prior upper bound in the form $\|\eta\|_2 \leq \epsilon$ is available; in other words, ϵ is a known number. In this set-up, the vector $\hat{x} = \Delta(y)$ is the approximation to the original vector x. With these conventions, we can now state the following.

Definition 1 *Suppose $p \in [1, 2]$. The pair (A, Δ) is said to achieve **robust sparse recovery** of order k with respect to $\|\cdot\|_p$ if there exist constants C and D that might depend on A and Δ but not on x or η, such that*

$$\|\hat{x} - x\|_p \leq \frac{1}{k^{1-1/p}}[C\sigma_k(x, \|\cdot\|_1) + D\epsilon]. \tag{6}$$

The restriction that $p \in [1, 2]$ is tied up with the fact that the bound on the noise is for the Euclidean norm $\|\eta\|_2$. The usual choices for p in (6) are $p = 1$ and $p = 2$.

Among the most popular approaches to compressed sensing is ℓ_1-norm minimization, which was popularized in a series of papers, of which we cite only (Candès and Tao, 2005; Candès et al., 2006; Candès, 2008; Donoho, 2006). The survey paper (Davenport et al.,

2012) has an extensive bibliography on the topic, as does the recent book (Foucart and Rauhut, 2013). In this approach, the estimate $\hat{x}$ is defined as

$$\hat{x} := \underset{z}{\text{argmin}} \|z\|_1 \text{ s.t. } \|Az - y\|_2 \leq \epsilon. \tag{7}$$

Note that the above definition does indeed define a decoder map $\Delta : \mathbb{R}^m \to \mathbb{R}^n$. In order for the above pair (A, Δ) to achieve robust sparse recovery, the matrix A is chosen so as to satisfy a condition defined next.

Definition 2 *A matrix $A \in \mathbb{R}^{m \times n}$ is said to satisfy the **Restricted Isometry Property (RIP)** of order k with constant δ_k if*

$$(1 - \delta_k)\|u\|_2^2 \leq \|Au\|_2^2 \leq (1 + \delta_k)\|u\|_2^2, \ \forall u \in \Sigma_k. \tag{8}$$

Starting with (Candès and Tao, 2005), several papers have derived sufficient conditions that the RIP constant of the matrix A must satisfy in order for ℓ_1-norm minimization to achieve robust sparse recovery. Recently, the "best possible" bound has been proved in (Cai and Zhang, 2014). These results are stated here for the convenience of the reader.

Theorem 3 *(See (Cai and Zhang, 2014, Theorem 2.1)) Suppose A satisfies the RIP of order tk for some number $t \geq 4/3$ such that tk is an integer, with $\delta_{tk} < \sqrt{(t-1)/t}$. Then the recovery procedure in (7) achieves robust sparse recovery of order k.*

Theorem 4 *(See (Cai and Zhang, 2014, Theorem 2.2)) Let $t \geq 4/3$. For all $\gamma > 0$ and all $k \geq 5/\gamma$, there exists a matrix A satisfying the RIP of order tk with constant $\delta_{tk} \leq \sqrt{(t-1)/t} + \gamma$ such that the recovery procedure in (7) fails for some k-sparse vector.*

Observe that the Lagrangian formulation of the LASSO approach is

$$\hat{x} := \underset{z}{\text{argmin}}[\|Az - y\|_2^2 + \lambda\|z\|_1],$$

whereas the Lagrangian formulation of (7) is

$$\hat{x} := \underset{z}{\text{argmin}}[\|z\|_1 + \beta\|Az - y\|_2],$$

which is essentially the same as the Lagrangian formulation of

$$\hat{x} = \underset{z}{\text{argmin}} \|Az - y\|_2 \text{ s.t. } \|z\|_1 \leq \gamma.$$

This last formulation of sparse regression is known as "square-root LASSO" (Belloni et al., 2014). Therefore the community refers to the approach to compressed sensing given in (7) as the LASSO, though this may not be strictly accurate.

1.3 Compressed Sensing with Group Sparsity

Over the years some variants of LASSO have been proposed for compressed sensing, such as the Group LASSO (GL) (Yuan and Lin, 2006) and the Sparse Group LASSO (SGL) (Simon et al., 2013). In the GL formulation, the index set $\{1, \ldots, n\}$ is partitioned into g disjoint sets $G_1, \ldots, G_g$, and the associated norm is defined as

$$\|z\|_{\mathrm{GL}} := \sum_{i=1}^{g} \|z_{G_i}\|_2, \tag{9}$$

where z_{G_i} denotes the projection of the vector z onto the components in G_i. The notation is intended to remind us that the norm depends on the specific partitioning $\mathcal{G}$. Some authors divide the term $\|z_{G_i}\|_2$ by $|G_i|$, but we do not do that. A further refinement of GL is the sparse group LASSO (SGL), in which the group structure is as before, but the norm is now defined as

$$\|z\|_{\mathrm{SGL},\mu} := \sum_{i=1}^{g} (1-\mu)\|z_{G_i}\|_1 + \mu\|z_{G_i}\|_2, \tag{10}$$

where as before $\mu \in [0,1]$. If $x \in \mathbb{R}^n$ is an unknown vector, then recovery of x is attempted via

$$\hat{x} = \underset{z}{\operatorname{argmin}} \|z\|_{\mathrm{GL}} \text{ s.t. } \|Az - y\|_2 \leq \epsilon \tag{11}$$

in Group LASSO, and via

$$\hat{x} = \underset{z}{\operatorname{argmin}} \|z\|_{\mathrm{SGL},\mu} \text{ s.t. } \|Az - y\|_2 \leq \epsilon \tag{12}$$

in Sparse Group LASSO.

The main idea behind GL is that one is less concerned about the number of nonzero components of $\hat{x}$, and more concerned about the number of distinct groups containing these nonzero components. Therefore GL attempts to choose an estimate $\hat{x}$ that has nonzero entries in as few distinct sets as possible. In principle, SGL tries to choose an estimate $\hat{x}$ that not only has nonzero components within as few groups as possible, but within those groups, has as few nonzero components as possible. Note that if $\mu = 0$, then SGL reduces to LASSO (because of the summability of the ℓ_1-norm), whereas if $\mu = 1$, then SGL reduces to GL. Note too that if $g = n$ and every set G_i is a singleton $\{i\}$, then GL reduces to LASSO.

1.4 Motivation and Contributions of the Paper

Now we come to the motivation and contributions of the present paper. The LASSO formulation is well-suited for compressed sensing (see Theorem 3), but not so well-suited for sparse regression, because it lacks the grouping effect. The EN formulation is well-suited for sparse regression as it exhibits the grouping effect, but it is not known whether it can achieve compressed sensing.

The first result presented in the paper is that if the EN regularizer of (4) is used instead of the ℓ_1-norm in (7), then the resulting approach *does not achieve* robust sparse recovery unless $m \geq n/4$, that is, the number of measurements grows linearly with respect to the size

of the vector. This would not be considered "compressed" sensing. This led us to formulate another regularizer, namely

$$\|z\|_{C,\mu} = (1-\mu)\|z\|_1 + \mu\|z\|_2. \tag{13}$$

Note that, while the EN regularizer in (4) is a convex function, it is not a norm. In contrast, $\|\cdot\|_{C,\mu}$ is not just convex but is also a norm. Also, the EN regularizer in its original form in (3) is intended to have *two* adjustable parameters. Our intent is that, in compressed sensing applications, the constant μ in (13) is a *fixed constant*, and not intended to be varied. Therefore, if the ℓ_1-norm in (7) is replaced by $\|\cdot\|_{C,\mu}$, then there is only one adjustable parameter, namely the Lagrange multiplier associated with the constraint. The same remark applies also to GL and SGL, that is, (11) and (12) respectively. We refer to $\|\cdot\|_{C,\mu}$ as the CLOT norm, with CLOT standing for Combined L-One and Two. It is shown that the CLOT norm combines the best features of both LASSO and EN, in that

- When the CLOT norm is used as the regularizer in sparse regression, the resulting solution exhibits the grouping effect.
- When the ℓ_1-norm is replaced by the CLOT norm in (7), the resulting solution achieves robust sparse recovery if the matrix A satisfies the RIP.
- Moreover, if μ in CLOT is set to zero so that CLOT becomes LASSO, the bound on the RIP constant reduces to the "best possible" bound in Theorem 3.

Clearly the CLOT norm is a special case of the SGL norm with the entire index set $[n]$ being taken as a single group (though the adjective "sparse" is no longer appropriate). This led us to explore whether the SGL norm achieves either grouping effect or robust sparse recovery. We are able to show that SGL does indeed achieve both.

Now we place these contributions in perspective. There is empirical evidence to support the belief that both the GL and the SGL formulations work well for compressed sensing. However, until the publication of a companion paper by a subset of the present authors (Ahsen and Vidyasagar, 2016), there were no proofs that either of these formulations achieved robust sparse recovery. In (Ahsen and Vidyasagar, 2016), it is shown that both the GL and SGL formulations achieve robust sparse recovery *provided the group sizes are sufficiently small*. This restriction on group sizes is removed in the present paper. Moreover, so far as the authors are aware, until now there are no results on the grouping effect for either of these formulations. In the present paper, it is shown that if two columns of the measurement matrix A *that belong to the same group* are highly correlated, then the corresponding components of the estimate $\hat{x}$ have nearly equal values. However, if two columns that belong to different groups are highly correlated, then their coefficients need not be nearly equal. From the standpoint of applications, this is a highly desirable property. To illustrate, suppose the groups represent biological pathways. Then one would wish to assign roughly similar weights to genes in the same pathway, but not necessarily to those in disjoint pathways.

Thus the contributions of the present paper are:

- To show that the EN *does not achieve* robust sparse recovery.
- To show that both the CLOT and SGL formulations achieve both robust sparse recovery as well as the grouping effect.

- To derive a condition under which CLOT achieves robust sparse recovery, which reduces to the "best possible" condition in Theorem 3 when μ is set to zero, so that CLOT becomes LASSO.

Taken together, these results might indicate that CLOT and SGL are attractive alternatives to the LASSO and EN formulations.

2. Main Theoretical Results

This section contains the main contributions of the paper. We begin by showing in Section 2.1 that the solution paths of EN and CLOT are identical if both λ_1 and λ_2 are treated as adjustable parameters. Therefore further research would be needed to establish whether CLOT offers any advantages over EN in numerical performance in sparse regression.

Then we present several theoretical advantages of CLOT over both EN and LASSO. First it is shown in Section 2.2 that the EN approach does not achieve robust sparse recovery, and is therefore not suitable for compressed sensing applications. Next, it is shown in Section 2.3 that the SGL formulation assigns nearly equal weights to highly correlated features *within the same group*, though not necessarily to highly correlated features from different groups. It follows as a corollary that CLOT assigns nearly equal weights to highly correlated features. Then it is shown in Section 2.4 that the SGL formulation achieves robust sparse recovery. The contents of a companion paper by a subset of the present authors Ahsen and Vidyasagar (2016) establish that SGL achieves robust sparse recovery of order k *provided that each group size is smaller than* k. There is no such restriction here. It follows as a corollary that CLOT also achieves robust sparse recovery.

2.1 Relationship Between Solution Paths of EN and CLOT

In this subsection, it is shown that if both μ and λ are tuned via cross-validation in (3), then the solution paths of CLOT are identical to those of EN when both λ_1 and λ_2 are tuned. However, it is shown via an example that if μ is kept fixed and *only* λ is tuned in (3), then CLOT and EN have different solution paths.

Towards this end, we rewrite the CLOT formulation with both μ and λ being tuned in the form

$$\hat{x}_{\rm CLOT} := \underset{z}{\rm argmin}[\|y - Ax\|_2^2 + \lambda_1\|z\|_1 + \lambda_2\|z\|_2]. \tag{14}$$

It is easy to see that the transformation

$$\mu = \frac{\lambda_2}{\lambda_1 + \lambda_2}, \lambda = \lambda_1 + \lambda_2$$

maps (14) into (3). In the other direction, we would define

$$\lambda_1 = (1 - \mu)\lambda, \lambda_2 = \mu\lambda.$$

We are grateful to one of the reviewers for pointing out the result as described in Theorem 5, and providing a proof.

Theorem 5 *Given $y \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m\times n}$, define two vectors:*

$$\hat{x}_{\rm CLOT}(\lambda_1, \tilde{\lambda}_2) = \underset{z}{\rm argmin}[\|y - Az\|_2^2 + \lambda_1\|z\|_1 + \tilde{\lambda}_2\|z\|_2],$$

$$\hat{x}_{\rm EN}(\lambda_1, \hat{\lambda}_2) = \underset{z}{\rm argmin}[\|y - Az\|_2^2 + \lambda_1\|z\|_1 + \hat{\lambda}_2\|z\|_2^2].$$

Then for each fixed $\lambda_1 > 0$ and each $\tilde{\lambda}_2 > 0$, there exists a $\hat{\lambda}_2 > 0$ such that

$$\hat{x}_{\rm CLOT}(\lambda_1, \tilde{\lambda}_2) = \hat{x}_{\rm EN}(\lambda_1, \hat{\lambda}_2),$$

and vice versa.

Proof We begin with the following rather obvious observation. Suppose $f(\cdot)$ and $g(\cdot)$ are convex functions, and consider two problems:

$$\text{(P1)} \quad \hat{x}_1(\lambda) = \underset{z}{\rm argmin}[f(z) + \lambda g(z)],$$

$$\text{(P2)} \quad \hat{x}_2(c) = \underset{z}{\rm argmin}\, f(z) \text{ s.t. } g(z) \le c.$$

Then for each λ there exists a c such that $\hat{x}_1(\lambda) = \hat{x}_2(c)$, and vice versa. To establish this, write down the optimality conditions for the two problems, with $\partial f(\cdot)$, $\partial g(\cdot)$ denoting the subgradient sets of $f(\cdot)$, $g(\cdot)$ respectively. Then a necessary and sufficient condition for $\hat{x}_1(\lambda)$ to be the solution of (P1) is:

$$\mathbf{0} \in \partial f(\hat{x}_1(\lambda)) + \lambda\partial g(\hat{x}_1(\lambda)), \tag{15}$$

where $\mathbf{0}$ denotes the zero vector. Similarly, for (P2) the necessary and sufficient conditions are the existence of a constant λ^* such that

$$\mathbf{0} \in \partial f(\hat{x}_1(\lambda^*)) + \lambda\partial g(\hat{x}_1(\lambda^*)), \text{ and } \lambda^*(g(\hat{x}_1(\lambda^*)) - c) = 0. \tag{16}$$

Suppose (15) holds; then (16) holds with $c = g(\hat{x}_1(\lambda))$. Conversely, suppose (16) holds; then (15) holds with $\lambda = \lambda^*$.

Now apply this reasoning with $f(z) = \|y - Az\|_2^2 + \lambda_1\|z\|_1$, $g_1(z) = \|z\|_2$, $g_2(z) = \|z\|_2^2$. Then each $\hat{x}_{\rm CLOT}(\lambda_1, \tilde{\lambda}_2)$ equals the minimizer of $f(z)$ subject to $\|z\|_2 \le c$ for some c, while each $\hat{x}_{\rm EN}(\lambda_1, \hat{\lambda}_2)$ equals the minimizer of $f(z)$ subject to $\|z\|_2^2 \le c'$ for some c'. However, it is obvious that

$$[\|z\|_2 \le c] \iff [\|z\|_2^2 \le c^2].$$

Therefore the theorem is proved. □ ■

2.2 Lack of Robust Sparse Recovery of the Elastic Net Formulation

The first result of this section shows that EN formulation does not achieve robust sparse recovery, and therefore is not suitable for compressed sensing applications.

Theorem 6 *Suppose a matrix $A \in \mathbb{R}^{m\times n}$ has the following property: There exist constants C and D such that, whenever $y = Ax + \eta$ for some $x \in R^n$ and $\eta \in \mathbb{R}^m$ with $\|\eta\|_2 \leq \epsilon$, the solution*

$$\hat{x}_{\text{EN}} := \underset{z\in\mathbb{R}^n}{\text{argmin}}[(1-\mu)\|z\|_1 + \mu\|z\|_2^2] \text{ s.t. } \|y - Az\|_2 \leq \epsilon$$

satisfies

$$\|\hat{x}_{\text{EN}} - x\|_2 \leq C\sigma_k(x, \|\cdot\|_1) + D\epsilon. \tag{17}$$

Then

$$m \geq n/4. \tag{18}$$

Proof: Let $\mathcal{N}(A)$ denote the null space of the matrix A, that is, the set of all $h \in \mathbb{R}^n$ such that $Ah = 0$. Let $h \in \mathcal{N}(A)$ be arbitrary, and let $\Lambda \subseteq \{1, \ldots, n\}$ denote the index set of the k largest components of h by magnitude. Therefore

$$\|h_{\Lambda^c}\|_2 = \sigma_k(h, \|\cdot\|_2).$$

Next, (17) implies that, if $\eta = 0$, then $\hat{x}_{\text{EN}} = x$ for all $x \in \Sigma_k$. In other words,

$$x = \underset{z\in\mathbb{R}^n}{\text{argmin}}[(1-\mu)\|z\|_1 + \mu\|z\|_2^2] \text{ s.t. } Az = Ax,$$

or equivalently,

$$(1-\mu)\|x\|_1 + \mu\|x\|_2^2 \leq (1-\mu)\|z\|_1 + \mu\|z\|_2^2, \ \forall z \in A^{-1}(\{Ax\}), \ \forall x \in \Sigma_k. \tag{19}$$

Now observe that, because $h \in \mathcal{N}(A)$, we have that

$$Ah_\Lambda = -Ah_{\Lambda^c},$$

and more generally,

$$A(\beta h_\Lambda) = A(-\beta h_{\Lambda^c}), \ \forall \beta > 0.$$

Apply (19) with $x = \beta h_\Lambda \in \Sigma_k$ and $z = -A\beta h_{\Lambda^c} \in A^{-1}(\{Ax\})$. This leads to

$$(1-\mu)\beta\|h_\Lambda\|_1 + \beta^2\mu\|h_\Lambda\|_2^2 \leq (1-\mu)\beta\|h_{\Lambda^c}\|_1 + \beta^2\mu\|h_{\Lambda^c}\|_2^2.$$

Now divide both sides by $\beta^2\mu$, and observe that, for each fixed $\mu > 0$,

$$\frac{1-\mu}{\beta\mu} \to 0 \text{ as } \beta \to \infty.$$

Therefore

$$\|h_\Lambda\|_2^2 \leq \|h_{\Lambda^c}\|_2^2, \text{ or } \|h_\Lambda\|_2 \leq \|h_{\Lambda^c}\|_2, \ \forall h \in \mathcal{N}(A).$$

Next

$$\|h\|_2 \leq \|h_\Lambda\|_2 + \|h_{\Lambda^c}\|_2 \leq 2\|h_{\Lambda^c}\|_2, \ \forall h \in \mathcal{N}(A).$$

Equivalently

$$\|h\|_2 \leq 2\sigma_k(h, \|\cdot\|_2), \ \forall h \in \mathcal{N}(A).$$

This is Equation (5.2) of Cohen-Dahmen-Devore (2009) with $C_0 = 2$. As shown in Theorem 5.1 of that paper, this implies that $m \geq n/C_0^2 = n/4$, which is the desired conclusion. □

Note that the proof of Theorem 6 remains valid even if we were to allow the constant μ to be "tuned", provided that it is bounded away from zero. In other words, the proof does not make use of the fact that μ is a fixed constant. Therefore even in the "naive" version of EN, in which the regularizer is defined as in (3), and both constants α_1 and α_2 are adjusted, robust sparse recovery requires that $m \geq n/4$ provided only that the ratio α_1/α_2 remains bounded away from zero as both parameters are tuned.

2.3 Grouping Property of the SGL and CLOT Formulations

One advantage of the EN over LASSO is that the former assigns roughly equal weights to highly correlated features, as shown in (Zou and Hastie, 2005, Theorem 1) and referred to as the grouping effect. In contrast, if LASSO chooses one feature among a set of highly correlated features, then generically it assigns a zero weight to all the rest. To illustrate, if two columns of A are identical, then in principle LASSO could assign nonzero weights to both columns; however, the slightest perturbation in the data would cause one or the other weight to become zero. The drawback of this is that the finally selected feature set is very sensitive to noise in the measurements. In this section we prove an analog of (Zou and Hastie, 2005, Theorem 1) for SGL formulation. Our result states that if two highly correlated features *within the same group* are chosen by SGL, then they will have roughly similar weights. Since CLOT is a special case of SGL with the entire feature set treated as one group, it follows that CLOT assigns roughly similar weights to highly correlated features in the entire set of features. As a result, the final feature sets obtained using SGL or CLOT are less sensitive to noise in measurements than the ones obtained using LASSO.

Theorem 7 *Let $y \in \mathbb{R}^m, A \in \mathbb{R}^{m\times n}$ be some vector and matrix respectively. Without loss of generality, suppose that y is centered, i.e. $y^t\mathbf{e}_m = 0$, where $\mathbf{e}_m$ denotes a column vector consisting of m ones, and that A is standardized, i.e. $\|a_j\|_2 = 1$ where a_j denotes the j-th column of A. Suppose $\mu > 0$, and let $\mathcal{G}$ denote a partition of $\{1, \ldots, n\}$ into g disjoint subsets. Define*

$$\hat{x} := \underset{z}{\text{argmin}}\, \lambda\|y - Az\|_2^2 + \|z\|_{\text{SGL},\mu}, \tag{20}$$

where $\lambda > 0$ is a Lagrange multiplier. Suppose that, for two indices i, j belonging to the same group G_s, we have that $\hat{x}_i\hat{x}_j \neq 0$, where $\hat{x}_i, \hat{x}_j$ denote the components of the vector $\hat{x}$. By changing the sign of one of the columns of A if necessary, it can be assumed that $\hat{x}_i\hat{x}_j > 0$. Define

$$d(i,j) := \frac{|\hat{x}_i - \hat{x}_j|}{2\lambda\|y\|_2}, \rho(i,j) = a_i^t a_j.$$

Then

$$d(i,j) \leq \frac{\sqrt{2(1-\rho(i,j))}\|\hat{x}^s\|_2}{\mu}, \tag{21}$$

where $\hat{x}^s$ is shorthand for $\hat{x}_{G_s}$.

Proof: Define

$$\begin{aligned} L(z,\mu) &:= \lambda\|y-Az\|_2^2 + \|z\|_{\text{SGL},\mu} \\ &= \lambda\|y-Az\|_2^2 + (1-\mu)\|z\|_1 + \mu\sum_{s=1}^{g}\|z^s\|_2, \end{aligned}$$

where, as above, z^s denotes Z_{G_s}. Then L is differentiable with respect to z_i whenever $z_i \neq 0$. In particular, since both $\hat{x}_i$ and $\hat{x}_j$ are nonzero by assumption, it follows that

$$\left.\frac{\partial L}{\partial z_i}\right|_{z=\hat{x}} = \left.\frac{\partial L}{\partial z_j}\right|_{z=\hat{x}} = 0.$$

Expanding the partial derivatives leads to

$$-2\lambda a_i^t(y-A\hat{x}) + (1-\mu)\text{sign}(\hat{x}_i) + \mu\frac{\hat{x}_i}{\|\hat{x}^s\|_2} = 0,$$

$$-2\lambda a_j^t(y-A\hat{x}) + (1-\mu)\text{sign}(\hat{x}_j) + \mu\frac{\hat{x}_j}{\|\hat{x}^s\|_2} = 0.$$

Subtracting one equation from the other gives

$$2\lambda(a_j^t - a_i^t)(y-A\hat{x}) + \mu\frac{\hat{x}_i - \hat{x}_j}{\|\hat{x}^s\|_2} = 0.$$

Hence

$$\begin{aligned} \frac{|\hat{x}_i - \hat{x}_j|}{\|\hat{x}^s\|_2} &= \frac{2\lambda}{\mu}|(a_j^t - a_i^t)(y-A\hat{x})| \\ &\leq \frac{2\lambda}{\mu}\|a_j - a_i\|_2\|y-A\hat{x}\|_2 \\ &\leq \frac{2\lambda}{\mu}\sqrt{2(1-\rho(i,j))}\|y\|_2. \end{aligned}$$

In the last step, we use the fact that

$$\|y-A\hat{x}\|_2 \leq \|y-A\mathbf{0}\|_2 = \|y\|_2.$$

Rearranging gives

$$\frac{1}{2\lambda}\frac{|\hat{x}_i - \hat{x}_j|}{\|y\|_2} \leq \frac{\sqrt{2(1-\rho(i,j))}\|\hat{x}^s\|_2}{\mu},$$

which is the desired conclusion. □

Let us illustrate the above result using the CLOT formulation. In the case of CLOT formulation we have $g = 1$, $\mathcal{G} = \{G_1\}$, $G_1 = \{1, \cdots, n\}$, and the inequality in (21) becomes

$$d(i,j) \leq \frac{\sqrt{2(1-\rho(i,j))}\|\hat{x}\|_2}{\mu}, \tag{22}$$

where $\hat{x}$ is the solution of the CLOT formulation, and

$$d(i,j) = \frac{|\hat{x}_i - \hat{x}_j|}{2\lambda\|y\|_2}. \tag{23}$$

Now suppose that two indices i and j are highly correlated such that $\rho(i,j) \approx 1$, so that the right hand side of the inequality in (22) is almost equal to zero. Combining this with (23) we can conclude $\hat{x}_i \approx \hat{x}_j$, so CLOT assigns similar weights to highly correlated variables. □

Though the focus of the present paper is not on the GL formulation, we digress briefly to discuss the implications of Theorem 7 for GL. This theorem also implies that the GL formulation exhibits the grouping effect, because GL is a special case of SGL with $\mu = 1$. Indeed, it can be observed from (21) that the bound on the right side is minimized by setting $\mu = 1$, that is, using GL instead of SGL. This is not surprising, because SGL not only tries to minimize the number of distinct groups containing the support of $\hat{x}$, but within each group, tries to choose as few elements as possible. Thus, within each group, SGL inherits the weaknesses of LASSO. Thus one would expect that, within each group, the feature set chosen would become more sensitive as we decrease μ.

2.4 Robust Sparse Recovery of the SGL and CLOT Formulations

In this subsection, we present some sufficient conditions for the SGL and CLOT formulations to achieve robust sparse recovery. When CLOT is specialized to LASSO by setting $\mu = 0$, the sufficient condition reduces to the "tight" bound given in Theorem 3.

Recall the definitions. The CLOT norm with parameter μ is given by

$$\|z\|_{C,\mu} := (1-\mu)\|z\|_1 + \mu\|z\|_2,$$

while the SGL norm is given by

$$\|z\|_{SGL,\mu} := \sum_{i=1}^{g}[(1-\mu)\|z_{G_i}\|_1 + \mu\|z_{G_i}\|_2]$$

Recall also the problem set-up. The measurement vector y equals $Ax + \eta$ where $\|\eta\|_2 \leq \epsilon$, a known upper bound. The recovered vector $\hat{x}$ is defined as

$$\hat{x} = \underset{z}{\text{argmin}}\,\|z\|_{\text{SGL},\mu} \text{ s.t. } \|Az - y\|_2 \leq \epsilon \tag{24}$$

if SGL is used, and as

$$\hat{x} = \underset{z}{\text{argmin}}\,\|z\|_{C,\mu} \text{ s.t. } \|Az - y\|_2 \leq \epsilon \tag{25}$$

if CLOT is used.

Definition 8 *A matrix $A \in \mathbb{R}^{m\times n}$ is said to satisfy the ℓ_2* ***robust null space property (RNSP)*** *if there exist constants $\rho \in (0,1)$ and $\tau \in \mathbb{R}_+$ such that, for all sets $S \subseteq [n]$ with $|S| \leq k$, we have*

$$\|h_S\|_2 \leq \frac{\rho}{\sqrt{k}}\|h_{S^c}\|_1 + \frac{\tau}{\sqrt{k}}\|Ah\|_2. \tag{26}$$

This property was apparently first introduced in (Foucart and Rauhut, 2013, Definition 4.21). Note that the definition in Foucart and Rauhut (2013) has just τ in place of $\tau/\sqrt{k}$. It is easy to show that, if (26) holds, then

$$\|h_S\|_1 \leq \rho\|h_{S^c}\|_1 + \tau\|Ah\|_2.$$

The following result is established in Ranjan and Vidyasagar (2016) in the context of group sparsity, but is new even for conventional sparsity. The reader is directed to that source for the proof.

Theorem 9 *((Ranjan and Vidyasagar, 2016, Theorem 5)) Suppose that, for some number $t > 1$, the matrix A satisfies the RIP of order tk with constant $\delta_{tk} < \sqrt{(t-1)/t}$. Let δ be an abbreviation for δ_{tk}, and define the constants*

$$\nu := \sqrt{t(t-1)} - (t-1) \in (0, 0.5), \tag{27}$$

$$a := [\nu(1-\nu) - \delta(0.5 - \nu + \nu^2)]^{1/2}, \tag{28}$$

$$b := \nu(1-\nu)\sqrt{1+\delta}, \tag{29}$$

$$c := \left[\frac{\delta\nu^2}{2(t-1)}\right]^{1/2}. \tag{30}$$

Then A satisfies the ℓ_2-robust null space property with

$$\rho = \frac{c}{a} < 1, \tau = \frac{b\sqrt{k}}{a^2}. \tag{31}$$

In Theorem 10 below, it is assumed that the matrix A satisfies the RIP of order tk with $\delta_{tk} < \sqrt{(t-1)/t}$, in accordance with Theorem 3. With this assumption, we prove bounds on the residual error $\|\hat{x} - x\|_1$ with SGL; the bounds for CLOT can be obtained simply by setting $g = 1$ in the SGL bounds. Note that, once bounds for $\|\hat{x} - x\|_1$ are proved, it is possible to extend the bounds to $\|\hat{x} - x\|_p$ for all $p \in [1, 2]$; see (Foucart and Rauhut, 2013, Theorem 4.22).

Theorem 10 *Suppose $x \in \mathbb{R}^n$ and that $A \in \mathbb{R}^{m\times n}$ satisfies the RIP of order tk with constant $\delta = \delta_{tk} < \sqrt{(t-1)/t}$, and define constants ρ, τ as in* (31)*. Suppose that*

$$\mu < \frac{1-\rho}{\sqrt{g}(1+\rho)}. \tag{32}$$

Define

$$\gamma = \frac{\mu\sqrt{g}}{1-\mu}. \tag{33}$$

With these assumptions,

$$\|\hat{x} - x\|_1 \leq C\sigma_k(x, \|\cdot\|_1) + D\epsilon, \tag{34}$$

$$\|\hat{x} - x\|_p \leq \frac{1}{k^{1-1/p}}[(1+\rho)C\sigma_k(x, \|\cdot\|_1) + ((1+\rho)D + 2\tau)\epsilon], \ \forall p \in (1, 2], \tag{35}$$

where

$$C = \frac{2(1+\rho)}{(1-\gamma) - (1+\gamma)\rho}, D = \frac{4\tau}{(1-\gamma) - (1+\gamma)\rho}. \tag{36}$$

The proof of the above theorem is presented in an appendix, due to its length.

In the above theorem, we started with the restricted isometry constant δ and computed an upper bound on μ in order for SGL and CLOT to achieve robust sparse recovery. As δ_{tk} gets closer to the limit $\delta_{tk} < \sqrt{(t-1)/t}$ (which is known to be the best possible in view of Theorem 4), the limit on μ would approach zero. It is also possible to start with μ and find an upper bound on δ, by rearranging the inequalities. As this involves just routine algebra, we simply present the final bound. Given the number $t \geq 4/3$, define ν as in (27), and define

$$\theta_1 := \nu(1-\nu) \in (0, 0.25), \theta_2 := 0.5 - \theta_1 \in (0.25, 0.5), \tag{37}$$

$$\theta_3 := \frac{\nu^2}{2(t-1)}. \tag{38}$$

Given $\mu > 0$, define γ as in (33), and define

$$\bar{\rho} := \frac{1-\gamma}{1+\gamma}.$$

If the matrix A satisfies the RIP of order tk with a constant $\delta = \delta_{tk}$, then SGL achieves robust sparse recovery of order k provided

$$\delta < \bar{\rho}^2 \frac{\theta_1}{\theta_3 + \bar{\rho}^2\theta_2}, \tag{39}$$

3. Numerical Examples

In this section we present two simulation studies, to demonstrate the the grouping effect of CLOT and the lack of robust sparse recovery of EN, respectively.

3.1 Grouping Effect of CLOT

To illustrate that CLOT demonstrates the grouping effect as does EN (see Theorem 7), we ran the same example as at the end of (Zou and Hastie, 2005, Section 5). Specifically, we chose Z_1 and Z_2 to be two independent $U(0, 20)$ random variables, and the observation y as $N(Z_1 + 0.1Z_2, 1)$. The six observations were

$$x_1 = Z_1 + \epsilon_1, x_2 = -Z_1 + \epsilon_2, x_3 = Z_1 + \epsilon_3, x_4 = Z_2 + \epsilon_4, x_5 = -Z_2 + \epsilon_5, x_6 = Z_2 + \epsilon_6,$$

where the ϵ_i are i.i.d. $N(0, 1/16)$. The objective was to express y as a linear combination of x_1 through x_6. In other words, we wished to express $y = X\boldsymbol{\beta}$, where X is the matrix with the x_i as columns, and $\boldsymbol{\beta}$ is a 6×1 vector. Ideally the outcome should be to assign

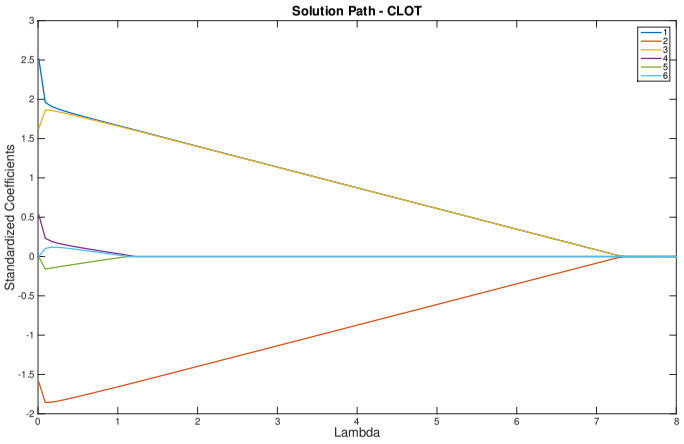


Figure 1: Solution Paths for CLOT

high weights to the correlated group x_1, x_2, x_3 and low weights to x_4, x_5, x_6. Therefore, if $\boldsymbol{\beta}$ denotes the six-dimensional coefficient vector, we should have that

$$\beta_1 \approx -\beta_2 \approx \beta_3,$$

and that β_4 through β_6 are much smaller than β_1 through β_3.

The three algorithms LASSO, EN, and CLOT were implemented via the Lagrangian formulation in (2), with λ being increased. Clearly, when λ is sufficiently large, the optimal value of $\boldsymbol{\beta}$ is the zero vector. The "sufficiently large" value of λ varies from one algorithm to the next. Figures 1 through 3 show the solution trajectories of the three algorithms as functions of λ, with μ set equal to 0.5. From Figures 1 and 2, it is clear that both CLOT and EN very quickly reach the correct proportionalities between the large coefficient values, which eventually become smaller and go to zero as λ becomes larger. In contrast, the LASSO solutions are quite inaccurate.

3.2 Lack of Robust Sparse Recovery by EN

In this subsection we illustrate Theorem 6. In this set-up, $n = 4,000$ and $k = 3$. The first three components of the vector x are assigned values at random using the `Matlab rand` function, which resulted in [0.8147 0.9058 0.1270]. The remaining components of x were set equal to zero. We could have chosen not just the values but also the location of the nonzero components at random; but this would have been just a permuation of the above example.

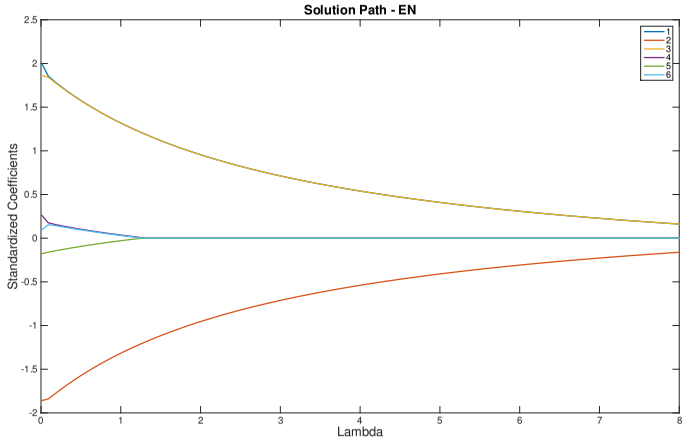


Figure 2: Solution Path for EN

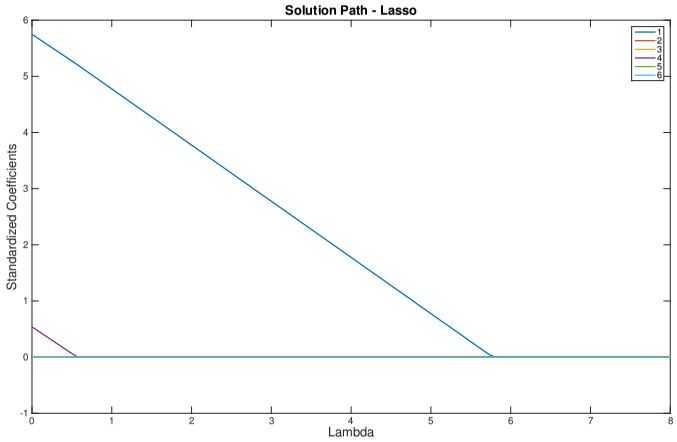


Figure 3: Solution Path for LASSO

Next, in order to achieve robust sparse recovery of order k, following Theorem 3 we needed to choose a $t \geq 4/3$ and then choose a matrix A such that A satisfied the RIP of order $\lceil tk \rceil$ with constant $\delta < \sqrt{(t-1)/t}$. We chose $t = 1.5$, which resulted in $\lceil tk \rceil = 5$, and $\delta_5 < \sqrt{1/3} \approx 0.5774$. We chose $\delta_5 = 0.4$. Therefore we had to choose an integer m and a matrix $A \in \mathbb{R}^{m \times n}$ such that A satisfied the RIP of order 5 with constant $\delta_5 \leq 0.4$. Such a matrix was constructed using the deterministic procedure suggested in (DeVore, 2007). This required the choice of an integer $r \geq 2$ and a prime number p such that

$$p \geq \max \left\{ \frac{(\lceil tk \rceil - 1)r}{\delta}, n^{1/(r+1)} \right\} = \max\{20, 15.87\} = 20.$$

and led to a binary matrix of dimensions $p^2 \times p^{r+1}$. The smallest prime number greater than 20 is $p = 23$, and $m = p^2 = 529$. Note that $m < n/4$. The main advantages of the construction in (DeVore, 2007) are that (i) the construction is deterministic, (ii) the matrix A is binary, and (iii) only a fraction $1/p$ of the elements of A are equal to 1, and the rest are equal to zero. This makes computation very fast.

Once the matrix A was chosen, we defined the measured vector as $y = Ax$; that is, we did not introduce any measurement noise. Then we computed estimates of x using both CLOT and EN. With $\delta = 0.4$, the constant ρ defined in (31) becomes 0.6551, and the bound in (32), after substituting $g = 1$, becomes $\mu < 0.2084$. Therefore we chose $\mu = 0.2$, and defined

$$\hat{x}_{\text{CLOT}} = \underset{z}{\text{argmin}} \, \|z\|_{C,\mu} \text{ s.t. } Az = y,$$

$$\hat{x}_{\text{EN}} = \underset{z}{\text{argmin}} (1-\mu)\|z\|_1 + \mu\|z\|_2^2 \text{ s.t. } Az = y,$$

with the same value of $\mu = 0.2$ in both cases. Then we replaced x by $10^c x$ for $c = 1, 2, 3, 4$ and recomputed the estimates. Because $m < n/4$, according to Theorem 7, EN should fail as the norm of the vector x is increased.

The above constrained optimization problems were solved using the cvx package in `Matlab`, on an HP Pavilion laptop. The actual results are shown in Table 1. For compactness only the first three components of $\hat{x}$ are shown. As can be seen, when $c = 0$ or $c = 1$, both CLOT and EN give the correct answer. However, for larger values of c, the CLOT estimate simply got multiplied by the same scale factor, whereas the EN estimate started diverging from the true value with $c = 2$; the algorithm failed to converge with $c = 4$.

4. Discussion and Concluding Remarks

In this paper we have introduced a new optimization formulation called CLOT (Combined L-One and Two), wherein the regularizer is a convex combination of the ℓ_1- and ℓ_2-norms. This formulation differs from the Elastic Net (EN) formulation, in which the regularizer is a convex combination of the ℓ_1- and ℓ_2-norm *squared*. This seemingly simple modification has fairly significant consequences. In particular, it is shown in this paper that the EN formulation *does not achieve* robust recovery of sparse vectors in the context of compressed sensing, whereas the new CLOT formulation does so. Also, like EN but unlike LASSO, the CLOT formulation achieves the grouping effect, wherein coefficients of highly correlated columns of the measurement (or design) matrix are assigned roughly comparable values. It

c	$\hat{x}_{\text{CLOT}}$	$\hat{x}_{\text{EN}}$
0	0.8147	0.8147
	0.9058	0.9058
	0.1270	0.1270
1	8.1472	8.1472
	9.0579	9.0579
	1.2699	1.2699
2	81.4724	24.1502
	90.5792	31.8304
	12.6987	4.7944
3	814.7236	111.2433
	905.7918	132.8940
	126.9868	15.3818
4	8.1472×10^3	NaN
	9.0579×10^3	NaN
	1.2699×10^3	NaN

Table 1: Comparison of solutions of the CLOT and EN algorithms as the unknown vector is scaled by successive powers of 10. The CLOT output simply scales by the same factor, whereas the EN output begins to be incorrect already when the scale factor is 100. For a scale factor of 10^4, the algorithm fails.

is noteworthy that LASSO does not have the grouping effect and EN (as shown here) does not achieve robust sparse recovery. Therefore the CLOT formulation combines the best features of both LASSO (robust sparse recovery) and EN (grouping effect).

The CLOT formulation is a special case of another one called SGL (Sparse Group LASSO) which was introduced into the literature previously, but without any analysis of either the grouping effect or robust sparse recovery (Simon et al., 2013). It is shown here that SGL achieves robust sparse recovery, and also achieves a version of the grouping effect in that coefficients of highly correlated columns of the measurement (or design) matrix are assigned roughly comparable values, *if the columns belong to the same group*.

There are several papers in the literature that discuss LASSO-like formulations for group sparsity; some of these are discussed here. First, there is a companion paper by a subset of the present authors (Ahsen and Vidyasagar, 2016), which studies the problem of robust sparse recovery with SGL-like formulations, *but with restrictions on the group size*. In contrast, in the present paper, there is no such restriction, which is why the results derived here for the SGL formulation can be directly applied to the CLOT formulation. Second, for the case where several columns of the matrix A are highly correlated, (Bühlmann et al., 2013) suggests a two-stage process whereby first correlated columns are clustered, and second, a variant of LASSO is applied. In a discussion of this paper, namely (Bien and Wegkamp, 2013), all the LASSO variants together with EN are run on various test data. For the purposes of the present discussion, the salient observation is that the EN formulation performed roughly as well – no better and no worse – compared almost all the

LASSO variants. Finally, in (van de Geer, 2014), a general theory is presented whereby the fully decomposable ℓ_1-norm is replaced by a weakly decomposable norm, and oracle bounds are derived for $\|\hat{x} - x\|$ provided that the index set $[n]$ is divided into an "allowed set" and its complement. There is also some discussion of overlapping group decompositions. Specifically, when an element of the index set $[n]$ appears in two groups, the corresponding column of A is simply replicated to remove the overlap. However, if two columns of A are identical (and normalized), then the RIP constant δ_2 would equal zero, as would δ_k for $k \geq 2$. Therefore, if the case of overlapping groups is handled in this manner, then any analysis based on RIP would be infructuous. The above discussion is quite cursory, and the reader may consult these references for fuller details.

It would be worthwhile to study the behavior of SGL with overlapping groups. There are variants of SGL with overlapping groups, provided they satisfy some additional constraints; see (Jenetton et al., 2011; Obozinski et al., 2011) for example. However, in a companion paper (Ahsen and Vidyasagar, 2016), it is shown that the assumptions of (Jenetton et al., 2011; Obozinski et al., 2011) still enforce a nonoverlap constraint, but in a nonobvious fashion. As pointed out in the previous paragraph, the approach of introducing duplicate columns into A to eliminate overlap would render any analysis based on RIP impossible. Thus a suitable approach remains to be discovered.

Acknowledgments

The authors thank two anonymous reviewers and the Handling Editor for their careful reading of earlier versions of this paper and many helpful comments. Theorem 5 and its proof were supplied by one of the reviewers. This research was supported by the National Science Foundation under Award #ECCS-1306630 and by the Cancer Prevention and Research Institute of Texas (CPRIT) under Award No. RP140517, and by the Department of Science and Technology, Government of India.

Appendix A: Proof of Theorem 10

Proof Hereafter we write z^i instead of z_{G_i} in the interests of brevity.

Define $h = \hat{x} - x \in \mathbb{R}^n$. From the definition of the estimate, we have that

$$\|x\|_{SGL,\mu} \geq \|\hat{x}\|_{SGL,\mu} = \|x + h\|_{SGL,\mu}$$

From the definition of the SGL norm, this expands to

$$(1-\mu)\|x\|_1 + \mu \sum_{j=1}^{g} \|x^j\|_2 \geq (1-\mu)\|x+h\|_1 + \mu \sum_{j=1}^{g} \|x^j + h^j\|_2.$$

This can be rearranged as

$$(1-\mu)(\|x\|_1 - \|x+h\|_1) + \mu \sum_{j=1}^{g} (\|x^j\|_2 - \|(x^j + h^j\|_2) \geq 0. \tag{40}$$

We will work separately on each of the two terms separately. First, by the triangle inequality, we have that

$$\|x^j + h^j\|_2 \geq \|x^j\|_2 - \|h^j\|_2, \text{ or } \|h^j\|_2 \geq \|x^j\|_2 - \|x^j + h^j\|_2.$$

As a consequence,

$$\sum_{j=1}^{g} \|h^j\|_2 \geq \sum_{j=1}^{g} [\|x^j\|_2 - \|x^j + h^j\|_2]$$

From Schwarz' inequality, we get

$$\sum_{j=1}^{g} \|h^j\|_2 \leq \sqrt{g}\|h\|_2 \leq \sqrt{g}\|h\|_1 \leq \sqrt{g}(\|h_S\|_1 + \|h_{S^c}\|_1),$$

for any $S \subseteq [n]$. Combining everything gives

$$\sum_{j=1}^{g} [\|x^j\|_2 - \|x^j + h^j\|_2] \leq \sqrt{g}(\|h_S\|_1 + \|h_{S^c}\|_1), \tag{41}$$

for any subset $S \subseteq [n]$. Second, for any subset $S \subseteq [n]$, the decomposability of $\|\cdot\|_1$ implies that

$$\|x\|_1 = \|x_S\| + \|x_{S^c}\|_1,$$

while the triangle inequality implies that

$$\begin{aligned} \|x+h\|_1 &= \|x_S + h_S\|_1 + \|x_{S^c} + h_{S^c}\|_1 \\ &\geq \|x_S\|_1 - \|h_S\|_1 + \|h_{S^c}\|_1 - \|x_{S^c}\|_1. \end{aligned}$$

Therefore

$$\|x\|_1 - \|x+h\|_1 \leq \|h_S\|_1 - \|h_{S^c}\|_1 + 2\|x_{S^c}\|_1. \tag{42}$$

If we now choose S to be the set corresponding to the k largest elements of x by magnitude, then

$$\|x_{S^c}\|_1 = \sigma_k(x, \|\cdot\|_1) =: \sigma_k.$$

With this choice of S, (42) becomes

$$\|x\|_1 - \|x+h\|_1 \leq \|h_S\|_1 - \|h_{S^c}\|_1 + 2\sigma_k. \tag{43}$$

Substituting the bounds (41) and (42) into (40) gives

$$0 \leq \mu\sqrt{g}(\|h_S\|_1 + \|h_{S^c}\|_1) + (1-\mu)(\|h_S\|_1 - \|h_{S^c}\|_1 + 2\sigma_k).$$

Now recall the definition of the constant γ from (33). Using this definition, the above inequality can be rearranged as

$$0 \leq \gamma(\|h_S\|_1 + \|h_{S^c}\|_1) + (\|h_S\|_1 - \|h_{S^c}\|_1 + 2\sigma_k).$$

and equivalently as

$$(1-\gamma)\|h_{S^c}\|_1 - (1+\gamma)\|h_S\|_1 \leq 2\sigma_k. \tag{44}$$

This is the first of two equations that we need.

Now we derive the second equation. From Theorem 9, we know that the matrix A satisfies the l_2-robust null space property, namely (26). An application of Schwarz' inequality shows that

$$\|h_S\|_1 \leq \rho\|h_{S^c}\|_1 + \tau\|Ah\|_2.$$

However, because both x and $\hat{x}$ are feasible for the optimization problem in (24), we get

$$\|Ah\|_2 = \|A\hat{x} + \eta - (Ax + \eta)\|_2 \leq 2\epsilon.$$

Substituting this bound for $\|Ah\|_2$ gives us the second equation we need, namely

$$\|h_S\|_1 \leq \rho\|h_{S^c}\|_1 + 2\tau\epsilon,$$

or equivalently

$$-\rho\|h_{S^c}\|_1 + \|h_S\|_1 \leq 2\tau\epsilon. \tag{45}$$

The two inequalities (44) and (45) can be written compactly as

$$M\begin{bmatrix}\|h_{S^c}\|_1 \\ \|h_S\|_1\end{bmatrix} \leq \left\{\begin{bmatrix}2\\0\end{bmatrix}\sigma_k + \begin{bmatrix}0\\2\tau\end{bmatrix}\epsilon\right\}, \tag{46}$$

where the coefficient matrix M is given by

$$M = \begin{bmatrix}(1-\gamma) & -(1+\gamma)\\ -\rho & 1\end{bmatrix}$$

The matrix M has positive diagonal elements (recall that $\gamma < 1$), and negative off-diagonal elements. Therefore, if $\det(M) > 0$, then every element of M^{-1} is positive, in which we can multiply both sides of (46) by M^{-1}. Now

$$\det(M) = 1 - \gamma - \rho(1+\gamma) > 0 \iff \rho < \frac{1-\gamma}{1+\gamma}.$$

Recall the definition of γ from (33). Now routine algebra shows that

$$\rho < \frac{1-\gamma}{1+\gamma} \iff \rho < \frac{1-\mu\sqrt{g}}{1+\mu\sqrt{g}} \iff \mu < \frac{1-\rho}{\sqrt{g}(1+\rho)},$$

which is precisely (32). Thus we can multiply both sides of (46) by M^{-1}, which gives

$$\begin{bmatrix}\|h_{S^c}\|_1 \\ \|h_S\|_1\end{bmatrix} \leq \frac{1}{\det(M)}\begin{bmatrix}1 & (1+\gamma)\\ \rho & (1-\gamma)\end{bmatrix}\left\{\begin{bmatrix}2\\0\end{bmatrix}\sigma_k + \begin{bmatrix}0\\2\tau\end{bmatrix}\epsilon\right\}.$$

Clearing out the matrix multiplication gives

$$\begin{bmatrix}\|h_{S^c}\|_1 \\ \|h_S\|_1\end{bmatrix} \leq \frac{1}{\det(M)}\left\{\begin{bmatrix}2\\2\rho\end{bmatrix}\sigma_k + \begin{bmatrix}2\tau(1+\gamma)\\2\tau(1-\gamma)\end{bmatrix}\epsilon\right\}. \tag{47}$$

Now the triangle inequality states that

$$\|h\|_1 \leq \|h_{S^c}\|_1 + \|h_S\|_1 = \begin{bmatrix}1 & 1\end{bmatrix}\begin{bmatrix}\|h_{S^c}\|_1 \\ \|h_S\|_1\end{bmatrix}$$

Substituting from (47) gives

$$\|h\|_1 \leq \frac{1}{\det(M)}[2(1+\rho)\sigma_k + 4\tau\epsilon].$$

By substituting that $\det(M) = (1-\gamma) - (1+\gamma)\rho$, we get (34) with the constants as defined in (36).

To prove (35), suppose $p \in (1, 2]$. This part of the proof closely follows that of (Foucart and Rauhut, 2013, Theorem 4.22), except that we provide explicit values for the constants. Let Λ_0 denote the index set of the k largest components of h by magnitude. Then

$$\|h\|_p \leq \|h_{\Lambda_0}\|_p + \|h_{\Lambda_0^c}\|_p.$$

We will bound each term separately. First, by (Foucart and Rauhut, 2013, Theorem 2.5) and (34), we get

$$\|h_{\Lambda_0^c}\|_p \leq \frac{1}{k^{1-1/p}}\|h\|_1 \leq \frac{1}{k^{1-1/p}}(C\sigma_k + D\epsilon). \tag{48}$$

Now we apply in succession Hölder's inequality, the robust null space property, the fact that $\|Ah\|_2 \leq 2\epsilon$, and (34). This gives

$$\begin{aligned}\|h_{\Lambda_0}\|_p &\leq k^{1/p-1/2}\|h_{\Lambda_0}\|_2\\ &\leq \frac{k^{1/p-1/2}}{\sqrt{k}}[\rho\|h_{\Lambda_0^c}\|_1 + \tau\|Ah\|_2]\\ &\leq \frac{1}{k^{1-1/p}}[\rho(C\sigma_k + D\epsilon) + 2\tau\epsilon]\\ &= \frac{1}{k^{1-1/p}}[\rho C\sigma_k + (\rho D + 2\tau)\epsilon]. \end{aligned}\tag{49}$$

Adding (48) and (49) gives (35).

■

References

Mehmet Eren Ahsen and Mathukumalli Vidyasagar. Error bounds for compressed sensing algorithms with group sparsity: A unified approach. *Applied and Computational Harmonic Analysis*, page to appear, 2016.

Alexandre Belloni, Victor Chernozhukov, and Lie Wang. Pivotal estimation via square-root Lasso in nonparametric regression. *The Annals of Statistics*, 42(2):757–788, 2014.

Jacob Bien and Marten Wegkamp. Discussion of correlated variables in regression: Clustering and sparse estimation". *Journal of Statistical Planning and Inference*, 143(11): 1859–1862, November 2013.

Peter Bühlmann, Philipp Rütimann, Sara van de Geer, and Cun-Hui Zhang. Correlated variables in regression: Clustering and sparse estimation. *Journal of Statistical Planning and Inference*, 143(11):1835–1858, November 2013.

T. Cai and A. Zhang. Sparse representation of a polytope and recovery of sparse signals and low-rank matrices. *IEEE Transactions on Information Theory*, 60(1):122–132, 2014.

Emmanuel Candès. The restricted isometry property and its implications for compresed sensing. *Comptes rendus de l'Académie des Sciences, Série I*, 346:589–592, 2008.

Emmanuel J. Candès and Terence Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, December 2005.

Emmanuel J. Candès, J. Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications in Pure and Applied Mathematics*, 59(8): 1207–1223, August 2006.

Albert Cohen, Wolfgang, Dahmen, and Ronald DeVore. Compressed sensing and best k-term approximation. *Journal of the American Mathematical Society*, 22(1):211–231, January 2009.

Mark A. Davenport, Marco F. Duarte, Yonina C. Eldar, and Gitta Kutyniok. Introduction to compressed sensing. In Yonina C. Eldar and Gitta Kutyniok, editors, *Compressed Sensing: Theory and Applications*, pages 1–68. Cambridge University Press, Cambridge, UK, 2012.

Ronald DeVore. Deterministic construction of compressed sensing matrices. *Journal of Complexity*, 23:918–925, 2007.

D. L. Donoho. For most large underdetermined systems of linear equations, the minimal ℓ_1-norm solution is also the sparsest solution. *Communications in Pure and Applied Mathematics*, 59(6):797–829, 2006.

Simon Foucart and Holger Rauhut. *A Mathematical Introduction to Compressive Sensing*. Springer-Verlag, 2013.

A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

R. Jenetton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for hierarchical sparse coding. *Journal of Machine Learning Research*, 12:2297–2334, 2011.

B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal on Computing*, 24:227–234, 1995.

S. Negabhan, P. Ravikumar, M. J. Wainwright, and B. Yu. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. *Statistical Science*, 27(4):538–557, December 2012.

Guillaume Obozinski, Laurent Jacob, and Jean-Philippe Vert. Group lasso with overlaps: The latest group lasso approach. *arxiv*, page 1110.0413, 2011.

Michael R. Osborne, Brett Presnell, and Berwin A. Turlach. On the LASSO and its dual. *Journal of Computational and Graphical Statistics*, 9:319–337, 2000.

Shashank Ranjan and Mathukumalli Vidyasagar. Tight performance bounds for compressed sensing with group sparsity. *arXiv*, page 1606.05889v1, 2016.

Noah Simon, Jerome Friedman, Trevor Hastie, and Rob Tibshirani. A sparse group lasso. *Journal of Computational and Graphical Statistics*, 22(2):231–245, 2013.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1), 1996.

A. N. Tikhonov. On the stability of inverse problems. *Doklady Akademii Nauk SSSR*, 39(5): 195–198, 1943.

Sara van de Geer. Weakly decomposable regularization penalties and structured sparsity. *Scandinavian Journal of Statistics*, 41:72–86, 2014.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society, Series B*, 68:49–67, 2006.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society B*, 67:301–320, 2005.

Journal of Machine Learning Research 18 (2017) 1-35 Submitted 9/15; Revised 4/17; Published 6/17

On the Consistency of Ordinal Regression Methods

Fabian Pedregosa f@bianp.net
INRIA
Département d'informatique de l'ENS, École normale supérieure, CNRS, PSL Research University
Paris, France

Francis Bach francis.bach@ens.fr
INRIA
Département d'informatique de l'ENS, École normale supérieure, CNRS, PSL Research University
Paris, France

Alexandre Gramfort alexandre.gramfort@inria.fr
LTCI, Télécom ParisTech, Université Paris-Saclay
INRIA, Université Paris-Saclay
Saclay, France

Editor: Tong Zhang

Abstract

Many of the ordinal regression models that have been proposed in the literature can be seen as methods that minimize a convex surrogate of the zero-one, absolute, or squared loss functions. A key property that allows to study the statistical implications of such approximations is that of *Fisher consistency*. Fisher consistency is a desirable property for surrogate loss functions and implies that in the population setting, i.e., if the probability distribution that generates the data were available, then optimization of the surrogate would yield the best possible model. In this paper we will characterize the Fisher consistency of a rich family of surrogate loss functions used in the context of ordinal regression, including support vector ordinal regression, ORBoosting and least absolute deviation. We will see that, for a family of surrogate loss functions that subsumes support vector ordinal regression and ORBoosting, consistency can be fully characterized by the derivative of a real-valued function at zero, as happens for convex margin-based surrogates in binary classification. We also derive excess risk bounds for a surrogate of the absolute error that generalize existing risk bounds for binary classification. Finally, our analysis suggests a novel surrogate of the squared error loss. We compare this novel surrogate with competing approaches on 9 different datasets. Our method shows to be highly competitive in practice, outperforming the least squares loss on 7 out of 9 datasets.

Keywords: Fisher consistency, ordinal regression, calibration, surrogate loss, excess risk bound.

1. Introduction

In ordinal regression the goal is to learn a rule to predict labels from an ordinal scale, i.e., labels from a discrete but ordered set. This arises often when the target variable consists of human generated ratings, such as ("do-not-bother" $\prec$ "only-if-you-must" $\prec$ "good" $\prec$ "very-good" $\prec$ "run-to-see") in movie ratings (Crammer and Singer, 2001), ("absent" $\prec$ "mild" $\prec$ "severe") for the symptoms of a physical disease (Armstrong and Sloan, 1989) and the NRS-11 numeric rating scale for clinical pain measurement (Hartrick et al., 2003). Ordinal regression models have been successfully applied to fields as diverse as econometrics (Greene, 1997), epidemiology (Ananth and Kleinbaum, 1997), fMRI-based brain decoding (Doyle et al., 2013) and collaborative filtering (Rennie and Srebro, 2005).

Ordinal regression shares properties–and yet is fundamentally different–from both multiclass classification and regression. As in the multiclass classification setting, the target variables consist of discrete values, and as in the regression setting (but unlike the multiclass setting) there is a meaningful order between the classes. If we think of the symptoms of a physical disease, it is clear that if the true label is "severe" it is preferable to predict "mild" than "absent". Ordinal regression models formalize this notion of order by ensuring that predictions farther from the true label incur a greater penalty than those closer to the true label.

The ordinal regression approach also shares properties with the learning-to-rank problem (Liu, 2011), in which the goal is to predict the relative order of a sequence of instances. Hence, this approach focuses on predicting a relative order while ordinal regression focuses on predicting a label for each instance. In this sense, it is possible for a ranking model (but not for an ordinal regression one) that predicts the wrong labels to incur no loss at all, as long as the relative order of those labels are correct, e.g. if the prediction is given by the true label plus an additive bias. Although ordinal regression and ranking are different problems, the distinction between both has not always been clear, generating some confusion. For example, in the past some methods presented with the word "ranking" in the title would be considered today ordinal regression methods (Crammer and Singer, 2001; Shashua and Levin, 2003; Crammer and Singer, 2005) and likewise some of the first pairwise ranking methods (Herbrich et al., 1999) featured the word ordinal regression in the title.

Despite its widespread applicability, there exists a relative paucity in the understanding of the theoretical properties behind ordinal regression methods, at least compared to that of binary and multiclass classification. One such example is the notion of *Fisher consistency*, which relates the minimization of a given loss to the minimization of a surrogate with better computational properties. The importance of this property stems from the fact that many supervised learning methods, such as support vector machines, boosting and logistic regression for binary classification, can be seen as methods that minimize a convex surrogate on the 0-1 loss. Such results have emerged in recent years for classification (Bartlett et al., 2003; Zhang, 2004a; Tewari and Bartlett, 2007), ranking (Duchi et al., 2010; Calauzenes et al., 2012), structured prediction (Ciliberto et al., 2016; Osokin et al., 2017) and multiclass classification with an arbitrary loss function (Ramaswamy and Agarwal, 2012, 2016), a setting that subsumes ordinal regression. Despite these recent progress, the Fisher consistency of most surrogates used within the context of ordinal regression remains elusive. The aim of this paper is to bridge the gap by providing an analysis of Fisher consistency for a wide family of ordinal regression methods that parallels the ones that already exist for other multiclass classification and ranking.

Notation. Through the paper, we will use k to denote the number of classes (i.e., labels) in the learning problem. We will denote by $\mathcal{S}$ the subset of $\mathbb{R}^{k-1}$ for which the

©2017 Pedregosa, Bach and Gramfort.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-495.html.

components are non-decreasing, that is,

$$\mathcal{S} := \left\{ \alpha : \alpha \in \mathbb{R}^{k-1} \text{ and } \alpha_i \leq \alpha_{i+1} \text{ for } 1 \leq i \leq k-2 \right\} \quad .$$

Δ^p denotes the p-dimensional simplex, defined as

$$\Delta^p := \left\{ x \in \mathbb{R}^p : x_i \geq 0 \text{ and } \sum_{i=1}^{p} x_i = 1 \right\} \quad .$$

Following Knuth (1992) we use the Iverson bracket $[\![\cdot]\!]$ as

$$[\![q]\!] := \begin{cases} 1 \text{ if q is true} \\ 0 \text{ otherwise} \end{cases} \quad .$$

We will also make reference to loss functions commonly used in binary classification. These are the hinge loss ($\varphi(t) = \max(1-t, 0)$), the squared hinge loss ($\varphi(t) = \max(1-t, 0)^2$), the logistic loss ($\varphi(t) = \log(1+e^{-t})$), exponential loss ($\varphi(t) = e^{-t}$) and the squared loss ($\varphi(t) = (1-t)^2$).

1.1 Problem setting

Here we present the formalism that we will be using throughout the paper. Let $(\mathcal{X}, \mathcal{A})$ be a measurable space. Let (X, Y) be two random variables with joint probability distribution P, where X takes its values in $\mathcal{X}$ and Y is a random label taking values in a finite set of k *ordered categories* that we will denote $\mathcal{Y} = \{1, \ldots, k\}$. In the ordinal regression problem, we are given a set of n observations $\{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ drawn i.i.d. from $X \times Y$ and the goal is to learn from the observations a measurable mapping called a *decision function* $f : \mathcal{X} \to \mathcal{S} \subseteq \mathbb{R}^{k-1}$ so that the *risk* given below is as small as possible:

$$\mathcal{L}(f) := \mathbb{E}(\ell(Y, f(X))) \quad , \tag{1}$$

where $\ell : \mathcal{Y} \times \mathcal{S}$ is a *loss function* that measures the disagreement between the true label and the prediction. For ease of optimization, the decision function has its image in a subset of $\mathbb{R}^{k-1}$, and the function that converts an element of $\mathcal{S}$ into a class label is called a *prediction function*. The prediction function that we will consider through the paper is given for $\alpha \in \mathcal{S}$ by the number of coordinates below zero plus one, that is,

$$\text{pred}(\alpha) := 1 + \sum_{i=1}^{k-1} [\![\alpha_i < 0]\!] \quad . \tag{2}$$

Note that for the case of two classes $\mathcal{Y} = \{1, 2\}$, the decision function is real-valued and the prediction defaults the common binary classification rule in which prediction depends on the sign of this decision function.

Different loss functions can be used within the context of ordinal regression. The most commonly used one is the absolute error, which measures the absolute difference between the predicted and true labels. For $\alpha \in \mathcal{S}$, this is defined as

$$\ell(y, \alpha) := \left| y - \text{pred}(\alpha) \right| \quad . \tag{3}$$

The absolute error loss is so ubiquitous in ordinal regression that some authors refer to it simply as *the* ordinal regression loss (Agarwal, 2008; Ramaswamy and Agarwal, 2012). For this reason we give special emphasis on this loss. However, we will also describe methods that minimize the 0-1 loss (i.e., the classification error) and in Section 3.4 we will see how some results can be generalized beyond these and to general loss functions that verify a certain admissibility criterion.

In order to find the decision function with minimal risk it might seem appropriate to minimize Eq. (1). However, this is not feasible in practice for two reasons. First, the probability distribution P is unknown and the risk must be minimized approximately based on the observations. Second, ℓ is typically discontinuous in its second argument, hence the empirical approximation to the risk is difficult to optimize and can lead to an NP-hard problem (Feldman et al., 2012; Ben-David et al., 2003)[1]. It is therefore common to approximate ℓ by a function $\psi : \mathcal{Y} \times \mathcal{S} \to \mathbb{R}$, called a *surrogate loss function*, which has better computational properties. The goal becomes then to find the decision function that instead minimizes the *surrogate risk*, defined as

$$\mathcal{A}(f) := \mathbb{E}(\psi(Y, f(X))) \ . \tag{4}$$

We are interested by the statistical implications of such approximation. Assuming that we have full knowledge of the probability distribution that generates the data P, what are the consequences of optimizing a convex surrogate of the risk instead of the true risk?

The main property that we will study in order to answer this question is that of *Fisher consistency*. Fisher consistency is a desirable property for surrogate loss functions (Lin, 2004) and implies that in the population setting, i.e., if the probability distribution P were available, then optimization of the surrogate would yield a function with minimal risk. From a computational point of view, this implies that the minimization of the surrogate risk, which is usually a convex optimization problem and hence easier to solve than the minimization of the risk, does not penalize the quality (always in the population setting) of the obtained solution.

We will use the following notation for the optimal risk and optimal surrogate risk:

$$\mathcal{L}^* := \inf_f \mathcal{L}(f) \quad \text{and} \quad \mathcal{A}^* := \inf_f \mathcal{A}(f) \quad ,$$

where the minimization is done over all measurable functions $\mathcal{X} \to \mathcal{S}$. $\mathcal{L}^*$ is sometimes referred to as the *Bayes risk*, and a decision function (not necessarily unique) that minimizes the risk is called a *Bayes decision function*.

We will now give a precise definition of Fisher consistency. This notion originates from a classical parameter estimation setting. Suppose that an estimator T of some parameter θ is defined as a functional of the empirical distribution P_n. We denote it $T(P_n)$. The estimator is said to be Fisher consistent if its population analog, $T(P)$, coincides with the parameter θ. Adapting this notion to the context of risk minimization (in which the optimal risk is the parameter to estimate) yields the following definition, adapted from Lin (2004) to an arbitrary loss function ℓ:

1. Note that binary classification can be seen as a particular case of ordinal regression.

Definition 1 ***(Fisher consistency)*** *Given a surrogate loss function $\psi : \mathcal{Y} \times \mathcal{S} \to \mathbb{R}$, we will say that the surrogate loss function ψ is consistent with respect to the loss $\ell : \mathcal{Y} \times \mathcal{S} \to \mathbb{R}$ if for every probability distribution over $X \times Y$ it is verified that every minimizer f of the surrogate risk reaches Bayes optimal risk, that is,*

$$\mathcal{A}(f) = \mathcal{A}^* \implies \mathcal{L}(f) = \mathcal{L}^* \quad .$$

For some surrogates we will be able to derive not only Fisher consistency, but also *excess risk bounds*. These are bounds of the form

$$\gamma(\mathcal{L}(f) - \mathcal{L}^*) \leq \mathcal{A}(f) - \mathcal{A}^* \quad ,$$

for some real-valued function γ with $\gamma(0) = 0$. These inequalities not only imply Fisher consistency, but also allow to bound the excess risk by the excess in surrogate risk. These inequalities play an important role in different areas of learning theory, as they can be used for example to obtain rates of convergence (Bartlett et al., 2003) and oracle inequalities (Boucheron et al., 2005).

1.2 Full and conditional risk

The above definition of Fisher consistency is often replaced by a point-wise version that is easier to verify in practice. Two key ingredients of this characterization are the notions of *conditional risk* and *surrogate conditional risk* that we will now define. These are denoted by L and A respectively, and defined for any $\alpha \in \mathcal{S}$, $p \in \Delta^k$ by

$$L(\alpha, p) := \sum_{i=1}^{k} p_i \ell(i, \alpha) \quad \text{and} \quad A(\alpha, p) := \sum_{i=1}^{k} p_i \psi(i, \alpha) \quad . \tag{5}$$

The full and conditional risk are then related by the equations

$$\begin{aligned} \mathcal{L}(f) &= \mathbb{E}_{X\times Y}(\ell(Y, f(X))) = \mathbb{E}_X \mathbb{E}_{Y|X}(\ell(Y, f(X))) = \mathbb{E}_X(L(f(X), \eta(X))) \\ \mathcal{A}(f) &= \mathbb{E}_{X\times Y}(\psi(Y, f(X))) = \mathbb{E}_X \mathbb{E}_{Y|X}(\psi(Y, f(X))) = \mathbb{E}_X(A(f(X), \eta(X))) \quad , \end{aligned}$$

where $\eta : \mathcal{X} \to \Delta^k$ is the vector of conditional probabilities given by $\eta_i(x) = P(y = i|X = x)$. As for the full risk, we will denote by L^*, A^* the infimum of its value for a given $p \in \Delta^k$, i.e.,

$$L^*(p) = \inf_{\alpha \in \mathcal{S}} L(\alpha, p) \quad \text{and} \quad A^*(p) = \inf_{\alpha \in \mathcal{S}} A(\alpha, p) \quad .$$

When the risk infimum over functions that can be defined independently at every $x \in \mathcal{X}$, it is possible to relate the minimization of the risk with that of the conditional risk since

$$\begin{aligned} \inf_f \mathcal{L}(f) &= \inf_f \mathbb{E}_{X\times Y}\left(\ell(Y, f(X))\right) = \mathbb{E}_X\left[\inf_f \mathbb{E}_{Y|X}(\ell(Y, f(X)))\right] \\ &= \mathbb{E}_X\left[\inf_\alpha L(\alpha, \eta(X))\right] \quad . \end{aligned} \tag{6}$$

This equation implies that the minimal risk can be achieved by minimizing pointwise the conditional risk $L(\cdot)$, which–in general–will be easier that direct minimization of the full risk.

The condition for this, i.e., that the functions be estimated independently at every sample point, is verified by the set of measurable functions from the sample space into a subset of $\mathbb{R}^k$ (in this case $\mathcal{S}$), which is the typical setting in studies of Fisher consistency. However, this is no longer true when inter-observation constraints are enforced (e.g. smoothness). As is common in studies of Fisher consistency, we will suppose that the function class verifies the property of Eq. (6) and we will discuss in Section 4 an important family of functions in which this requisite is not met.

We will now present a characterization of Fisher consistency based on the pointwise risk which we will use throughout the paper. Equivalent forms of this characterization have appeared under a variety of names in the literature, such as classification calibration (Bartlett et al., 2003; Ramaswamy and Agarwal, 2012), infinite sample consistency (Zhang, 2004b) and proper surrogates (Buja et al., 2005; Gneiting and Raftery, 2007).

Lemma 2 (Pointwise characterization of Fisher consistency) *Let A and L be defined as in Eq (5). Then ψ is Fisher consistent with respect to ℓ if and only if for all $p \in \Delta^k$ it is verified that*

$$A(\alpha, p) = A^*(p) \implies L(\alpha, p) = L^*(p) \quad . \tag{7}$$

Proof Let $\mathcal{L}$ and $\mathcal{A}$ denote the expected value of ℓ and ψ, as defined in Equations (1) and (4) respectively.

($\Longleftarrow$) We prove that Eq. (7) implies Fisher consistency. Let f be such that $\mathcal{A}(f) = \mathcal{A}^*$. Then it is verified that

$$\mathcal{A}(f) - \mathcal{A}^* = \mathbb{E}_X(A(f(X), \eta(X)) - A^*(\eta(X))) = 0 \quad .$$

The value inside the expectation is non-negative by definition of A^*. Since this is verified for all probability distributions over $X \times Y$, then it must be true that $A(f(x), \eta(x)) = A^*(\eta(x))$ for all $x \in \mathcal{X}$. By assumption $L(f(X), \eta(X)) = L^*(\eta(X))$. Hence the excess risk verifies

$$\mathcal{L}(f) - \mathcal{L}^* = \mathbb{E}_X(L(f(X), \eta(X)) - L^*(\eta(X))) = \mathbb{E}(0) = 0 \quad .$$

and so ψ is Fisher consistent with respect to ℓ.

($\Longrightarrow$) We prove that Fisher consistency implies Eq. (7). We do so by contradiction: first suppose that there exists a surrogate that is Fisher consistent but Eq. (7) is not verified and arrive to a contradiction. If Eq. (7) is not verified then there exists $\tilde{\alpha} \in \mathcal{S}$ and $\tilde{p} \in \Delta^k$ such that

$$A(\tilde{\alpha}, \tilde{p}) = A^*(\tilde{p}) \text{ and } L(\tilde{\alpha}, \tilde{p}) > L^*(\tilde{p}) \quad .$$

We now construct a probability distribution (X, Y) such that the Fisher consistency characterization is not verified in order to arrive to a contradiction. For this, consider the probability distribution P such that $\eta(x) = \tilde{p}$ for all $x \in \mathcal{X}$. Consider also $f : \mathcal{X} \to \mathcal{S}$, the mapping that is constantly $\tilde{\alpha}$. Then it is verified that

$$\mathcal{A}(f) - \mathcal{A}^* = \mathbb{E}_X(A(f(X), \eta(X)) - A^*(\eta(X))) = \mathbb{E}_X(A(\tilde{\alpha}, \tilde{p}) - A^*(\tilde{p})) = 0 \quad ,$$

and so $\mathcal{A}(f) = \mathcal{A}^*$. Likewise, the excess risk verifies

$$\mathcal{L}(f) - \mathcal{L}^* = \mathbb{E}_X(L(f(X), \eta(X)) - L^*(\eta(X))) = \mathbb{E}_X(L(\tilde{\alpha}, \tilde{p}) - L^*(\tilde{p})) > 0$$

and so ψ cannot be Fisher consistent with respect to ℓ. This is a contradiction, and concludes the proof.

■

1.3 Summary of main results

The main contribution of this paper is to characterize the Fisher consistency of a wide family of surrogate loss functions used for the task of ordinal regression. Contrary to known results for multiclass classification and ranking, where One-vs-All and RankSVM have been proven to be inconsistent, in the ordinal regression setting common surrogates such as ORSVM and proportional odds will be proven to be Fisher consistent. One of the most surprising results of this paper is that for a particular class of surrogates that verify a *decomposability* property, it is possible to provide a characterization of Fisher consistency and excess risk bounds that generalize those known for convex margin-based surrogates (loss functions of the form $\varphi(Yf(X))$) in binary classification.

We will introduce the surrogate loss functions that we consider in Section 2. These will be divided between surrogates of the absolute error and surrogate of the 0-1 loss. We organize their study as follows:

- In Sections 3.1 and 3.2 we characterize the **Fisher consistency for surrogates of the absolute and squared error**. The surrogates that we consider in this section are the all threshold (AT), the cumulative link (CL), the least absolute deviation (LAD) and the least squares (LS). Besides Fisher consistency, a decomposability of the AT loss will allow us to provide excess risk bounds for this surrogate.

- In Section 3.3 we characterize the **Fisher consistency of the surrogates of the 0-1 loss**. For this loss, denoted immediate threshold (IT), its Fisher consistency will depend on the derivative at zero of a real-valued convex function.

- In Section 3.4 we **construct a surrogate for an arbitrary loss function** that verifies an admissibility condition. We name this surrogate generalized all threshold (GAT). This loss function generalizes the AT and IT loss functions introduced earlier. We will characterize the Fisher consistency of this surrogate.

- Turning back to one of the topics mentioned in the introduction, we discuss in Section 4 the **implications of inter-observational constraints in Fisher consistency**. Following Shi et al. (2015), we define a restricted notion of consistency known as $\mathcal{F}$-consistency of parametric consistency and give sufficient conditions for the $\mathcal{F}$-consistency of two surrogates.

- In Section 5 we **examine the empirical performance of a novel surrogate**. This novel surrogate is a particular instance of the GAT loss function introduced in Section 3.4 when considering the squared error as evaluation metric. We compare this novel surrogate against a least squares model on 9 different datasets, where the novel surrogate outperforms the least squares estimate on 7 out of the 9 datasets.

1.4 Related work

Fisher consistency of binary and multiclass classification for the zero-one loss has been studied for a variety of surrogate loss functions, see e.g. (Bartlett et al., 2003; Zhang, 2004a; Tewari and Bartlett, 2007; Reid and Williamson, 2010). Some of the results in this paper generalize known results for binary classification to the ordinal regression setting. In particular, Bartlett et al. (2003) provide a characterization of the Fisher consistency for convex margin-based surrogates that we extend to the all threshold (AT) and immediate threshold (IT) family of surrogate loss functions. The excess error bound that we provide for the AT surrogate also generalizes the excess error bound given in (Bartlett et al., 2003, Section 2.3).

Fisher consistency of arbitrary loss functions (a setting that subsumes ordinal regression) has been studied for some surrogates. Lee et al. (2004) proposed a surrogate that can take into account generic loss functions and for which Fisher consistency was proven by Zhang (2004b). In a more general setting, Ramaswamy and Agarwal (2012, 2016) provide necessary and sufficient conditions for a surrogate to be Fisher consistent with respect to an arbitrary loss function. Among other results, they prove consistency of least absolute deviation (LAD) and an ε-insensitive loss with respect to the absolute error for the case of three classes ($k = 3$). In this paper, we extend the proof of consistency for LAD to an arbitrary number of classes. Unlike previous work, we consider the so-called *threshold-based surrogates* (AT, IT and CL), which rank among the most popular ordinal regression loss functions and for which its Fisher consistency has not been studied previously.

Fisher consistency has also been studied in the pairwise ranking setting, where it has been proven (Duchi et al., 2010; Calauzenes et al., 2012) that some models (such as RankSVM) are not consistent. Despite similarities between ranking and ordinal regression, we will see in this paper that most popular ordinal regression models are Fisher consistent under mild conditions.

There are few studies on the theoretical properties of ordinal regression methods. A notable example comes from Agarwal (2008), where the authors study generalization bounds for some ordinal regression algorithms. Some of the surrogate loss functions used by these models (such as the support vector ordinal regression of Chu and Keerthi (2005)) are analyzed in this paper. In that work, the authors outline the study of consistency properties of ordinal regression models as an important question to be addressed in the future.

A related, yet different, notion of consistency is *asymptotic consistency*. A surrogate loss is said to be asymptotically consistent if the minimization of the ψ-risk converges to the optimal risk as the number of samples tends to infinity. It has also been studied in the setting of supervised learning (Stone, 1977; Steinwart, 2002). This paper focuses solely on Fisher consistency, to whom we will refer simply as consistency from now on.

2. Ordinal regression models

We introduce the different ordinal regression models that we will consider within this paper. Considering first the absolute error, we will write this loss as a sum of binary 0-1 loss

functions[2]. This is a key reformulation of the absolute error that we will use throughout the paper. For any $y \in \mathcal{Y}$ and $\alpha \in \mathcal{S}$ we have the following sequence of equivalences

$$\begin{aligned}\ell(y,\alpha) &= \big|y - \text{pred}(\alpha)\big| = \left|y - 1 - \sum_{i=1}^{k-1} [\![\alpha_i < 0]\!]\right| \\ &= \left|y - 1 - \sum_{i=1}^{y-1} [\![\alpha_i < 0]\!] - \sum_{i=y}^{k-1} [\![\alpha_i < 0]\!]\right| \\ &= \left|\sum_{i=1}^{y-1} [\![\alpha_i \geq 0]\!] - \sum_{i=y}^{k-1} [\![\alpha_i < 0]\!]\right| \quad .\end{aligned} \tag{8}$$

If $\alpha_y \geq 0$ then the second summand of the last equation equals zero. Otherwise, if $\alpha_y < 0$, then the first summand equals zero. In either case, we have

$$\ell(y,\alpha) = \sum_{i=1}^{y-1} [\![\alpha_i \geq 0]\!] + \sum_{i=y}^{k-1} [\![\alpha_i < 0]\!] \quad . \tag{9}$$

This expression suggests that a natural surrogate can be constructed by replacing the binary 0-1 loss in the above expression function by a convex surrogate such as the logistic or hinge loss. Denoting by $\varphi : \mathbb{R} \to \mathbb{R}$ such surrogate, we obtain the following loss function that we denote *all threshold (AT)*:

$$\psi_{\text{AT}}(y,\alpha) := \sum_{i=1}^{y-1} \varphi(-\alpha_i) + \sum_{i=y}^{k-1} \varphi(\alpha_i) \quad . \tag{10}$$

This function has appeared under different names in the literature. When φ is the hinge loss, this model is known as support vector ordinal regression with implicit constraints (Chu and Keerthi, 2005) and support vector with sum-of-margins strategy (Shashua and Levin, 2003). When φ is the exponential loss, this model has been described in (Lin and Li, 2006) as ordinal regression boosting with all margins. Finally, Rennie and Srebro (2005) provided a unifying formulation for this approach considering for the hinge, logistic and exponential loss under the name of All-Threshold loss, a name that we will adopt in this paper.

The name *thresholds* comes from the fact that in the aforementioned work, the decision function is of the form $\alpha_i = \theta_i - f(\cdot)$, where $(\theta_1, \ldots, \theta_{k-1})$ is a vector estimated from the data known as the vector of thresholds. We will discuss in Section 4 the implications of such decision function. For the prediction rule to give meaningful results it is important to ensure that the thresholds are ordered, i.e., $\theta_1 \leq \theta_2, \leq \cdots, \leq \theta_{k-1}$ (Chu and Keerthi, 2005). In our setting, we enforce this through the constraint $\alpha \in \mathcal{S}$, hence the importance of restricting the problem to this subset of $\mathbb{R}^{k-1}$.

Another family of surrogate loss functions takes a probabilistic approach and models instead the posterior probability. This is the case of the *cumulative link* models of McCullagh

2. The 0-1 loss, defined as the function that is 1 for negative values and 0 otherwise can be defined in bracket notation as $\ell_{0-1}(t) = [\![\alpha_i \leq 0]\!]$.

(1980). In such models the decision function f is selected to approximate $\sigma(f_i(x)) = P(Y \leq i | X = x)$, where $\sigma : \mathbb{R} \to [0,1]$ is a function referred to as *link function*. Several functions can be used as link function, although the most common ones are the sigmoid function and the Gaussian cumulative distribution. The sigmoid function, i.e., $\sigma(t) = 1/(1+\exp(-t))$, leads to a model sometimes referred as proportional odds (McCullagh, 1980) and cumulative logit (Agresti, 2010), although for naming consistency we will refer to it as *logistic cumulative link*. Another important link function is given by the Gaussian cumulative distribution, $\sigma(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{t} e^{-x^2/2}$, used in the Gaussian process ordinal regression model of Chu and Ghahramani (2004). The cumulative link (CL) loss function is given by its negative likelihood, that is,

$$\psi_{\text{CL}}(y,\alpha) := \begin{cases} -\log(\sigma(\alpha_1)) & \text{if } y = 1 \\ -\log(\sigma(\alpha_y) - \sigma(\alpha_{y-1})) & \text{if } 1 < y < k \\ -\log(1 - \sigma(\alpha_{k-1})) & \text{if } y = k \quad . \end{cases} \tag{11}$$

We will now consider the multiclass 0-1 loss. In this case, the loss will be 1 if the prediction is below or above y (i.e., if $\alpha_{y-1} \geq 0$ or $\alpha_y < 0$) and 0 otherwise. Hence, it is also possible to write the multiclass 0-1 loss as a sum of binary 0-1 loss functions:

$$\ell(y,\alpha) = \begin{cases} [\![\alpha_1 < 0]\!] & \text{if } y = 1 \\ [\![\alpha_{y-1} \geq 0]\!] + [\![\alpha_y < 0]\!] & \text{if } 1 < y < k \\ [\![\alpha_{k-1} \geq 0]\!] & \text{if } y = k \quad . \end{cases}$$

Given this expression, a natural surrogate is given by replacing the binary 0-1 loss by a convex surrogate as the hinge or logistic function. Following Rennie and Srebro (2005), we will refer to this loss function as *immediate threshold (IT)*:

$$\psi_{\text{IT}}(y,\alpha) := \begin{cases} \varphi(\alpha_1) & \text{if } y = 1 \\ \varphi(-\alpha_{y-1}) + \varphi(\alpha_y) & \text{if } 1 < y < k \\ \varphi(-\alpha_{k-1}) & \text{if } y = k \quad . \end{cases} \tag{12}$$

As with the AT surrogate, this loss has appeared under a variety of names in the literature. When φ is the hinge loss, this model is known as support vector ordinal regression with explicit constraints (Chu and Keerthi, 2005) and support vector with fixed-margins strategy (Shashua and Levin, 2003). When φ is the exponential loss, this model has been described by Lin and Li (2006) as ordinal regression boosting with left-right margins. We note that the construction of the AT and IT surrogates are similar, and in fact, we will see in Section 3.4 that both can be seen as a particular instance of a more general family of loss functions.

The aforementioned approaches can be seen as methods that adapt known binary classification methods to the ordinal regression setting. A different approach consists in treating the labels as real values and use regression algorithms to learn a real-valued mapping between the samples and the labels. This ignores the discrete nature of the labels, thus it is

necessary to introduce a prediction function that converts this real value into a label in $\mathcal{Y}$. This prediction function is given by rounding to the closest label (see, e.g., (Kramer et al., 2001) for a discussion of this method using regression trees). This approach is commonly referred to as the *regression-based* approach to ordinal regression. If we are seeking to minimize the absolute error, a popular loss function is to minimize the least absolute deviation (LAD). For any $\beta \in \mathbb{R}$, this is defined as

$$\psi_{\text{LAD}}(y, \beta) := |y - \beta| \quad ,$$

and prediction is then given by rounding β to the closest label. This setting departs from the approaches introduced earlier by using a different prediction function. However, via a simple transformation it is possible to convert this prediction function (rounding to the closest label) to the prediction function that counts the number of non-zero components defined in Eq. (2). For a given $\beta \in \mathbb{R}$, this transformation is given by

$$\alpha_1 = \frac{3}{2} - \beta, \quad \alpha_2 = \frac{5}{2} - \beta, \quad \ldots, \quad \alpha_i = i + \frac{1}{2} - \beta \quad . \tag{13}$$

It is immediate to see that this vector α belongs to $\mathcal{S}$ and

$$\begin{aligned}
\text{pred}(\alpha) &= 1 + \sum_{i=1}^{k-1} [\![i + \frac{1}{2} < \beta]\!] \\
&= \begin{cases} 1 & \text{if } \beta \leq 1 + \frac{1}{2} \\ i & \text{if } i - \frac{1}{2} \leq \beta < i + \frac{1}{2}, 1 < i < k \\ k & \text{if } \beta \geq k - \frac{1}{2} \end{cases} \\
&= \arg\min_{1 \leq i \leq k} |\beta - i| \quad \text{(rounding to the lower label in case of ties)} \quad ,
\end{aligned}$$

hence predicting in the transformed vector α is equivalent to the closest label to β. We will adopt this transformation when considering LAD for convenience, in order to analyze it within the same framework as the rest. With the aforementioned transformation, the least absolute deviation surrogate is given by

$$\psi_{\text{LAD}}(y, \alpha) = \left| y + \alpha_1 - \frac{3}{2} \right| \tag{14}$$

Although the surrogate loss function LAD and the absolute loss of Eq. (3) look very similar, they differ in that the LAD surrogate is convex on α, while the absolute error is not, due to the presence of the discontinuous function $\text{pred}(\cdot)$.

In this section we have introduced some of the most common ordinal regression methods based on the optimization of a convex loss function. These are summarized in Table 1.

3. Consistency results

In this section we present consistency results for different surrogate loss functions. We have organized this section by the different loss functions against which we test for consistency.

Table 1: Surrogate loss functions considered in this paper.

Model	**Loss Function**	**Also known as**
All thresholds (AT)	$\sum_{i=1}^{y-1} \varphi(-\alpha_i) + \sum_{i=y}^{k-1} \varphi(\alpha_i)$	Implicit constraints (Chu and Keerthi, 2005), all margins (Lin and Li, 2006).
Cumulative link (CL)	$-\log(\sigma(\alpha_y) - \sigma(\alpha_{y-1}))$	Proportional odds (McCullagh, 1980), cumulative logit (Agresti, 2010).
Immediate threshold (IT)	$\varphi(-\alpha_{y-1}) + \varphi(\alpha_y)$	Explicit constraints (Chu and Keerthi, 2005), Fixed-margins (Shashua and Levin, 2003)
Least absolute deviation (LAD)	$\lvert y + \alpha_1 - \frac{3}{2} \rvert$	Least absolute error, least absolute residual, Sum of absolute deviations, ℓ_1 regression.
Least squares (LS)	$\left(y + \alpha_1 - \frac{3}{2}\right)^2$	Squared error, sum of squares, ℓ_2 regression.

The first subsection presents results for the absolute error, which is the most popular loss for ordinal regression. In the second subsection we provide consistency results for a surrogate of the squared loss. Finally, in the third subsection we show results for the 0-1 loss as, perhaps surprisingly, several commonly used surrogates turn out to be consistent with respect to this loss.

3.1 Absolute error surrogates

In this section we will assume that the loss function is the absolute error, i.e., $\ell(y, \alpha) = \left| y - \text{pred}(\alpha) \right|$ and we will focus on surrogates of this loss. For an arbitrary $\alpha \in \mathcal{S}$, the conditional risk for the absolute error can be reformulated using the development of the absolute error from Eq. (8):

$$\begin{aligned}
L(\alpha, p) &= \sum_{i=1}^{k} p_i \left(\sum_{j=1}^{i-1} [\![\alpha_j \geq 0]\!] + \sum_{j=i}^{k-1} [\![\alpha_j < 0]\!] \right) \\
&= \sum_{i=1}^{k} [\![\alpha_j \geq 0]\!] (1 - u_i(p)) + \sum_{j=1}^{k} [\![\alpha_j < 0]\!] u_i(p) \quad ,
\end{aligned}$$

where $u(p)$ is the vector of cumulative probabilities, i.e., $u_i(p) := \sum_{j=1}^{i} p_j$. Let $r = \text{pred}(\alpha)$. Then $\alpha_{r-1} < 0$ and $\alpha_r \geq 0$, from where the above can be simplified to

$$L(\alpha, p) = \sum_{i=1}^{r-1} u_i(p) + \sum_{i=r}^{k-1} (1 - u_i(p)) \quad . \tag{15}$$

Using this expression, we will now derive an explicit minimizer of the conditional risk. Note that because of the prediction function counts the number of nonzero coefficients, only the sign of this vector is of true interest.

Lemma 3 *For any $p \in \Delta^k$, let $\underline{\alpha}(p)$ be defined as*

$$\underline{\alpha}(p) = (2u_1(p) - 1, \ldots, 2u_{k-1}(p) - 1) \quad .$$

Then, $L(\cdot, p)$ achieves its minimum at $\underline{\alpha}(p)$, that is,

$$\underline{\alpha}(p) \in \arg\min L(\alpha, p) \quad .$$

Proof We will prove that for any $\alpha \in \mathcal{S}$ and any $p \in \Delta^k$, $L(\alpha, p) \geq L(\underline{\alpha}(p), p)$. We consider p and α fixed and we denote $r^* = \text{pred}(\underline{\alpha}(p))$ and $r = \text{pred}(\alpha)$. We distinguish three cases, $r < r^*$, $r > r^*$ and $r = r^*$.

- $r < r^*$. In this case, Eq. (15) implies that

$$L(\alpha, p) - L(\underline{\alpha}(p), p) = -\sum_{i=r}^{r^*-1} u_i(p) + \sum_{i=r}^{r^*-1} (1 - u_i(p)) = -\sum_{i=r}^{r^*-1} 2u_i(p) - 1 \quad .$$

  Now, by the definition of prediction function, $2u_i(p) - 1 < 0$ for $i < r^*$, so we have

$$L(\alpha, p) - L(\underline{\alpha}(p), p) = \sum_{i=r}^{r^*-1} \left|2u_i(p) - 1\right| \quad .$$

- $r > r^*$. Similarly, in this case Eq. (15) implies that

$$L(\alpha, p) - L(\underline{\alpha}(p), p) = \sum_{i=r^*}^{r-1} u_i(p) - \sum_{i=r^*}^{r-1} (1 - u_i(p)) = \sum_{i=r^*}^{r-1} 2u_i(p) - 1 \quad .$$

  Since by definition of prediction function $2u_i(p) - 1 \geq 0$ for $i \geq r^*$, it is verified that

$$L(\alpha, p) - L(\underline{\alpha}(p), p) = \sum_{i=r^*}^{r-1} \left|2u_i(p) - 1\right| \quad .$$

- $r = r^*$. In this case, Eq. (15) yields

$$L(\alpha, p) - L(\underline{\alpha}(p), p) = 0 \quad .$$

Let I denote the set of indices for which α disagrees in sign with $\underline{\alpha}$, that is, $I = \{i : \alpha_i(2u_i(p) - 1) < 0\}$. Then, combining the three cases we have the following formula for the excess in conditional risk

$$L(\alpha, p) - L(\underline{\alpha}(p), p) = \sum_{i \in I} \left|2u_i(p) - 1\right| \quad , \tag{16}$$

which is always non-negative and hence $L^*(p) = L(\underline{\alpha}(p), p)$. ■

All threshold (AT). We will now consider the AT surrogate. We will prove that some properties known for binary classification are inherited by this loss function. More precisely, we will provide a characterization of consistency for convex φ in Theorem 5 and excess risk bounds in Theorem 6 that parallel those of Bartlett et al. (2003) for binary classification.

Through this section A will represent the conditional risk of the AT surrogate, which can be expressed as:

$$\begin{aligned} A(\alpha, p) &= \sum_{j=1}^{k} p_j \psi_{\text{AT}}(j, \alpha) = \sum_{j=1}^{k} p_j \left(\sum_{i=1}^{j-1} \varphi(-\alpha_i) + \sum_{i=j}^{k-1} \varphi(\alpha_i) \right) \\ &= \sum_{i=1}^{k-1} (1 - u_i(p))\varphi(-\alpha_i) + u_i(p)\varphi(\alpha_i) \quad , \end{aligned} \tag{17}$$

where as in the previous section $u_i(p) = \sum_{j=1}^{i} p_i$, $\alpha \in \mathcal{S}$ and $p \in \Delta^k$. This surrogate verifies a decomposable property that will be key to further analysis. The property that we are referring to is that the above conditional risk it can be expressed as the sum of $k-1$ binary classification conditional risks. For $\beta \in \mathbb{R}, q \in [0, 1]$, we define C as follows

$$C(\beta, q) = q\varphi(\beta) + (1 - q)\varphi(-\beta) \quad ,$$

where C can be seen as the conditional risk associated with the binary classification loss function φ. Using this notation, the conditional risk A can be expressed in terms of C as:

$$A(\alpha, p) = \sum_{i=1}^{k-1} C(\alpha_i, u_i(p)) \quad .$$

Our aim is to compute A^* in terms of the infimum of C, denoted $C^*(q) := \inf_\beta C(q, \beta)$. Since C is the conditional risk of a binary classification problem, this would yield a link between the optimal risk for the AT surrogate and the optimal risk for a binary classification surrogate. However, this is in general not possible because of the monotonicity constraints in $\mathcal{S}$: the infimum over $\mathcal{S}$ need not equal the infimum over the superset $\mathbb{R}^{k-1}$. We will now present a result that states sufficient conditions under which the infimum over $\mathcal{S}$ and over $\mathbb{R}^{k-1}$ do coincide. This implies that A^* can be estimated as the sum of $k-1$ different surrogate conditional risks, each one corresponding to a binary classification surrogate. A similar result was proven in the empirical approximation setting by Chu and Keerthi (2005, Lemma 1). In this work, the authors consider the hinge loss and show that any minimizer of this loss automatically verifies the monotonicity constraints in $\mathcal{S}$.

In the following lemma we give sufficient conditions on φ under which the monotonicity constraints can be ignored when computing A^*. This is an important step towards obtaining an explicit expression for A^*:

Lemma 4 *Let $\varphi : \mathbb{R} \to \mathbb{R}$ be a function such that $\varphi(\beta) - \varphi(-\beta)$ is a non-increasing function of $\beta \in \mathbb{R}$. Then for all $p \in \Delta^k$, it is verified that*

$$A^*(p) = \sum_{i=1}^{k-1} C^*(u_i(p)) \quad .$$

Proof Let $p \in \Delta^k$ be fixed and let $\alpha^* \in \arg\min_{\alpha \in \mathbb{R}^{k-1}} A(\alpha, p)$. If $\alpha^* \in \mathcal{S}$, then the result is immediate since

$$\sum_{i=1}^{k-1} C^*(u_i(p)) = A(\alpha^*, p) = \inf_{\alpha \in \mathcal{S}} A(\alpha, p) = A^*(p) \ .$$

Suppose now $\alpha^* \notin \mathcal{S}$. We will prove that in this case it is possible to find another vector $\tilde{\alpha} \in \mathcal{S}$ sith the same surrogate risk. By assumption there exists a i in the range $1 \leq i \leq k-2$ for which the monotonicity conditions in $\mathcal{S}$ are not verified. In this case it is verified that $\alpha_{i+1} < \alpha_i$. Since $(u_1(p), \ldots, u_{k-1}(p))$ is a non-decreasing sequence, for a fixed p it is possible to write $u_{i+1}(p) = u_i(p) + \varepsilon$, with $\varepsilon \geq 0$. Then it is true that

$$\begin{aligned} C(\alpha_i^*, u_{i+1}(p)) &= (1 - u_i(p) - \varepsilon)\varphi(-\alpha_i^*) + (u_i(p) + \varepsilon)\varphi(\alpha_i^*) \\ &= C(\alpha_i^*, u_i(p)) + \varepsilon(\varphi(\alpha_i^*) - \varphi(-\alpha_i^*)) \quad . \end{aligned}$$

By assumption $\varepsilon(\varphi(\alpha_i^*) - \varphi(-\alpha_i^*))$ is a non-increasing function of α_i^* and so $\alpha_{i+1}^* < \alpha_i^* \implies C(\alpha_i, u_{i+1}(p)) \leq C(\alpha_{i+1}, u_{i+1}(p))$. By the optimality of α_{i+1}^*, it must be $C(\alpha_i^*, u_{i+1}(p)) = C(\alpha_{i+1}^*, u_{i+1}(p))$. This implies that the vector in which α_{i+1}^* is replaced by α_i^* has the same conditional risk and hence suggest a procedure to construct a vector that satisfies the constraints in $\mathcal{S}$ and achieves the minimal risk in $\mathbb{R}^{k-1}$. More formally, we define $\tilde{\alpha} \in \mathcal{S}$ as:

$$\tilde{\alpha}_i = \begin{cases} \alpha_1^* \text{ if } i = 1 \\ \alpha_i^* \text{ if } \alpha_{i-1}^* \leq \alpha_i^* \\ \alpha_{i-1}^* \text{ if } \alpha_{i-1}^* > \alpha_i^* \end{cases} \quad .$$

Then by the above $C(\alpha_i^*, u_i(p)) = C(\tilde{\alpha}_i, u_i(p))$ for all i and so $A(\alpha^*, p) = A(\tilde{\alpha}, p)$. Now, since $\tilde{\alpha}$ is a non-decreasing vector by construction, $\tilde{\alpha} \in \mathcal{S}$ and we have the sequence of equalities

$$A(\alpha^*, p) = A(\tilde{\alpha}, p) = \sum_{i=1}^{k-1} C^*(u_i(p)) \quad ,$$

which completes the proof. ■

It is easy to verify that the condition on φ of this theorem is satisfied by all the binary losses that we consider: hinge loss, the squared hinge loss, the logistic loss, exponential loss

and the squared loss. With this result, if α_i^* is a minimizer of $C(u_i(p))$, then $(\alpha_1^*, \ldots, \alpha_{k-1}^*)$ will be a minimizer of $A(p)$. Hence, the optimal decision function for the aforementioned values of φ is simply the concatenation of known results for binary classification, which have been derived in Bartlett et al. (2003) for the hinge, squared hinge and Exponential loss and in (Zhang, 2004a) for the logistic loss. Using the results from binary classification we list the values of α^* and A^* in the case of AT for different values of φ:

- *Hinge AT*, : $\alpha_i^*(p) = \text{sign}(2u_i(p) - 1)$, $\quad A^*(p) = \sum_{i=1}^{k-1}\{1 - \big|2u_i(p) - 1\big|\}$.
- *Squared hinge AT*, : $\alpha_i^*(p) = (2u_i(p) - 1)$, $\quad A^*(p) = \sum_{i=1}^{k-1} 4u_i(p)(1 - u_i(p))$.
- *Logistic AT*: $\alpha_i^*(p) = \log\left(\frac{u_i(p)}{1-u_i(p)}\right)$, $A^*(p) = \sum_{i=1}^{k-1}\{-E(u_i(p)) - E(1 - u_i(p))\}$, where $E(t) = t\log(t)$.
- *Exponential AT*: $\alpha_i^*(p) = \frac{1}{2}\log\left(\frac{u_i(p)}{1-u_i(p)}\right)$, $\quad A^*(p) = \sum_{i=1}^{k-1} 2\sqrt{u_i(p)(1 - u_i(p))}$.
- *Squared AT*, $\alpha_i^*(p) = 2u_i(p) - 1$, $\quad A^*(p) = \sum_{i=1}^{k-1}(2 - 2u_i(p))^2$.

It is immediate to check that the models mentioned above are consistent since the decision functions coincides in sign with the minimizer of the risk defined in Lemma 3. Note that the sign of $\alpha_i^*(p)$ at $u_i(p) = \frac{1}{2}$ is irrelevant, since by Eq. (15) both signs have equal risk. We now provide a result that characterizes consistency for a convex φ:

Theorem 5 *Let $\varphi : \mathbb{R} \to \mathbb{R}$ be convex. Then the AT surrogate is consistent if and only if φ is differentiable at 0 and $\varphi'(0) < 0$.*

Proof We postpone the proof until Section 3.4, where this will follow as a particular case of Theorem 13. ■

We will now derive excess risk bounds for AT. These are inequalities that relate the excess conditional risk $L(\alpha) - L^*$, to the excess in surrogate conditional risk $A(\alpha) - A^*$. For this, we will make use of the γ-transform[3] of a binary loss function (Bartlett et al., 2003). For a convex function φ this is defined as

$$\gamma(\theta) = \varphi(0) - C^*\left(\frac{1+\theta}{2}\right) \quad . \tag{18}$$

We will now state the excess risk bound of the AT surrogate in terms of the γ-transform:

Theorem 6 (Excess risk bounds) *Let $\varphi : \mathbb{R} \to \mathbb{R}$ be a function that verifies the following conditions:*

- *φ is convex.*
- *φ is differentiable at 0 and $\varphi'(0) < 0$.*

3. Bartlett et al. (2003) define this as the ψ-transform. However, since we already use ψ to denote the surrogate loss functions we will use letter γ in this case.

- $\varphi(\beta) - \varphi(-\beta)$ *is a non-increasing function of* β.

Then for any $\alpha \in \mathcal{S}, p \in \Delta^k$*, the following excess risk bound is verified:*

$$\gamma\left(\frac{L(\alpha,p) - L^*(p)}{k-1}\right) \leq \frac{A(\alpha,p) - A^*(p)}{k-1} \quad . \tag{19}$$

Proof Let I denote the set of indices in which the sign of α does not coincide with $\underline{\alpha}$, that is, $I = \{i : \alpha_i(2u_i(p) - 1) < 0\}$. From Bartlett et al. (2003, Lemma 7), we know that if φ is convex and consistent (in the context of binary classification), then γ is convex and we can write

$$\begin{aligned}
\gamma\left(\frac{L(\alpha,p) - L^*(p)}{k-1}\right) &= \gamma\left(\frac{\sum_{i\in I} |2u_i(p) - 1|}{k-1}\right) && \text{(by Eq. (16))} \\
&\leq \frac{\sum_{i\in I} \gamma(|2u_i(p) - 1|)}{k-1} && \text{(by Jensen's inequality)} \\
&= \sum_{i\in I} \frac{\gamma(2u_i(p) - 1)}{k-1} && \text{(by symmetry of } \gamma) \\
&= \sum_{i\in I} \frac{\varphi(0) - C^*\left(u_i(p)\right)}{k-1} && \text{(by definition of } \gamma).
\end{aligned} \tag{20}$$

Let $q \in [0,1], \beta \in \mathbb{R}$. If we can further show that $\beta(2q-1) \leq 0$ implies $\varphi(0) \leq C(\beta, q)$, then

$$\begin{aligned}
\sum_{i\in I} \left(\varphi(0) - C^*(u_i(p))\right) &\leq \sum_{i\in I} \left(C(\alpha_i, u_i(p)) - C^*(u_i(p))\right) \\
&\leq \sum_{i=1}^{k-1} \left(C(\alpha_i, u_i(p)) - C^*(u_i(p))\right) \\
&= A(\alpha,p) - A^*(p) \quad \text{(by Lemma 4)}.
\end{aligned}$$

Combining this inequality with Eq. (20), we obtain the theorem. Therefore we only need to prove that $\beta(2q-1) \leq 0$ implies $\varphi(0) \leq C(\beta, q)$. Suppose $\beta(2q-1) \leq 0$. Then by Jensen's inequality

$$C(\beta, q) = q\varphi(\beta) + (1-q)\varphi(-\beta) \geq \varphi(q\beta - (1-q)\beta) = \varphi(\beta(2q-1)) \quad .$$

Now, by convexity of φ we have

$$\varphi(\beta(2q-1)) \geq \varphi(0) + \beta(2q-1)\varphi'(0) \geq \varphi(0) \quad ,$$

where the last inequality follows from the fact that $\varphi'(0) < 0$ and $\beta(2q-1) \leq 0$. This concludes the proof. ■

Note that we have given the excess risk bounds in terms of the conditional risk. These can also be expressed in terms of the full risk, as done for example by Bartlett et al. (2003);

Zhang (2004a). Within the conditions of the theorem, γ is convex and because of Jensen inequality, it is verified that

$$\gamma\left(\mathbb{E}_X\left[L(f(X), \eta(X)) - L^*(\eta(X))\right]\right) \leq \mathbb{E}_X\left[\gamma(L(f(X), \eta(X))\right] \quad .$$

This, together with Eq. (19) yields the following bound in terms of the full risk

$$\begin{aligned}
\gamma\left(\frac{\mathcal{L}(f) - \mathcal{L}}{k-1}\right) &\leq \mathbb{E}_X\left[\gamma\left(\frac{L(f(X), \eta(X)) - L^*(\eta(X))}{k-1}\right)\right] \\
&\leq \mathbb{E}_X\left[\frac{A(f(X), \eta(X)) - A^*(\eta(X))}{k-1}\right] \\
&= \frac{\mathcal{A}(f) - \mathcal{A}^*}{k-1}
\end{aligned}$$

Examples of excess risk bounds. We will now derive excess bounds for different instances of the AT loss function. The values of γ only depend on φ, so we refer the reader to Bartlett et al. (2003) on the estimation of γ for the hinge, squared hinge and Exponential loss and to (Zhang, 2004a) for the logistic loss. Here, we will merely apply the known form of γ to the aforementioned surrogates.

- *Hinge AT,* : $\gamma(\theta) = |\theta| \implies L(\alpha,p) - L^*(p) \leq A(\alpha,p) - A^*$.
- *Squared hinge AT,* : $\gamma(\theta) = \theta^2 \implies \left(\frac{L(\alpha,p) - L^*(p)}{k-1}\right)^2 \leq A(\alpha,p) - A^*$.
- *Logistic AT*: $\gamma(\theta) = \frac{\theta^2}{2} \implies \left(\frac{L(\alpha,p) - L^*(p)}{\sqrt{2}(k-1)}\right)^2 \leq A(\alpha,p) - A^*$.
- *Exponential AT*: $\gamma(\theta) = 1 - \sqrt{1-\theta^2} \implies$ $(k-1)(1 - \sqrt{1 - \frac{(L(\alpha,p) - L^*(p))^2}{k-1}}) \leq A(\alpha,p) - A^*$.
- *Squared AT*: $\gamma(\theta) = \theta^2 \implies \left(\frac{L(\alpha,p) - L^*(p)}{k-1}\right)^2 \leq A(\alpha,p) - A^*$.

For $k = 2$, these results generalize the known excess risk bounds for binary surrogates. For $k > 2$, the normalizing factor $\frac{1}{k-1}$ is not surprising, since the absolute error is bounded by $k-1$ while the 0-1 loss is bounded by 1. While similar excess risk bounds are known for multiclass classification (Zhang, 2004b; Ávila Pires et al., 2013), to the best of our knowledge this is the first time that such bounds have been developed for the AT surrogate ($k > 2$).

Cumulative link (CL). We now focus on the CL loss function defined in Eq. (21), which we restate here for convenience:

$$\psi_{\text{CL}}(y, \alpha) := \begin{cases} -\log(\sigma(\alpha_1)) & \text{if } y = 1 \\ -\log(\sigma(\alpha_y) - \sigma(\alpha_{y-1})) & \text{if } 1 < y < k \\ -\log(1 - \sigma(\alpha_{k-1})) & \text{if } y = k \quad . \end{cases} \tag{21}$$

The terms of this surrogate can be seen as the negative log-likelihood of a probabilistic model in which $\sigma(\alpha)$ model the cumulative probabilities:

$$\sigma(\alpha_i) = P(Y \leq i | X = x) = u_i(p) \quad , \tag{22}$$

and so the likelihood is maximized for $\sigma(\alpha_i^*) = u_i(p)$. Assuming that the inverse of the link function σ exists, this implies that the minimizer of the surrogate loss function (given by the negative log-likelihood) is given by $\alpha_i^*(p) = \sigma^{-1}(u_i(p))$. Plugging into the formula for the surrogate risk yields $A^*(p) = \sum_{i=1}^k p_i \log(p_i)$. This immediately leads to a characterization of consistency based on the link function σ:

Theorem 7 *Suppose σ is an invertible function. Then the CL surrogate is consistent if and only if the inverse link function verifies*

$$\left(\sigma^{-1}(t)\right)(2t-1) > 0 \text{ for } t \neq \frac{1}{2} \quad . \tag{23}$$

Proof ($\Longrightarrow$) Suppose CL is consistent but σ^{-1} does not verify Eq. (23), i.e., there exists a $\xi \neq 1/2$ such that $\sigma^{-1}(\xi)(2\xi - 1) \leq 0$. We consider a probability distribution P such that $u_1(p) = \xi$ for all $p \in \Delta^k$. In that case, by Eq. (22) $\alpha_1^*(p) = \sigma^{-1}(\xi)$ and so this has a sign opposite to the Bayes decision function $2\xi - 1$. By Eq. (16) this implies that $L(\alpha^*) - L^* \geq 2\xi - 1 > 0$. The last inequality implies that α^* does not reach the minimal risk, contradiction since CL is consistent by assumption.

($\Longleftarrow$) Let $0 < i < k$. For $u_i(p) \neq 1/2$, $\alpha_i^*(p) = \sigma^{-1}(u_i(p))$ by Eq. (22) agrees in sign with $2u_i(p) - 1$ and so by Lemma 3 has minimal risk. If $u_i(p) = 1/2$, then in light of Eq. (15) the risk is the same no matter the value of $\alpha_i^*(p)$. We have proven that $\alpha^*(p)$ has the same risk as a Bayes decision function, hence the CL model is consistent. This completes the proof. ■

The previous theorem captures the notion that the inverse of the link function should agree in sign with $2t-1$. When the link function is the sigmoid function, i.e., $\sigma(t) = 1/(1+e^{-t})$ this surrogate is convex and its inverse link function is given by the logit function, which verifies the assumptions of the theorem and hence is consistent. Its optimal decision function is given by

$$\alpha_i^*(p) = \log\left(\frac{u_i(p)}{1 - u_i(p)}\right) \quad ,$$

which coincides with the logistic AT surrogate. Despite the similarities between both surrogates, we have not been able to derive excess risk bounds for this surrogate since the separability properties of the AT are not met in this case. Furthermore, it is possible to construct a counter example that the γ-transform for the Logistic AT ($\gamma(\theta) = \theta^2/2$) loss does not yield a valid risk bound in this case. To see this, let $k = 3$ and p, α be as follows:

$$p = (1 - 2\varepsilon, 1.5\varepsilon, 0.5\varepsilon), \qquad \alpha = (-0.1, -0.05) \quad .$$

For these values we can compute the excess risk and excess surrogate risk as

$$L(\alpha, p) - L^* = \sum_{i=1}^{2}(2u_i(p) - 1) = 2 - 6\varepsilon$$

$$\mathcal{A}(\alpha, p) - \mathcal{A}^* = \sum_{i=1}^{3} p_i \psi_{\text{CL}}(i, \alpha) - \sum_{i=1}^{3} p_i \log(p_i)$$

If the risk bound is satisfied, then the residuals, defined as

$$\text{residuals} = \gamma\left(\frac{L(\alpha, p) - L^*}{k-1}\right) - \frac{\mathcal{A}(\alpha, p) - \mathcal{A}^*}{k-1}$$

must be always negative. However, as it can be seen in Figure 1, the residuals are increasing as ε goes to 0. Also, in the region $\varepsilon < 0.02$ the residuals become positive and hence the inequality does not hold.

We finish our treatment of the CL surrogate by stating the convexity of the logistic CL loss, which despite being a fundamental property of the loss, has not been proven before to the best of our knowledge.

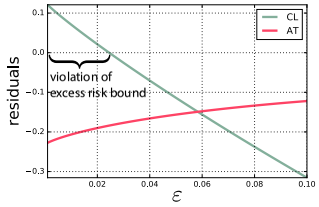


Figure 1: counterexample for the CL excess risk bound.

Lemma 8 *The logistic CL surrogate, $\alpha \mapsto \psi_{CL}(y, \alpha)$, is convex on $\mathcal{S}$ for every value of y.*

Proof $\psi_{\text{CL}}(1, \alpha)$ and $\psi_{\text{CL}}(k, \alpha)$ are convex because they are log-sum-exp functions. It is thus sufficient to prove that $\psi_{\text{CL}}(i, \cdot)$ is convex for $1 < i < k$. For convenience we will write this function as $f(a, b) = -\log\left(\frac{1}{1+\exp(a)} - \frac{1}{1+\exp(b)}\right)$, where $a > b$ is the domain of definition.

By factorizing the fraction inside f to a common denominator, f can equivalently be written as $-\log(\exp(a) - \exp(b)) + \log(1 + \exp(a)) + \log(1 + \exp(b))$. The last two terms are convex because they can be written as a log-sum-exp. The convexity of the first term, or equivalently the log-concavity of the function $f(a, b) = \exp(a) - \exp(b)$ can be settled by proving the positive-definiteness of the matrix $Q = \nabla f(a, b) \nabla f(a, b)^T - f(a, b) \nabla^2 f(a, b)$ for all (a, b) in the domain $\{b > a\}$ (Boyd and Vandenberghe, 2004). In our case,

$$Q = \begin{pmatrix} \exp(a+b) & -\exp(a+b) \\ -\exp(a+b) & \exp(a+b) \end{pmatrix} = \exp(a+b) \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \quad ,$$

which is a positive semidefinite matrix with eigenvalues $2\exp(a+b)$ and 0. This proves that Q is positive semidefinite and thus $\psi_{\text{CL}}(i, \cdot)$ is a convex function. ■

Least absolute deviation. We will now prove consistency of the least absolute deviation (LAD) surrogate. Consistency of this surrogate was already proven for the case $k = 3$

by Ramaswamy and Agarwal (2012). For completeness, we provide here an alternative proof for an arbitrary number of classes.

Theorem 9 *The least absolute deviation surrogate is consistent.*

Proof Recall that for $y \in \mathcal{Y}, \alpha \in \mathcal{S}$, the LAD surrogate is given by

$$\psi_{\text{LAD}}(y, \alpha) = \left| y + \alpha_1 - \frac{3}{2} \right| \quad .$$

The pointwise surrogate risk is then given by

$$A(\alpha, p) = \sum_{i=1}^{k} p_i \psi_{\text{LAD}}(y, \alpha) = \mathbb{E}_{Y \sim p} \left[\left| Y + \alpha_1 - \frac{3}{2} \right| \right] \quad ,$$

where $Y \sim p$ means that Y is distributed according to a multinomial distribution with parameter $p \in \Delta^k$. By the optimality conditions of the median, a value that minimizes this conditional risk is given by

$$\alpha_1^*(p) \in \text{Median}_{Y \sim p} \left(\frac{3}{2} - Y \right) \quad ,$$

where Med is the median, that is, $\alpha_1^*(p)$ is any value that verifies

$$P\left(\frac{3}{2} - Y \leq \alpha_1^*(p) \right) \geq \frac{1}{2} \text{ and } P\left(\frac{3}{2} - Y \geq \alpha_1^*(p) \right) \geq \frac{1}{2} \quad .$$

We will now prove that LAD is consistent by showing that $L(\alpha^*(p), p) = L(\underline{\alpha}(p), p)$, where $\underline{\alpha}$ is the Bayes decision function described in Lemma 3. Let $r^* = \text{pred}(\underline{\alpha}(p))$ and I denote the set $I = \{i : \alpha_i^*(p)(2u_i(p) - 1) < 0\}$. Suppose this set is non-empty and let $i \in I$. We distinguish the cases $\alpha_i^*(p) > 0$ and $\alpha_i^*(p) < 0$:

- $\alpha_i^*(p) < 0$. By Eq. (13), α_i^* and α_1^* are related by $\alpha_i^* = i - 1 + \alpha_1^*$. Then it is verified that

$$\begin{aligned} P\left(\frac{3}{2} - Y \geq \alpha_1^*(p) \right) &= P\left(\frac{3}{2} - Y \geq \alpha_i^* - i + 1 \right) = P\left(\frac{1}{2} + i - \alpha_i^* \geq Y \right) \\ &\geq P\left(\frac{1}{2} + i \geq Y \right) = u_i(p) \quad . \end{aligned}$$

  By assumption, $\alpha_i^*(p)(2u_i(p) - 1) < 0$, which implies $u_i(p) > 1/2$. Hence, by the above we have that $P\left(\frac{3}{2} - Y \geq \alpha_1^*(p) \right) > 1/2$. At the same time, by the definition of median, $P\left(\frac{3}{2} - Y \geq \alpha_1^*(p) \right) \leq 1/2$, contradiction.

- $\alpha_i^*(p) > 0$. Using the same reasoning as before, it is verified that

$$\begin{aligned} P\left(\frac{3}{2} - Y \leq \alpha_1^*(p) \right) &= P\left(\frac{3}{2} - Y \leq \alpha_i^* - i + 1 \right) = P\left(\frac{1}{2} + i - \alpha_i^* \leq Y \right) \\ &\geq P\left(\frac{1}{2} + i \leq Y \right) = 1 - u_i(p) \quad . \end{aligned}$$

  By assumption $u_i(p) < 1/2 \implies P\left(\frac{3}{2} - Y \leq \alpha_1^*(p) \right) > 1/2$. At the same time, by the definition of median, $P\left(\frac{3}{2} - Y \geq \alpha_1^*(p) \right) \leq 1/2$, contradiction.

Supposing I not empty has lead to contradictions in both cases, hence $I = \emptyset$. By Eq. (16), $L(\alpha^*(p), p) = L(\underline{\alpha}(p), p)$, which concludes the proof. ■

3.2 Squared error

We now consider the squared error, defined as

$$\ell(y, \alpha) = (y - \text{pred}(\alpha))^2 \quad ,$$

and its surrogate, the least squares loss,

$$\psi_{\text{LS}}(y, \beta) = \left(y + \alpha_1 - \frac{3}{2} \right)^2 \quad .$$

As for the case of the least absolute deviation, the only difference between the loss and its surrogate is the presence of the prediction function in the first.

We will now prove that the least squares surrogate is consistent with the squared error: we will first derive a value of α, denoted $\underline{\alpha}$ that reaches the Bayes optimal error and then show that the solution to the least squares surrogate agrees in sign with $\underline{\alpha}$ and so also reaches the Bayes optimal error.

Lemma 10 *Let $\underline{\alpha} \in \mathbb{R}^{k-1}$, be defined component-wise as*

$$\underline{\alpha}(p)_i = i - \left(\sum_{j=1}^{k} j p_j \right) + \frac{1}{2} \quad .$$

Then, $\underline{\alpha}$ is a Bayes predictor, that is, $\underline{\alpha} \in \arg\min_\alpha L(\alpha, p)$.

Proof Following our proof for the absolute error, we will show that $L(\alpha) - L(\underline{\alpha})$ is always non-negative, which implies that $\underline{\alpha}$ reaches the Bayes optimal error.

Let r and s be defined as $r = \text{pred}(\alpha)$, $s = \text{pred}(\underline{\alpha})$. Then we have the following sequence of equalities:

$$\begin{aligned} L(\alpha) - L(\underline{\alpha}) &= \sum_{i=1}^{k} p_i((i - r)^2 - (i - s))^2) \\ &= \sum_{i=1}^{k} p_i(-2ir + r^2 + 2is - s^2) \qquad \text{(developing the square)} \qquad (24) \\ &= r^2 - s^2 - 2(r - s) \sum_{i=1}^{k} i p_i \qquad (\text{using } \sum_{i=1}^{k} p_i = 1) \end{aligned}$$

We will now distinguish three cases: $r > s$, $r < s$ and $r = s$. For each of these cases we will show that $L(\alpha) - L(\underline{\alpha}) \geq 0$, which implies that $\underline{\alpha}$ has a smaller risk than any other α, and hence is a Bayes predictor.

- $s > r$. This implies that s is greater than 1 since $s > r \geq 1$. We can conclude from the definition of prediction function in Eq. (2) that the $(s-1)$-th coordinate of $\underline{\alpha}$ is strictly negative. By the definition of $\underline{\alpha}$ we have that $s - 1 - \sum_{i=1}^{k} ip_i + (1/2) < 0$ or equivalently $\sum_{i=1}^{k} ip_i > s - (1/2)$. Using this in Eq. (24) we have the following sequence of inequalities:

$$\begin{aligned} L(\alpha) - L(\underline{\alpha}) &= r^2 - s^2 - 2(r-s)\sum_{i=1}^{k} ip_i = r^2 - s^2 + \underbrace{(-2)(r-s)}_{\text{positive}}\sum_{i=1}^{k} ip_i \\ &\geq r^2 - s^2 + (-2)(r-s)(s - \frac{1}{2}) \qquad (\text{since } \sum_{i=1}^{k} ip_i \geq s - (1/2)) \\ &= \underbrace{r^2 + s^2 - 2rs}_{(r-s)^2} + r - s = \underbrace{(r-s)}_{<0}\underbrace{(r-s+1)}_{\leq 0} \\ &\geq 0 \end{aligned}$$

- $s < r$. We follow a similar argument but reversing the inequalities. The assumption in this case implies that s is smaller than k since $s < r \leq k$. We can conclude from the definition of prediction function in Eq. (2) that the s-th coordinate of $\underline{\alpha}$ is positive. By the definition of $\underline{\alpha}$ we have that $s - \sum_{i=1}^{k} ip_i + (1/2) \geq 0$ or equivalently $-\sum_{i=1}^{k} ip_i \geq -s - (1/2)$. Using this in Eq. (24) we have the following sequence of inequalities:

$$\begin{aligned} L(\alpha) - L(\underline{\alpha}) &= r^2 - s^2 - 2(r-s)\sum_{i=1}^{k} ip_i = r^2 - s^2 + \underbrace{2(r-s)}_{\text{positive}}(-\sum_{i=1}^{k} ip_i) \\ &\geq r^2 - s^2 + 2(r-s)(-s - \frac{1}{2}) \qquad (\text{since } -\sum_{i=1}^{k} ip_i \geq -s - (1/2)) \\ &= \underbrace{r^2 + s^2 - 2rs}_{(s-r)^2} + s - r = \underbrace{(s-r)}_{<0}\underbrace{(s-r+1)}_{\leq 0} \\ &\geq 0 \end{aligned}$$

- $s = r$. Since the excess risk only depends on r, s and not on the particular values of $\alpha, \underline{\alpha}$ (see Eq. (24)), the excess risk is the same in this case, i.e., $L(\alpha) - L(\underline{\alpha}) = 0$.

■

Theorem 11 *The least squares surrogate ψ_{LS} is consistent with respect to the squared error.*

Proof The proof follows closely that of the least absolute deviation (Theorem 9) replacing the median by the expected value. In this case, the pointwise error can be written as

$$A(\alpha, p) = \sum_{i=1}^{k} p_i \psi_{\text{LS}}(y, \alpha) = \mathbb{E}_{Y \sim p}\left(Y + \alpha_1 - \frac{3}{2}\right)^2 \quad ,$$

and by the optimality conditions of the expected value, we have that $\alpha_1^* = \mathbb{E}(\frac{3}{2} - Y | X = x) = \sum_{i=1} p_i(\frac{3}{2} - i)$. Hence, the value for a general α_i^* is given by $i + \alpha_1^*$ (Eq. 13), which can be written as

$$\alpha_i^* = i - \sum_{i=1}^{k} ip_i + \frac{1}{2} \quad .$$

This corresponds to the Bayes predictor of Lemma 10 and hence is consistent. ■

We will encounter another consistent loss function in the Experiments section, when we derive a variant of the AT loss function that is consistent with respect to the square error.

3.3 Surrogates of the 0-1 loss

Perhaps surprisingly, some popular models for ordinal regression turn out to be surrogates, not of the absolute error, but of the 0-1 loss. In this section we focus on the 0-1 loss and we provide a characterization of consistency for the immediate threshold loss function.

Immediate thresholds (IT). In this case, the conditional risk can be expressed as

$$A(\alpha, p) = \sum_{i=1}^{k} p_j \psi_{\text{IT}}(j, \alpha) = \sum_{i=1}^{k-1} p_i \varphi(-\alpha_i) + p_{i+1}\varphi(\alpha_i)$$

As pointed out by Chu and Keerthi (2005), and contrary to what happened for AT surrogate, the constraints can not be ignored in general when computing A^*. Results that rely on this property such as the excess error bound of Theorem 6 will not translate directly for the IT loss. However, we will still be able to characterize the functions φ that lead to a consistent surrogate, in a result analogous to Theorem 5 for the AT surrogate.

Theorem 12 *Let φ be convex. Then the IT surrogate is Fisher consistent with respect to the 0-1 loss if and only if φ is differentiable at 0 and $\varphi'(0) < 0$.*

Proof As for the AT surrogate, this can be seen as a particular case of Theorem 13 with ℓ the 0-1 loss. We will postpone the proof until Section 3.4. ■

3.4 Extension to other admissible loss functions

In this section we will show that the AT and IT surrogates can be seen as particular instances of a family of loss functions for which we will be able to provide a characterization of consistency.

The admissibility criterion that we require on the loss function is that this is of the form $\ell(i, \alpha) = g\left(\left|i - \text{pred}(\alpha)\right|\right)$, where g is a non-decreasing function. Intuitively, this condition implies that labels further away from the true label are penalized more than those closer by. This criterion is general enough to contain all losses considered before such as the absolute error, the squared error and (albeit in a degenerate sense) 0-1 loss. A very similar condition is the V-shape property of (Li and Lin, 2007). This property captures the notion that the loss should not decrease as the predicted value moves away from the true value

by imposing that ℓ verifies $\ell(y,\alpha) \leq \ell(y,\alpha')$ for $\left|y - \text{pred}(\alpha)\right| \leq \left|y - \text{pred}(\alpha')\right|$. The only difference between the two conditions is that our admissibility criterion adds a symmetric condition, i.e., ℓ verifies that the loss of predicting a when the true label is b is the same as the loss of predicting b when the true label is a, which is not necessarily true for V-shaped loss functions. We conjecture that the results in this section are valid for general V-shaped loss functions, although for simplicity we have only proven results for symmetric V-shaped loss functions. For the rest of this section, we will consider that ℓ is a loss function that verifies the admissibility criterion.

We define c_i by $c_0 = g(0)$ and $c_i = g(i) - g(i-1)$ for $0 < i \leq k$. With this notation it is easy to verify by induction that g can be written as a sum of c_i with the formula $g(i) = \sum_{j=1}^{i} c_j$ and that $c_i \geq 0$ by the admissibility property. Following the same development as in Eq. (8), any admissible loss function can be written as a sum of c_i as

$$\ell(y,\alpha) = g\left(\sum_{i=1}^{y-1} [\![\alpha_i \geq 0]\!] + \sum_{i=y}^{k-1} [\![\alpha_i < 0]\!]\right) = \sum_{i=1}^{y-1} c_{y-i} [\![\alpha_i \geq 0]\!] + \sum_{i=y}^{k-1} c_{i-y+1} [\![\alpha_i < 0]\!] \quad . \quad (25)$$

In light of this, it seems natural to define a surrogate for this general loss function by replacing the 0-1 loss with a surrogate as the hinge or logistic that we will denote by φ. This defines a new surrogate that we will denote *generalized all threshold (GAT)*:

$$\psi_{\text{GAT}}(y,\alpha) := \sum_{i=1}^{y-1} \varphi(-\alpha_i) c_{y-i} + \sum_{i=y}^{k-1} \varphi(\alpha_i) c_{i-y+1} \quad .$$

In the special case of the absolute error, c_i is identically equal to 1 and we recover AT loss of Eq. (10). Likewise, for the zero-one loss, c_i will be one for $i \in \{y-1, y\}$ and zero otherwise, recovering the IT loss of Eq. (12). We will now present the main result of this section, which has Theorems 5 and 12 as particular cases.

Theorem 13 *Let φ be convex. Then the GAT surrogate is consistent if and only if φ is differentiable at 0 and $\varphi'(0) < 0$.*

Before presenting the proof of this theorem, we will need some auxiliary results. Unlike for the absolute error, in this case we will not be able to derive a closed form of the optimal decision function. However, we will be able to derive a formula for the excess risk in terms of the functions $u, v : \Delta^k \to \mathbb{R}^{k-1}$, defined as

$$u_i(p) = \sum_{j=1}^{i} p_j c_{i-j+1} \quad v_i(p) = \sum_{j=i+1}^{k} p_j c_{j-i} \quad .$$

Note that we have overloaded the function u defined in Section 3.1. This is not a coincidence, as when ℓ is the absolute error both definitions coincide. Using this notation, the surrogate risk can be conveniently written as

$$\begin{aligned} A(\alpha,p) &= \sum_{i=1}^{k} p_i \psi_{\text{GAT}}(i,\alpha) = \sum_{i=1}^{k} p_i \left(\sum_{j=1}^{i-1} \varphi(-\alpha_j) c_{i-j} + \sum_{j=i}^{k-1} \varphi(\alpha_j) c_{j-i+1}\right) \\ &= \sum_{i=1}^{k-1} v_i(p)\varphi(-\alpha_i) + u_i(p)\varphi(\alpha_i) \quad , \end{aligned}$$

and we have the following formulas for the excess risk:

Lemma 14 *Let $p \in \Delta^k$, $\alpha \in \mathcal{S}$, $r = \text{pred}(\alpha)$ and r^* be the label predicted by any Bayes decision function at p. Then, it is verified that*

$$L(\alpha,p) - L^*(p) = \begin{cases} \sum_{i=r}^{r^*} (v_i(p) - u_i(p)) & \text{if } r < r^* \\ \sum_{i=r^*}^{r-1} (u_i(p) - v_i(p)) & \text{if } r > r^* \end{cases} \quad .$$

Proof The risk can be expressed in terms of u_i and v_i (where the dependence of p is implicit) as

$$\begin{aligned} L(\alpha) &= \sum_{i=1}^{k} p_i g(|r-i|) = \sum_{i=1}^{r-1} p_i g(r-i) + \sum_{i=r+1}^{k} p_i g(i-r) \\ &= \sum_{i=1}^{r-1} p_i \sum_{j=1}^{r-i} c_j + \sum_{i=r+1}^{k} p_i \sum_{j=1}^{i-r} c_j \\ &= \sum_{i=1}^{r-1} u_i + \sum_{i=r}^{k-1} v_i \quad , \end{aligned} \quad (26)$$

hence for $r < r^*$,

$$0 \leq L(\alpha) - L^* = \sum_{i=1}^{r-1} u_i + \sum_{i=r}^{k-1} v_i - \left(\sum_{i=1}^{r^*-1} u_i + \sum_{i=r^*}^{k-1} v_i\right) = \sum_{i=r}^{r^*-1} (v_i - u_i) \quad ,$$

and similarly for $r > r^*$

$$0 \leq L(\alpha) - L^* = \sum_{i=1}^{r-1} u_i + \sum_{i=r}^{k-1} v_i - \left(\sum_{i=1}^{r^*-1} u_i + \sum_{i=r^*}^{k-1} v_i\right) = \sum_{i=r^*}^{r-1} (u_i - v_i) \quad ,$$

■

Proof [Proof of Theorem 13] This proof loosely follows the steps by Bartlett et al. (2003, Theorem 6), with the difference that we must ensure that the optimal value of the surrogate risk lies within $\mathcal{S}$ and adapted to consider multiple classes. We denote by α^* the value in $\mathcal{S}$ that minimizes $A(\cdot)$, by r the prediction at α^* and by r^* the prediction of any Bayes decision function, where the dependence on p is implicit.

($\implies$) We first prove that consistency implies φ is differentiable at 0 and $\varphi'(0) < 0$. We do so by proving that the subdifferential at zero is reduced to a single vector. Since φ is convex, we can find subgradients $g_1 \geq g_2$ of φ at zero such that, for all $\beta \in \mathbb{R}$

$$\begin{aligned} \varphi(\beta) &\geq g_1 \beta + \varphi(0) \\ \varphi(\beta) &\geq g_2 \beta + \varphi(0) \quad . \end{aligned}$$

Then we have for all i

$$\begin{aligned} v_i\varphi(-\beta) + u_i\varphi(\beta) &\geq v_i(g_1\beta + \varphi(0)) + u_i(-g_2\beta + \varphi(0)) \\ &= (v_i g_1 - u_i g_2)\beta + (v_i + u_i)\varphi(0) \\ &= \beta\left(\frac{1}{2}(v_i + u_i)(g_1 - g_2) + \frac{1}{2}(v_i - u_i)(g_1 + g_2)\right) + (v_i + u_i)\varphi(0) \quad . \end{aligned} \tag{27}$$

For $0 < \varepsilon < 1/2$, we will consider the following vector of conditional probabilities

$$p = \left(0, \cdots, 0, \frac{1}{2} - \varepsilon, \frac{1}{2} + \varepsilon\right) \quad ,$$

from where u_i and v_i take the following simple form

$$u_i = \begin{cases} p_{k-1}c_1 & \text{if } i = k-1 \\ 0 & \text{otherwise} \end{cases}, \quad v_i = \begin{cases} p_k c_1 & \text{if } i = k-1 \\ p_{k-1}c_{k-i-1} + p_k c_{k-i} & \text{otherwise} \end{cases}$$

hence by Eq. (26) consistency implies $r = k$ and so we must have $\alpha^*_{k-1} < 0$.

Let now $\tilde{\alpha} \in \mathcal{S}$ be a vector that equals α^* in all except the last component, which is zero (i.e., $\tilde{\alpha}_{k-1} = 0$). We will now prove that if $g_1 > g_2$ then $A(\tilde{\alpha}, p) < A(\alpha^*, p)$ leading to a contradiction. For the particular choice of u_{k-1}, v_{k-1} above, equation (27) can be simplified to

$$v_{k-1}\varphi(-\beta) + u_{k-1}\varphi(\beta) \geq \beta\left[\frac{1}{2}(g_1 - g_2) + \varepsilon(g_1 + g_2)\right] + (v_{k-1} + u_{k-1})\varphi(0) \quad .$$

Since by assumption $g_1 > g_2$, it is always possible to choose ε small enough such that the quantity inside the square brackets is strictly positive. Special casing at $\beta = \alpha^*_{k-1}$ and using $\alpha^*_{k-1} < 0$ yields the following inequality:

$$v_{k-1}\varphi(-\alpha^*_{k-1}) + u_{k-1}\varphi(\alpha^*_{k-1}) \geq (v_{k-1} + u_{k-1})\varphi(0) \quad .$$

We then have the following sequence of inequalities:

$$\begin{aligned} A(\alpha^*, p) &= \sum_{i=1}^{k-1} v_i(p)\varphi(-\alpha^*_i) + u_i\varphi(\alpha^*_i) \\ &\geq \sum_{i=1}^{k-2} \left\{v_i(p)\varphi(-\alpha^*_i) + u_i\varphi(\alpha^*_i)\right\} + (v_{k-1} - u_{k-1})\alpha^*_{k-1} + (v_{k-1} + u_{k-1})\varphi(0) \\ &\qquad \text{(by last inequality)} \\ &> \sum_{i=1}^{k-2} \left\{v_i(p)\varphi(-\alpha^*_i) + u_i\varphi(\alpha^*_i)\right\} + v_{k-1}\varphi(0) + u_{k-1}\varphi(0) \\ &= A(\tilde{\alpha}, p) \quad , \end{aligned}$$

which results in $A(\alpha^*, p) > A(\tilde{\alpha}, p)$, contradiction since α^* is the value with lowest surrogate risk. This implies that if the GAT loss is consistent, then φ is differentiable at 0. To see that we must also have $\varphi'(0) < 0$, notice that from Eq. (27) we have

$$A_i(\beta) \geq (v_i - u_i)\varphi'(0)\beta + A_i(0) \quad .$$

But for any $v_i > u_i$ and $\beta < 0$, if $\varphi'(0) \geq 0$, then this expression is greater than $A_i(0)$. Hence, if GAT is consistent then $\varphi'(0) < 0$, which concludes one of the implications of the proof.

($\Longleftarrow$) We now prove that if φ is differentiable at 0 and $\varphi'(0) < 0$, then GAT is consistent.

The first order optimality conditions states that there exists $\lambda_i \geq 0$ such that the optimal value of $A(\alpha, p)$ subject to $\alpha \in \mathcal{S}$ is the minimizer of the following unconstrained function:

$$G(\alpha) = A(\alpha, p) + \sum_{i=1}^{k-1} \lambda_i(\alpha_i - \alpha_{i+1}) \quad .$$

We show that assuming GAT is not consistent (i.e., $L(\alpha^*) > L^*$) leads to a contradiction and hence GAT must be consistent.

We start by computing the partial derivative of G at zero:

$$\frac{\partial G}{\partial \alpha_i}\Big|_{\alpha_i = 0} = (u_i - v_i)\varphi'(0) - \lambda_{i-1} + \lambda_i \quad ,$$

were for convenience $\lambda_0 = 0$. Note that $\alpha^*_{r-1} < 0 \leq \alpha^*_r$ verifies by definition of prediction function. Hence, the inequality constraints are verified with strict inequality and by complementary slackness $\lambda_{r-1} = 0$. Suppose first $r < r^*$. Then, the addition of all partial derivatives between r and r^* yields

$$\sum_{i=r}^{r^*} \frac{\partial G}{\partial \alpha_i}\Big|_{\alpha_i = 0} = \left(\sum_{i=r}^{r^*} u_i - v_i\right)\varphi'(0) + \lambda_{r^*} \quad ,$$

which by Lemma 14 is strictly positive. Consider the convex real-valued function of $\alpha_i \to G(\alpha^*_1, \ldots, \alpha^*_{r-1}, \ldots, \alpha_i, \ldots)$, that is, the function $G(\alpha^*)$ restricted to $\alpha_i, i \geq r$. Since $\alpha^*_i \geq 0$ for all $i \geq r$ by the definition of prediction function, and knowing the subdifferential of a convex function is a monotonous, this implies that $\partial G/\partial \alpha_{i \geq r} \leq 0$ at $\alpha_i = 0$, contradiction.

Now we suppose $r > r^*$. Then, the addition of all partial derivatives between r^* and r yields

$$\sum_{i=r^*}^{r} \frac{\partial G}{\partial \alpha_i}\Big|_{\alpha_i = 0} = \left(\sum_{i=r^*}^{r} u_i - v_i\right)\varphi'(0) - \lambda_{r^*-1} \quad ,$$

which by Lemma 14 is strictly negative. As before, we consider the function the function $G(\alpha^*)$ restricted to $\alpha_i, i \geq r$. By the definition of prediction function, $\alpha^*_{i<r} < 0$ and so the fact that the subdifferential of a real-valued function is monotonous, we have that $\partial G/\partial \alpha_{i \geq r} \geq 0$ at $\alpha_i = 0$, contradiction. ■

4. Threshold-based decision functions and parametric consistency

In this section we revisit the assumption that the optimal decision function can be estimated independently at each point $x \in \mathcal{X}$. This is implicitly assumed on most consistency studies,

however in practice models often enforce inter-observational constraints (e.g. smoothness). In the case of ordinal regression it is often the case that the decision functions are of the form

$$f(x) = (\theta_1 - g(x), \theta_2 - g(x), \ldots, \theta_{k-1} - g(x)) \quad , \tag{28}$$

where $(\theta_1, \ldots, \theta_{k-1})$ is a non-decreasing vector (i.e., its components form a non-decreasing sequence) known as the *vector of thresholds* (hence the appearance of the name thresholds in many models) and g is a measurable function. We will call decision functions of this form *threshold-based* decision functions. All the examined models are of this form with the exception of the least absolute deviation are commonly constrained to this family of decision functions (Chu and Keerthi, 2005; Rennie and Srebro, 2005; Lin and Li, 2006; Shashua and Levin, 2003).

The main issue with such decision functions is that since the vector of thresholds is estimated from the data, it is no longer true that the optimal decision function can be estimated independently at each point. This implies that the pointwise characterization of Fisher consistency described in Lemma 2 does no longer hold when restricted to this family and hence the consistency proofs in previous sections no longer hold.

Let $\mathcal{F}$ be the set of functions of the form of Eq. (28). We will now apply the notion of $\mathcal{F}$-*consistency* or *parametric consistency* of (Shi et al., 2015) to the threshold-based setting. This is merely the notion of Fisher consistency where the decision functions are restricted to a family of interest:

Definition 15 *($\mathcal{F}$-**Consistency**) Given a surrogate loss function $\psi : \mathcal{Y} \times \mathcal{S} \to \mathbb{R}$, we will say that the surrogate loss function ψ is $\mathcal{F}$-consistent with respect to the loss $\ell : \mathcal{Y} \times \mathcal{S} \to \mathcal{Y}$ if for every probability distribution over $X \times Y$ it is verified that every minimizer of the ψ-risk reaches the optimal risk in $\mathcal{F}$, that is,*

$$f^* \in \arg\min_{f \in \mathcal{F}} \mathcal{A}(f) \implies \mathcal{L}(f^*) = \inf_{f \in \mathcal{F}} \mathcal{L}(f) \quad .$$

We will show that by imposing additional constraints on the probability distribution P we will be able to derive $\mathcal{F}$-consistency for particular surrogates. In the following theorem we give one sufficient condition for $\mathcal{F}$-consistency of the logistic all threshold and the logistic CL. This condition involves the odds-ratio, defined as

$$R_i(x) = \frac{u_i(\eta(x))}{1 - u_i(\eta(x))} \quad ,$$

where $\eta(x)$ is the vector of conditional probabilities defined in Section 1.2 and u_i is the vector of conditional probabilities.

Theorem 16 *If the quotient of successive R_i is independent of x for all $0 < i < k-1$, that is, if*

$$\frac{R_i(x)}{R_{i+1}(x)} = a_i \quad \forall x \in \mathcal{X} \quad ,$$

for some real number a_i, then the logistic all threshold and the logistic CL are $\mathcal{F}$-consistent.

Proof It will be sufficient to prove that under the constraints on P, the optimal decision function for the unconstrained problem belongs to $\mathcal{F}$. In Section 3.1 we derived the optimal decision function for the logistic all threshold and the logistic CL. Hence, we can write

$$\alpha_i^*(\eta(x)) - \alpha_{i+1}^*(\eta(x)) = \log\left(\frac{u_i(\eta(x))}{1 - u_i(\eta(x))}\right) - \log\left(\frac{u_{i+1}(\eta(x))}{1 - u_{i+1}(\eta(x))}\right) = \log\left(\frac{R_i(x)}{R_{i+1}(x)}\right) \quad .$$

It is clear that α is a threshold-based decision function of the form Eq. (28) if and only if $\alpha_i - \alpha_{i+1}$ does not depend on $x \in \mathcal{X}$. Given the above, we can guarantee that the optimal α belongs to $\mathcal{F}$ if the last term is independent of x. One can then easily recognize the quotient of odds-ratio of the theorem's conditions and conclude that $\alpha_i^*(\eta(x)) - \alpha_{i+1}^*(\eta(x))$ is independent of x. This concludes the proof. ■

This sufficient condition is admittedly a very stringent one on the probability distribution P. Unfortunately, a deeper understanding of $\mathcal{F}$-consistency, while an interesting future direction, is outside the scope of the current paper.

5. Experiments: A novel surrogate for the squared error

While the focus of this work is a theoretical investigation of consistency, we have also conducted experiments to study a novel surrogate suggested by the results of the last section. There, we constructed a surrogate that is consistent with any loss function that verifies an admissible criterion. In particular, Theorem 13 applied to the squared loss yields the following consistent surrogate:

$$\psi(y, \alpha) = \sum_{i=1}^{y-1} \varphi(-\alpha_i)(2(y-i) - 1) + \sum_{i=y}^{k-1} \varphi(\alpha_i)(2(i-y) + 1) \quad .$$

In principle, any binary loss function can be used for φ, although in the experiments we set it to the hinge loss function. To the best of our knowledge, this is a novel surrogate. We compare the cross-validation error of this surrogate on different datasets against the least squares surrogate (for which we proved consistency in §3.2) where $\beta \in \mathbb{R}$ and prediction is given by rounding to the closest integer. In both cases, we consider linear decision functions, i.e.

$$\alpha = (\theta_1 - \langle w, x\rangle, \ldots, \theta_{k-1} - \langle w, x\rangle) \quad \text{and} \quad \beta = \langle w, x\rangle \quad .$$

In each case, the optimal values of w, θ were found by minimizing the empirical surrogate risk. For the training sample $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, $x_i \in \mathbb{R}^p$, it yielding the following optimization problems for GAT and least squares, respectively:

$$\arg\min_{\theta \in \mathcal{S}, w \in \mathbb{R}^p} \sum_{i=1}^{n} \left\{ \sum_{j=1}^{y_i - 1} \varphi(\langle w, x_i\rangle - \theta_j)(2(y-j) - 1) + \sum_{j=y_i}^{k-1} \varphi(\theta_j - \langle w, x_i\rangle)(2(j-y) + 1) \right\}$$

$$\arg\min_{w \in \mathbb{R}^p} \sum_{i=1}^{n} (y_i - \langle w, x_i\rangle)^2$$

The different datasets that we will consider are described in (Chu and Keerthi, 2005) and can be download from the authors website[4]. We display results for the 9 datasets of SET I using the version of the datasets with 5 bins, although similar results were observed when using the datasets with 10 bins. Given the small dimensionality of the datasets (between 6 and 60) and the comparatively high number of samples (between 185 and 4000), we did not consider the use of regularization.

Performance is computed as the squared error on left out data, averaged over 20 folds. We report this performance in Figure 2, where it can be seen that the GAT surrogate outperforms LS on 7 out of 9 datasets, although admittedly the difference is small and only statistically significant on 3 datasets. However, this shows that previous theoretical results can be used to generate consistent surrogates that are competitive in a practical scenario.

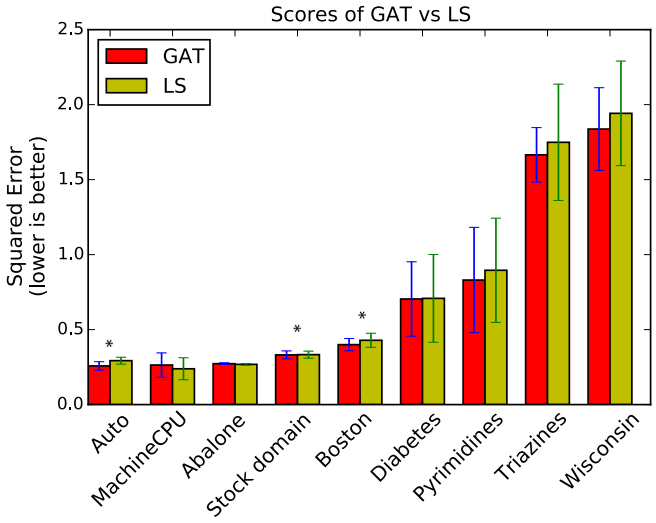


Figure 2: Scores of the generalized all threshold (GAT) and least squares (LS) surrogate on 6 different datasets. The scores are computed as the squared error between the prediction and the true labels on left out data, averaged over 20 cross-validation folds. On 7 out of 9 datasets all the GAT surrogate outperforms the least squares estimator, showing that this surrogate yields a highly competitive model. Datasets for which a Wilcoxon signed-rank test rejected the null hypothesis that the means are equal with p-value < 0.01 are highlighted by a $*$ symbol over the bars.

4. `http://www.gatsby.ucl.ac.uk/~chuwei/ordinalregression.html`.

6. Conclusions

In this paper we have characterized the consistency for a rich family of surrogate loss functions used for ordinal regression. Our aim is to bridge the gap between the consistency properties known for classification and ranking and those known for ordinal regression.

We have first described a wide family of ordinal regression methods under the same framework. The surrogates of the absolute error that we have considered are the all threshold (AT), cumulative link (CL), and least absolute deviation (LAD), while the surrogate for the 0-1 loss is the immediate threshold (IT).

For all the surrogates considered, we have characterized its consistency. For AT and IT, consistency was characterized by the derivative of a real-valued convex function at zero (Theorems 5 and 12 respectively). For CL, consistency was characterized by a simple condition on its link function (Theorem 7) and for LAD we have extended the proof of Ramaswamy and Agarwal (2012) to an arbitrary number of classes (Theorem 5). Furthermore, we have proven that AT verifies a decomposability property and using this property we have provided excess risk bounds that generalize those of Bartlett et al. (2003) for binary classification(Theorem 6).

The derivation we have given when introducing IT and AT are identical except for the underlying loss function. This suggest that both can be seen as special cases of a more general family of surrogates. In Section 3.4 we have constructed such surrogate and characterized its consistency with respect to a general loss function that verifies an admissibility condition. Again, the characterization only relies on the derivative at zero of a convex real-valued function. We named this surrogate generalized all threshold (GAT).

In Section 4 we have turned back to examine one of the assumptions described in the introduction and that is common to the vast majority of consistency studies, i.e., that the optimal decision function can be estimated independently at every sample. However, in the setting of ordinal regression it is common for decision functions to have a particular structure known as threshold-based decision functions and which violates this assumption. Following (Shi et al., 2015), we are able to prove a restricted notion of consistency known as $\mathcal{F}$-consistency or parametric consistency on two surrogates by enforcing constraints on the probability distribution P. We believe this restricted notion of consistency to be important in practice and we look forward to see consistency studies extended to consider different types of decision functions, such as smooth functions, polynomial functions, etc.

Finally, in Section 5 we provide an empirical comparison for the GAT surrogate. The underlying loss function that we consider in this case is the squared error, in which case GAT yields a novel surrogate. We compare this surrogate against the least squares surrogate in terms of cross-validation error. Our results show that GAT outperforms least squares on 7 out of 9 datasets, showing the pertinence of such surrogate on real-world datasets.

A direction for future work would be to study the related notion of asymptotic consistency for ordinal regression loss functions, similar to the results that already exist for binary classification (Devroye et al., 1994; Steinwart, 2002).

7. Acknowledgments

Most of this work was done while FP was a PhD student at the Parietal project-team at INRIA Saclay and acknowledges financial support under grants IRMGroup ANR-10-BLAN-0126-02 and BrainPedia ANR-10-JCJC 1408-01 and the "Chaire Économie des Nouvelles Données", under the auspices of Institut Louis Bachelier, Havas-Media and Université Paris-Dauphine. We would like to thank our colleague Guillaume Obozinski for fruitful discussions.

References

Shivani Agarwal. Generalization bounds for some ordinal regression algorithms. *ALT '08 Proceedings of the 19th international conference on Algorithmic Learning Theory*, 2008.

Alan Agresti. *Analysis of ordinal categorical data*. John Wiley & Sons, 2010.

Cande V. Ananth and David G. Kleinbaum. Regression models for ordinal responses: a review of methods and applications. *International journal of epidemiology*, 1997.

Ben G. Armstrong and Margaret Sloan. Ordinal regression models for epidemiologic data. *American Journal of Epidemiology*, 1989.

Bernardo Ávila Pires, Csaba Szepesvari, and Mohammad Ghavamzadeh. Cost-sensitive multiclass classification risk bounds. In *Proceedings of The 30th International Conference on Machine Learning*, 2013.

Peter L. Bartlett, Michael I. Jordan, and Jon D. Mcauliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 2003.

Shai Ben-David, Nadav Eiron, and Philip M. Long. On the difficulty of approximately maximizing agreements. *Journal of Computer and System Sciences*, 2003.

Stéphane Boucheron, Olivier Bousquet, and Gábor Lugosi. Theory of classification: A survey of some recent advances. *ESAIM: probability and statistics*, 9:323–375, 2005.

Stephen P. Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

Andreas Buja, Werner Stuetzle, and Yi Shen. Loss functions for binary class probability estimation and classification: Structure and applications. Technical report, University of Pennsylvania, November 2005.

Clément Calauzenes, Nicolas Usunier, Patrick Gallinari, et al. On the (non-) existence of convex, calibrated surrogate losses for ranking. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

Wei Chu and Zoubin Ghahramani. Gaussian processes for ordinal regression. *Journal of Machine Learning Research*, 2004.

Wei Chu and S Sathiya Keerthi. New approaches to support vector ordinal regression. In *Proceedings of the 22th International Conference on Machine Learning (ICML)*, 2005.

Carlo Ciliberto, Lorenzo Rosasco, and Alessandro Rudi. A consistent regularization approach for structured prediction. In *Advances in Neural Information Processing Systems*, 2016.

Koby Crammer and Yoram Singer. Pranking with ranking. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

Koby Crammer and Yoram Singer. Online ranking by projecting. *Neural Computation*, 2005.

Luc Devroye, Laszlo Gyorfi, Adam Krzyzak, and Gabor Lugosi. On the strong universal consistency of nearest neighbor regression function estimates. *The Annals of Statistics*, 1994.

Orla M. Doyle, John Ashburner, F.O. Zelaya, Stephen C.R. Williams, Mitul A. Mehta, and Andre F. Marquand. Multivariate decoding of brain images using ordinal regression. *NeuroImage*, 2013.

John C. Duchi, Lester W. Mackey, and Michael I. Jordan. On the consistency of ranking algorithms. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010.

Vitaly Feldman, Venkatesan Guruswami, Prasad Raghavendra, and Yi Wu. Agnostic learning of monomials by halfspaces is hard. *SIAM Journal on Computing*, 41(6):1558–1590, 2012.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

William H. Greene. *Econometric analysis*. Prentice-Hall, Inc, 1997.

Craig T. Hartrick, Juliann P. Kovan, and Sharon Shapiro. The numeric rating scale for clinical pain measurement: A ratio measure? *Pain Practice*, 3(4):310–316, 2003.

Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Support vector learning for ordinal regression. *IET Conference Proceedings*, 1999.

Donald E Knuth. Two notes on notation. *American Mathematical Monthly*, 1992.

Stefan Kramer, Gerhard Widmer, Bernhard Pfahringer, and Michael De Groeve. Prediction of ordinal classes using regression trees. *Fundamenta Informaticae*, 2001.

Yoonkyung Lee, Yi Lin, and Grace Wahba. Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 2004.

Ling Li and Hsuan-tien Lin. Ordinal regression by extended binary classification. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2007.

Hsuan-Tien Lin and Ling Li. Large-margin thresholded ensembles for ordinal regression: Theory and practice. In *Algorithmic Learning Theory*. Springer, 2006.

Yi Lin. A note on margin-based loss functions in classification. *Statistics & Probability Letters*, 2004.

Tie-Yan Liu. *Learning to rank for information retrieval*. Springer Science & Business Media, 2011.

Peter McCullagh. Regression models for ordinal data. *Journal of the Royal Statistical Society*, 1980.

Anton Osokin, Francis Bach, and Simon Lacoste-Julien. On structured prediction theory with calibrated convex surrogate losses. *arXiv preprint arXiv:1703.02403*, 2017.

Harish G. Ramaswamy and Shivani Agarwal. Classification Calibration Dimension for General Multiclass Losses. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2012.

Harish G Ramaswamy and Shivani Agarwal. Convex calibration dimension for multiclass loss matrices. *Journal of Machine Learning Research*, 2016.

Mark D Reid and Robert C Williamson. Composite binary losses. *Journal of Machine Learning Research*, 2010.

Jason D. M. Rennie and Nathan Srebro. Loss functions for preference levels : Regression with discrete ordered labels. In *Proceedings of the IJCAI Multidisciplinary Workshop on Advances in Preference Handling*, 2005.

Amnon Shashua and Anat Levin. Ranking with large margin principle : Two approaches. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2003.

Qinfeng Shi, Mark Reid, Tiberio Caetano, Anton Van den Hengel, and Zhenhua Wang. A hybrid loss for multiclass and structured prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

Ingo Steinwart. Support vector machines are universally consistent. *Journal of Complexity*, September 2002.

Charles J. Stone. Consistent nonparametric regression. *The Annals of Statistics*, 1977.

Ambuj Tewari and Peter L. Bartlett. On the consistency of multiclass classification methods. *Journal of Machine Learning Research*, 2007.

Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *The Annals of Statistics*, 2004a.

Tong Zhang. Statistical analysis of some multi-category large margin classification methods. *Journal of Machine Learning Research*, 2004b.

Journal of Machine Learning Research 18 (2017) 1-26 Submitted 11/15; Revised 10/16; Published 7/17

Statistical Inference with Unnormalized Discrete Models and Localized Homogeneous Divergences

Takashi Takenouchi* ttakashi@fun.ac.jp
Future University Hakodate
RIKEN Center for Advanced Intelligence Project
116-2 Kamedanakano, Hakodate, Hokkaido, 040-8655, Japan

Takafumi Kanamori kanamori@is.nagoya-u.ac.jp
Department of Computing and Software Systems, Nagoya University
RIKEN Center for Advanced Intelligence Project
Furocho, Chikusaku, Nagoya 464-8603, Japan

Editor: Guy Lebanon

Abstract

In this paper, we focus on parameters estimation of probabilistic models in discrete space. A naive calculation of the normalization constant of the probabilistic model on discrete space is often infeasible and statistical inference based on such probabilistic models has difficulty. In this paper, we propose a novel estimator for probabilistic models on discrete space, which is derived from an empirically localized homogeneous divergence. The idea of the empirical localization makes it possible to ignore an unobserved domain on sample space, and the homogeneous divergence is a discrepancy measure between two positive measures and has a weak coincidence axiom. The proposed estimator can be constructed without calculating the normalization constant and is asymptotically consistent and Fisher efficient. We investigate statistical properties of the proposed estimator and reveal a relationship between the empirically localized homogeneous divergence and a mixture of the α-divergence. The α-divergence is a non-homogeneous discrepancy measure that is frequently discussed in the context of information geometry. Using the relationship, we also propose an asymptotically consistent estimator of the normalization constant. Experiments showed that the proposed estimator comparably performs to the maximum likelihood estimator but with drastically lower computational cost.

Keywords: unnormalized model, homogeneous divergence, empirical localization, discrete model

1. Introduction

In the fields of machine learning and pattern recognition, probabilistic models on discrete space are useful for classification tasks or modeling discrete phenomena, so estimating parameters of probabilistic models on discrete space is a widely studied and important challenge. For example, the Boltzmann machine (with hidden variables) (Hinton and Sejnowski, 1986; Ackley et al., 1985; Amari et al., 1992) is a widely used probabilistic model to represent binary variables, and the restricted Boltzmann machine (RBM) is especially attracting increasing attention in the context of deep learning (Hinton, 2010; Hinton and Salakhutdinov, 2012). Training of probabilistic models on discrete space, i.e., estimation of parameters, is usually done by using the maximum likelihood estimation (MLE). An explicit expression of the MLE cannot generally be obtained, so the gradient-based optimization is usually used. A difficulty of the gradient-based optimization for such models comes from the calculation of the normalization constant. The calculation of the gradient requires calculation of the normalization constant in each step of the optimization and its computational cost sometimes increases exponentially. To tackle the problem of computational cost, various kinds of approximation methods have been proposed. One approach tries to approximate the probabilistic model (or expectation with the model) by using a tractable model or sampling techniques. The mean-field approximation is a popular method to approximate the model by assuming independence among variables (Opper and Saad, 2001). The contrastive divergence (Hinton, 2002) avoids the exponential order calculation using the Markov Chain Monte Carlo (MCMC) sampling: the method approximates the expectation using samples obtained by the MCMC that runs few steps from the empirical distribution with a transition matrix defined with a current model. Another approach is estimation based on an unnormalized model, which does not include the normalization constant and so is not necessarily a probability. In the literature of parameters estimation of probabilistic models for continuous variables, Hyvärinen (2005) used a score function which is a gradient of log-density with respect to the data vector rather than parameters. This approach makes it possible to estimate parameters without calculating the normalization term by focusing on the shape of the density function. Hyvärinen (2007) extended the method to discrete variables. This method uses information of a "neighbor" by contrasting its probability with that of a flipped variable. Gutmann and Hirayama (2012) extended the above framework and developed an approximated estimator on the basis of the Bregman divergence. Dawid et al. (2012) proposed a generalized local scoring rules on discrete sample spaces and theoretically discussed a class of scoring rules for appropriate estimation. Takenouchi (2015) focused on a specific class of models on discrete space and constructed an asymptotically consistent estimator using the Itakura-Saito distance.

In this paper, we propose a novel method of parameter estimation for probabilistic models on discrete space. The proposed estimator can be constructed without calculation of the normalization constant by utilizing the unnormalized model and has suitable statistical properties such as consistency and efficiency.[1] The proposed estimator is derived from minimization of a risk function defined by a combination of two ideas: the homogeneous divergence and an empirical localization. The conventional divergence measure that has a coincidence axiom ensures that the divergence is zero if and only if two positive measures are equal. On the other hand, the homogeneous divergence follows a weak coincidence axiom and is zero if and only if two positive measures have a proportional relationship. By virtue of the property, the homogeneous divergence that has the weak coincidence axiom can ignore the normalization constant, and the proposed estimator does not need to calculate the normalization constant. The empirical localization is a method of transformation of the unnormalized model and is an extension of the geometric mean of the model and the

1. A short version of this article was published as a conference paper (Takenouchi and Kanamori, 2015). In this paper, we show the possibility of generalization of the basic framework shown in (Takenouchi and Kanamori, 2015) and add some new theoretical results, including estimator for the normalization constant. Also we add experiments with various kinds of models and settings to assess the validity of the proposed method.

©2017 Takashi Takenouchi, and Takafumi Kanamori.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-596.html.

empirical distribution of a given dataset. By using the empirical localization, we can restrict the domain of the estimation problem to that of observed examples and omit calculations associated with the domain of unobserved examples. This restriction of the domain can drastically reduce computational cost. The derived risk function is convex for various kinds of models including the higher order Boltzmann machine and so is easy to optimize. Also, we investigate statistical properties of the proposed estimator and reveal that the proposed estimator has the same asymptotic efficiency as the MLE. The proposed risk function is closely related with the α-divergence (Amari and Nagaoka, 2000), which induces a Fisher efficient estimator for discrete probabilistic models. The α-divergence is not homogeneous divergence, and by utilizing the relationship, we also developed a feasible estimator of the normalization constant.

Basic settings are described in Section 2, and we introduce the homogeneous divergence and the idea of empirical localization of the unnormalized model in Section 3. We describe a novel estimator for probabilistic models on discrete spaces and investigate its statistical properties of the proposed estimator in Section 4. In Section 5, we discuss a characterization of the risk function of the proposed estimator with the α-divergence and construct a feasible estimator of the normalization constant. We numerically verify performance of the proposed estimator by using synthetic datasets in Section 6.

2. Settings

Let $\mathcal{X}$ be a discrete space such as $\{+1,-1\}^d$ or the set of natural numbers. The bracket $\langle f \rangle$ for a real-valued function f on $\mathcal{X}$ denotes the sum of $f(\boldsymbol{x})$ over $\mathcal{X}$ i.e., $\langle f \rangle = \sum_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x})$. We define $\mathcal{M}$ and $\mathcal{P}$ as the set of all non-negative functions and that of all probability functions on $\mathcal{X}$,

$$\mathcal{M} = \{ f : \mathcal{X} \to \mathbb{R}_{\geq 0} \,|\, \langle f \rangle < \infty \}, \ \mathcal{P} = \{ f \in \mathcal{M} \,|\, \langle f \rangle = 1 \}$$

where $\mathbb{R}_{\geq 0}$ is the set of all non-negative real numbers.

In this paper, we focus on parameter estimation of a probabilistic model $\bar{q}_{\boldsymbol{\theta}}(\boldsymbol{x})$ on $\mathcal{X}$ that is written as

$$\bar{q}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{Z_{\boldsymbol{\theta}}} \tag{1}$$

where $\boldsymbol{\theta}$ is an m-dimensional vector of parameters, $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ is an unnormalized model in $\mathcal{M}$ and $Z_{\boldsymbol{\theta}} = \langle q_{\boldsymbol{\theta}} \rangle$ is the normalization constant. The equality $\langle q_{\boldsymbol{\theta}} \rangle = \sum_{\boldsymbol{x}\in\mathcal{X}} q_{\boldsymbol{\theta}}(\boldsymbol{x}) = 1$ does not necessarily hold for unnormalized models, and the normalization constant $Z_{\boldsymbol{\theta}}$ typically requires a high computational cost. We thus need a device to circumvent the computation of Z_θ in statistical inferences. Throughout the paper, we assume without loss of generality that the unnormalized model $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ can be written as

$$q_{\boldsymbol{\theta}}(\boldsymbol{x}) = \exp(\psi_{\boldsymbol{\theta}}(\boldsymbol{x})), \tag{2}$$

where $\psi_{\boldsymbol{\theta}}(\boldsymbol{x})$ is a function on $\mathcal{X}$ with the parameter $\boldsymbol{\theta}$.

Remark 1 *By setting $\psi_{\boldsymbol{\theta}}(\boldsymbol{x})$ as $\psi_{\boldsymbol{\theta}}(\boldsymbol{x}) - \log Z_{\boldsymbol{\theta}}$, the normalized model* (1) *can be written as* (2).

Example 1 *The Bernoulli distribution on $\mathcal{X} = \{+1,-1\}$ is the simplest example of the probabilistic model* (1) *using the function $\psi_\theta(x) = \theta x$ for $x \in \mathcal{X}$ and $\theta \in \mathbb{R}$.*

Example 2 *For the parameter $\theta \in \mathbb{R}$, the function $\psi_\theta(x) = x\theta - \log x!$ leads to an unnormalized model for the Poisson distribution on $\mathcal{X} = \{0,1,2,\ldots\}$ with mean e^θ.*

Example 3 *Using a function $\psi_{\boldsymbol{\theta},k}(\boldsymbol{x}) = (x_1,\ldots,x_d,x_1x_2,\ldots,x_{d-1}x_d,x_1x_2x_3,\ldots)\boldsymbol{\theta}$ in which monomials of degree up to k appear, we can define a k-th order Boltzmann machine (Hinton and Sejnowski, 1986; Sejnowski, 1986).*

Example 4 *Let $\boldsymbol{x}_o \in \{+1,-1\}^{d_1}$ and $\boldsymbol{x}_h \in \{+1,-1\}^{d_2}$ be an observed vector and hidden vector, respectively, and $\boldsymbol{x} = \left(\boldsymbol{x}_o^T, \boldsymbol{x}_h^T\right) \in \{+1,-1\}^{d_1+d_2}$ where T indicates the transpose, be a concatenated vector. The Boltzmann machine with hidden variables is defined as $q_{h,\boldsymbol{\theta}}(\boldsymbol{x}_o) = \exp(\psi_{h,\boldsymbol{\theta}}(\boldsymbol{x}_o))$ where the function $\psi_{h,\boldsymbol{\theta}}(\boldsymbol{x}_o)$ is*

$$\psi_{h,\boldsymbol{\theta}}(\boldsymbol{x}_o) = \log \sum_{\boldsymbol{x}_h} \exp(\psi_{\boldsymbol{\theta},2}(\boldsymbol{x})) \tag{3}$$

and $\sum_{\boldsymbol{x}_h}$ denotes the summation with respect to the hidden variable $\boldsymbol{x}_h$.

Example 5 *Let $\boldsymbol{x}_o \in \{+1,-1\}^{d_1}$ and $\boldsymbol{x}_h \in \{+1,-1\}^{d_2}$ be the observed vector and hidden vector, respectively. The restricted Boltzmann machine is written as*

$$\begin{aligned} q_{\boldsymbol{\theta}}(\boldsymbol{x}_o) &= \sum_{\boldsymbol{x}_h} \exp\left(\boldsymbol{\theta}_o^T\boldsymbol{x}_o + \boldsymbol{\theta}_h^T\boldsymbol{x}_h + \boldsymbol{x}_h^T\boldsymbol{\theta}_{h,o}\boldsymbol{x}_o\right) \\ &= \exp\{\boldsymbol{\theta}_o^T\boldsymbol{x}_o\} \prod_{k=1}^{d_2} \left\{ e^{(\boldsymbol{\theta}_h+\boldsymbol{\theta}_{h,o}\boldsymbol{x}_o)_k} + e^{-(\boldsymbol{\theta}_h+\boldsymbol{\theta}_{h,o}\boldsymbol{x}_o)_k} \right\}, \end{aligned} \tag{4}$$

where $\boldsymbol{\theta}_o \in R^{d_1}, \boldsymbol{\theta}_h \in R^{d_2}$ are vectors of parameters, and $\boldsymbol{\theta}_{h,o} \in R^{d_2\times d_1}$ is a matrix of parameters. The index k in the above equation denotes the k-th element of the vector. Note that some parameters in the Boltzmann machine with hidden variables (3) *are restricted to* 0.

Suppose that a dataset $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^n$ generated from an underlying distribution $p(\boldsymbol{x})$ is observed. Let $\mathcal{Z}$ be a set of all patterns in the dataset $\mathcal{D}$. An empirical distribution $\tilde{p}(\boldsymbol{x})$ associated with the dataset $\mathcal{D}$ is defined as

$$\tilde{p}(\boldsymbol{x}) = \begin{cases} \frac{n_{\boldsymbol{x}}}{n} & \boldsymbol{x} \in \mathcal{Z}, \\ 0 & \text{otherwise}, \end{cases}$$

where $n_{\boldsymbol{x}}$ is the number of patterns $\boldsymbol{x}$ in the dataset $\mathcal{D}$. To estimate the parameter $\boldsymbol{\theta}$ of the probabilistic model $\bar{q}_{\boldsymbol{\theta}}$, the MLE defined by

$$\hat{\boldsymbol{\theta}}_{\mathrm{mle}} = \operatorname*{argmax}_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$$

is frequently used, where

$$L(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log \bar{q}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

is the log-likelihood of the parameter $\boldsymbol{\theta}$ using the normalized model $\bar{q}_{\boldsymbol{\theta}}$. The MLE is asymptotically consistent and efficient. However, the optimization of the log-likelihood function can be computationally demanding for normalized models on huge discrete sample spaces. Indeed, the gradient of $L(\boldsymbol{\theta})$ includes $\langle \tilde{p}\psi'_{\boldsymbol{\theta}} \rangle - \langle \bar{q}_{\boldsymbol{\theta}}\psi'_{\boldsymbol{\theta}} \rangle$, where $\psi'_{\boldsymbol{\theta}} = \frac{\partial \psi_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$. While the first term $\langle \tilde{p}\psi'_{\boldsymbol{\theta}} \rangle$ is the empirical mean that is easily calculated, the second term $\langle \bar{q}_{\boldsymbol{\theta}}\psi'_{\boldsymbol{\theta}} \rangle$ requires 2^d times summation for $\mathcal{X} = \{+1, -1\}^d$. Therefore, the gradient-based optimization is computationally infeasible when d is large. To resolve this problem, we propose a novel estimator using ideas of a homogeneous divergence and an empirical localization in the following section.

3. Homogeneous Divergences for Statistical Inference

A divergence is an extension of the squared distance and is often used in statistical inference. The formal definition of the divergence $D(f, g)$ is a non-negative valued function on $\mathcal{M} \times \mathcal{M}$ or $\mathcal{P} \times \mathcal{P}$ such that $D(f, f) = 0$ holds for arbitrary f. Many popular divergences such as the Kullback-Leilber (KL) divergence defined on $\mathcal{P} \times \mathcal{P}$ enjoy the *coincidence axiom*, i.e., $D(f, g) = 0$ leads to $f = g$. The parameter in the statistical model $\bar{q}_{\boldsymbol{\theta}}$ is estimated by minimizing the divergence $D(\tilde{p}, \bar{q}_{\boldsymbol{\theta}})$ with respect to $\boldsymbol{\theta}$.

In statistical inference using unnormalized models, the coincidence axiom of the divergence is not suitable since the probability and the unnormalized model generally do not exactly match. Our purpose is to estimate the underlying distribution up to a constant factor using unnormalized models. Hence, a divergence that has the property of the weak coincidence axiom, i.e., $D(f, g) = 0$ if and only if $g = cf$ for some $c > 0$, is a good candidate. For a class of divergences having the weak coincidence axiom, we focus on homogeneous divergences that satisfy the equality $D(f, g) = D(f, cg)$ for any $f, g \in \mathcal{M}$ and any $c > 0$.

A representative of homogeneous divergences is the pseudo-spherical (PS) divergence (Good, 1971), or in other words, γ–divergence (Fujisawa and Eguchi, 2008), that is defined from the (reverse) Hölder inequality. For a positive constant γ and all non-negative functions f, g in $\mathcal{M}$, the Hölder inequality

$$\left\langle f^{1+\gamma} \right\rangle^{\frac{1}{1+\gamma}} \left\langle g^{1+\gamma} \right\rangle^{\frac{\gamma}{1+\gamma}} \geq \langle fg^{\gamma} \rangle \quad (\gamma > 0)$$

holds. Also for a negative constant $\gamma(\neq -1)$, the reverse Hölder inequality

$$\left\langle f^{1+\gamma} \right\rangle^{\frac{1}{1+\gamma}} \left\langle g^{1+\gamma} \right\rangle^{\frac{\gamma}{1+\gamma}} \leq \langle fg^{\gamma} \rangle \quad (\gamma < 0, \gamma \neq -1)$$

holds. The above inequalities become an equality if and only if f and g are linearly dependent. From the standard and reverse Hölder inequalities, the PS-divergence $D_\gamma(f, g)$ for $f, g \in \mathcal{M}$ is defined as

$$D_\gamma(f, g) = \mathrm{sgn}(\gamma) \left\{ \frac{1}{1+\gamma} \log \left\langle f^{1+\gamma} \right\rangle + \frac{\gamma}{1+\gamma} \log \left\langle g^{1+\gamma} \right\rangle - \log \langle fg^{\gamma} \rangle \right\} \quad (\gamma \neq 0, -1), \quad (5)$$

where the sign function $\mathrm{sgn}(\gamma)$ takes 1 for $\gamma > 0$ and -1 for $\gamma < 0$. The PS divergence is homogeneous, and the Hölder inequalities ensure the non-negativity and the weak coincidence axiom of the PS-divergence. A scaled PS-divergence converges to the extended KL-divergence defined on $\mathcal{M} \times \mathcal{M}$, as $\gamma \to 0$. Fujisawa and Eguchi (2008) used the PS-divergence with $\gamma > 0$ to obtain a robust estimator in parametric statistical inference.

The PS-divergence $D_\gamma(f, g)$ from the empirical distribution $f = \tilde{p}$ to the unnormalized model $g = q_{\boldsymbol{\theta}}$ is written as

$$D_\gamma(\tilde{p}, q_{\boldsymbol{\theta}}) = Const + \mathrm{sgn}(\gamma) \frac{\gamma}{1+\gamma} \log \sum_{\boldsymbol{x} \in \mathcal{X}} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1+\gamma} - \mathrm{sgn}(\gamma) \log \sum_{\boldsymbol{x} \in \mathcal{Z}} \frac{n_{\boldsymbol{x}}}{n} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{\gamma} \quad (6)$$

and computation of the second term is infeasible in our setup. To avoid such expensive computation, some approximation techniques have been proposed such as the MCMC. Here, we employ a new trick called "empirical localization." The empirical localization is a way to transform the model $q_{\boldsymbol{\theta}}$, and its idea is to localize the domain $\mathcal{X}$ of $q_{\boldsymbol{\theta}}$ to $\mathcal{Z}$ by considering an extension of the geometric mean with the empirical distribution $\tilde{p}$. The empirical localization of $q_{\boldsymbol{\theta}}$ with $\tilde{p}$ is defined by

$$\tilde{p}(\boldsymbol{x})^{\alpha} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1-\alpha} = \begin{cases} \left(\frac{n_{\boldsymbol{x}}}{n}\right)^{\alpha} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1-\alpha} & \boldsymbol{x} \in \mathcal{Z} \\ 0 & \text{otherwise,} \end{cases} \quad (7)$$

where α is a non-negative real number. The empirical localization forces the transformed function to be zero on a domain $\mathcal{X} \setminus \mathcal{Z}$ of unobserved points, satisfying $\tilde{p}(\boldsymbol{x}) = 0$. By using the localization trick, the total sum $\langle q_{\boldsymbol{\theta}}^{\gamma+1} \rangle$ is replaced with a quantity similar to the empirical mean,

$$\left\langle \tilde{p}^{\alpha} q_{\boldsymbol{\theta}}^{1-\alpha} \right\rangle = \sum_{\boldsymbol{x} \in \mathcal{Z}} \left(\frac{n_{\boldsymbol{x}}}{n}\right)^{\alpha} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1-\alpha}. \quad (8)$$

The above quantity is easy to compute unless the sample size is extremely large. If the sample size is large, a sub-sampling technique for obtaining an approximate of the empirical mean can be used. Note that the empirical localization (7) is not defined when the value of α is negative and the empirical distribution $\tilde{p}(\boldsymbol{x})$ is zero at some points. However, we can formally employ the trick of empirical localization (8) in such cases by ignoring a domain of unobserved points. Thus, in the following, we assume that α is a non-zero real number.

Remark 2 *The summation in* (8) *is defined on* $\mathcal{Z}$ *and is then computable even when* $\alpha < 0$. *Also the summation includes only* $\mathcal{Z}(\leq n)$ *terms, and its computational cost is* $\mathcal{O}(n)$.

In addition, for convenience of notations, we define an e-mixture model $r_{\alpha,\boldsymbol{\theta}}$ ($\tilde{r}_{\alpha,\boldsymbol{\theta}}$) of the unnormalized model (2) and p ($\tilde{p}$) with ratio $\alpha \in \mathbb{R}$, as follows (Amari and Nagaoka, 2000).

$$r_{\alpha,\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{p(\boldsymbol{x})^{\alpha} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1-\alpha}}{\left\langle p^{\alpha} q_{\boldsymbol{\theta}}^{1-\alpha} \right\rangle}, \quad \tilde{r}_{\alpha,\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{\tilde{p}(\boldsymbol{x})^{\alpha} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1-\alpha}}{\left\langle \tilde{p}^{\alpha} q_{\boldsymbol{\theta}}^{1-\alpha} \right\rangle}.$$

Note that $\tilde{r}_{\alpha,\boldsymbol{\theta}}$ is a normalized version of the empirical localization (7). If p or q takes zero, the parameter α is properly restricted.

Remark 3 *We observe that* $r_{0,\boldsymbol{\theta}}(\boldsymbol{x}) = \tilde{r}_{0,\boldsymbol{\theta}}(\boldsymbol{x}) = \bar{q}_{\boldsymbol{\theta}}(\boldsymbol{x})$, $r_{1,\boldsymbol{\theta}}(\boldsymbol{x}) = p(\boldsymbol{x})$, $\tilde{r}_{1,\boldsymbol{\theta}}(\boldsymbol{x}) = \tilde{p}(\boldsymbol{x})$. *Also, if* $p(\boldsymbol{x}) = \bar{q}_{\boldsymbol{\theta}_0}(\boldsymbol{x})$, $r_{\alpha,\boldsymbol{\theta}_0}(\boldsymbol{x}) = \bar{q}_{\boldsymbol{\theta}_0}(\boldsymbol{x})$ *holds for an arbitrary* α.

To use the trick of empirical localization, we define the *localized PS-divergence* $S_{\alpha,\alpha',\gamma}(p,q)$ for the probability distribution $p \in \mathcal{P}$ and the unnormalized model $q \in \mathcal{M}$ by considering the homogeneous divergence between $f = (p^\alpha q^{1-\alpha})^{\frac{1}{1+\gamma}}$ and $g = (p^{\alpha'} q^{1-\alpha'})^{\frac{1}{1+\gamma}}$ where α, α' are two distinct real numbers:

$$\begin{aligned} S_{\alpha,\alpha',\gamma}(p,q) &= D_\gamma((p^\alpha q^{1-\alpha})^{1/(1+\gamma)}, (p^{\alpha'} q^{1-\alpha'})^{1/(1+\gamma)}) \\ &= \mathrm{sgn}(\gamma) \left\{ \frac{1}{1+\gamma} \log \left\langle p^\alpha q^{1-\alpha} \right\rangle + \frac{\gamma}{1+\gamma} \log \langle p^{\alpha'} q^{1-\alpha'} \rangle - \log \left\langle p^{\bar{\alpha}} q^{1-\bar{\alpha}} \right\rangle \right\}, \end{aligned} \tag{9}$$

where $\bar{\alpha} = (\alpha + \gamma\alpha')/(1+\gamma)$. Note that the localized PS-divergence vanishes if and only if $p^\alpha q^{1-\alpha} \propto p^{\alpha'} q^{1-\alpha'}$, i.e., $q \propto p$. Substituting the empirical distribution $\tilde{p}$ into p, the total sum over $\mathcal{X}$ is replaced with a variant of the empirical mean (8) on $\mathcal{Z}$.

Since $S_{\alpha,\alpha',\gamma}(p,q) = S_{\alpha',\alpha,1/\gamma}(p,q)$ holds, we can assume $\alpha > \alpha'$. Also we observe that for $\gamma < -1$

$$S_{\alpha,\alpha',\gamma}(p,q) = \frac{\gamma}{1+\gamma} S_{\alpha, \frac{\alpha+\gamma\alpha'}{1+\gamma}, -1-\gamma}(p,q),$$

and for $-1 < \gamma < 0$

$$S_{\alpha,\alpha',\gamma}(p,q) = \frac{1}{1+\gamma} S_{\alpha', \frac{\alpha+\gamma\alpha'}{1+\gamma}, -\gamma}(p,q),$$

respectively. Hence, we can assume $\gamma > 0$ without loss of generality. In summary, the conditions of the real parameters α, α', γ in $S_{\alpha,\alpha',\gamma}$ are given by

$$0 < \gamma, \quad \alpha > \alpha', \ \alpha \neq 0, \ \alpha' \neq 0, \ \alpha + \gamma\alpha' \neq 0,$$

where the last condition implies $\bar{\alpha} \neq 0$.

Let us consider another aspect of the computational issue of the localized PS-divergence (9). For the probability distribution p and the unnormalized exponential model $q_{\boldsymbol{\theta}}$, we show that the localized PS-divergence $S_{\alpha,\alpha',\gamma}(p, q_{\boldsymbol{\theta}})$ is convex in $\boldsymbol{\theta}$ when the parameters α, α' and γ are properly chosen.

Theorem 4 *Let* $p \in \mathcal{P}$ *be any probability distribution and* $q_{\boldsymbol{\theta}}$ *be the unnormalized exponential model* $q_{\boldsymbol{\theta}}(\boldsymbol{x}) = \exp\left(\boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{x})\right)$, *where* $\boldsymbol{\phi}(\boldsymbol{x})$ *is a vector-valued function corresponding to the sufficient statistic in the (normalized) exponential model* $\bar{q}_{\boldsymbol{\theta}}$. *When* α, α' *and* γ *satisfy* $\bar{\alpha} = (\alpha + \gamma\alpha')/(1+\gamma) = 1$, *the localized PS-divergence* $S_{\alpha,\alpha',\gamma}(p, q_{\boldsymbol{\theta}})$ *is convex in* $\boldsymbol{\theta}$. *Otherwise, there exist* p *and* $\boldsymbol{\phi}(\boldsymbol{x})$ *such that* $S_{\alpha,\alpha',\gamma}(p, q_{\boldsymbol{\theta}})$ *is not convex in* $\boldsymbol{\theta}$.

The proof of Theorem 4 is found in Appendix A.

When the value of α is negative and the probability p vanishes at some points, the total sum $\left\langle p^\alpha q^{1-\alpha} \right\rangle$ is not formally defined. We can avoid such a situation by setting both α and $\alpha\prime$ to positive values and observe the following proposition indicating that the result in Theorem 4 holds even if both α and α' are assumed to be positive.

Proposition 5 *Even though* α *and* α' *in Theorem 4 are both restricted to positive numbers, we need* $\bar{\alpha} = 1$ *to guarantee the convexity of the localized PS-divergence* $S_{\alpha,\alpha',\gamma}(p, q_{\boldsymbol{\theta}})$ *in* $\boldsymbol{\theta}$ *for any* p *and any* $\boldsymbol{\phi}(\boldsymbol{x})$.

The proof is found in Appendix B.

The localized PS-divergence with $\bar{\alpha} = 1$ characterized by Theorem 4 is denoted as $S_{\alpha,\alpha'}(p,q)$ where

$$S_{\alpha,\alpha'}(p,q) = \frac{1-\alpha'}{\alpha-\alpha'} \log \left\langle p^\alpha q^{1-\alpha} \right\rangle + \frac{\alpha-1}{\alpha-\alpha'} \log \langle p^{\alpha'} q^{1-\alpha'} \rangle. \tag{10}$$

for $\alpha > 1 > \alpha' \neq 0$. The parameter α' can be negative if the probability function p does not take zero on $\mathcal{X}$. Clearly, $S_{\alpha,\alpha'}(p,q)$ satisfies the homogeneity and the weak coincidence axiom as well as $S_{\alpha,\alpha',\gamma}(p,q)$.

The generalized Hölder's inequality admits an extension of the localized PS-divergence. For $f_1, \ldots, f_L \in \mathcal{M}$, the generalized Hölder's inequality

$$\left\langle \prod_{\ell=1}^L f_\ell^{\delta_\ell} \right\rangle \leq \prod_{\ell=1}^L \langle f_\ell \rangle^{\delta_\ell} \tag{11}$$

holds, where $\delta_1, \ldots, \delta_L$ satisfy $0 < \delta_\ell$ and $\sum_\ell \delta_\ell = 1$. The inequality becomes an equality if and only if all $f_1, \ldots, f_L$ are proportional to a non-negative function $g \in \mathcal{M}$. For the sake of completeness, the proof of the generalized Hölder's inequality is shown in Appendix C.

Substituting $p^{\alpha_\ell} q^{1-\alpha_\ell}$ into f_ℓ of the generalized Hölder's inequality, we obtain an extension of the localized PS-divergence defined as

$$S_{\boldsymbol{\alpha},\boldsymbol{\delta}}(p,q) = \sum_{\ell=1}^L \delta_\ell \log \left\langle p^{\alpha_\ell} q^{1-\alpha_\ell} \right\rangle - \log \left\langle p^{\bar{\alpha}} q^{1-\bar{\alpha}} \right\rangle \tag{12}$$

with $\bar{\alpha} = \sum_\ell \alpha_\ell \delta_\ell$. Theorem 4 can be extended for $S_{\boldsymbol{\alpha},\boldsymbol{\delta}}(p,q)$. We omit the proof since it is straightforward.

4. Estimation with Localized Pseudo-Spherical Divergences

Given the empirical distribution $\tilde{p}$ and the unnormalized model $q_{\boldsymbol{\theta}}$, we define a novel estimator $\hat{\boldsymbol{\theta}}$ with the localized PS-divergence $S_{\alpha,\alpha',\gamma}$ or $S_{\alpha,\alpha'}$:

$$\begin{aligned} \hat{\boldsymbol{\theta}} &= \operatorname*{argmin}_{\boldsymbol{\theta}} S_{\alpha,\alpha',\gamma}(\tilde{p}, q_{\boldsymbol{\theta}}) \\ &= \operatorname*{argmin}_{\boldsymbol{\theta}} \Big\{ \frac{1}{1+\gamma} \log \sum_{\boldsymbol{x} \in \mathcal{Z}} \left(\frac{n_{\boldsymbol{x}}}{n} \right)^\alpha q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1-\alpha} + \frac{\gamma}{1+\gamma} \log \sum_{\boldsymbol{x} \in \mathcal{Z}} \left(\frac{n_{\boldsymbol{x}}}{n} \right)^{\alpha'} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1-\alpha'} \\ &\qquad - \log \sum_{\boldsymbol{x} \in \mathcal{Z}} \left(\frac{n_{\boldsymbol{x}}}{n} \right)^{\bar{\alpha}} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1-\bar{\alpha}} \Big\}. \end{aligned} \tag{13}$$

Though the localized PS-divergence plugged-in the empirical distribution is not well-defined when $\alpha' < 0$ and $\tilde{p}(\boldsymbol{x}) = 0$, we can formally define the estimator by restricting the domain $\mathcal{X}$ to the observed set of examples $\mathcal{Z}$, even for such the case, as shown in Remark 2.

Proposition 6 *For the unnormalized model* (2), *the estimator* (13) *is Fisher consistent.*

Proof We observe

$$\frac{\partial}{\partial \boldsymbol{\theta}} S_{\alpha,\alpha',\gamma}(\bar{q}_{\boldsymbol{\theta}_0}, q_{\boldsymbol{\theta}})\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} = \left(\bar{\alpha} - \frac{\alpha+\gamma\alpha'}{1+\gamma}\right)\left\langle \bar{q}_{\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle = 0$$

implying the Fisher consistency of $\hat{\boldsymbol{\theta}}$. ■

Example 6 *For the Bernoulli distribution on $\mathcal{X}=\{+1,-1\}$, the estimator* (13) *is equivalent to the MLE, i.e., $\frac{1}{2}\log\frac{n_{+1}}{n_{-1}}$.*

Theorem 7 *Let $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ be the unnormalized model* (2)*, and $\boldsymbol{\theta}_0$ be the true parameter of the underlying distribution $p(\boldsymbol{x}) = \bar{q}_{\boldsymbol{\theta}_0}(\boldsymbol{x})$. Then, the asymptotic distribution of the estimator* (13) *is the normal distribution given as*

$$\sqrt{n}(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0) \sim \mathcal{N}(\mathbf{0}, I(\boldsymbol{\theta}_0)^{-1}),$$

where $I(\boldsymbol{\theta}_0)$ is the Fisher information matrix of the normalized model $\bar{q}_{\boldsymbol{\theta}_0}(\boldsymbol{x})$.

The proof is shown in Appendix D. The estimator defined from the general localized PS-divergence $S_{\boldsymbol{\alpha},\boldsymbol{\delta}}(\tilde{p}, q_{\boldsymbol{\theta}})$ defined by (12) has the same asymptotic property.

Remark 8 *The asymptotic distribution of* (13) *is equal to that of the MLE, and its variance does not depend on α, α', γ.*

Remark 9 *As shown in Remark 1, the normalized model* (1) *is a special case of the unnormalized model* (2) *and then Theorem 7 holds for the normalized model.*

5. Characterization of $S_{\alpha,\alpha'}$

Let us consider theoretical properties of the localized PS-divergence. In the following subsections, we discuss an influence of selection of α, α' and a characterization of $S_{\alpha,\alpha'}$ defined by (10).

5.1 Influence of Selection of α, α'

We investigate influence of selection of α, α' for the localized PS-divergence $S_{\alpha,\alpha'}$ with a view of the estimating equation. The estimator $\hat{\boldsymbol{\theta}}$ derived from $S_{\alpha,\alpha'}$ satisfies

$$\frac{\partial S_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}})}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}} \propto \left\langle \tilde{r}_{\alpha',\hat{\boldsymbol{\theta}}}\psi'_{\hat{\boldsymbol{\theta}}}\right\rangle - \left\langle \tilde{r}_{\alpha,\hat{\boldsymbol{\theta}}}\psi'_{\hat{\boldsymbol{\theta}}}\right\rangle = 0. \tag{14}$$

which is a moment matching with respect to two distributions $\tilde{r}_{\alpha,\boldsymbol{\theta}}$ and $\tilde{r}_{\alpha',\boldsymbol{\theta}}$ $(\alpha, \alpha' \neq 0, 1)$. On the other hand, the estimating equation of the MLE is written as

$$\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_{mle}} \propto \left\langle \tilde{p}\psi'_{\boldsymbol{\theta}_{mle}}\right\rangle - \left\langle \bar{q}_{\boldsymbol{\theta}_{mle}}\psi_{\boldsymbol{\theta}_{mle}}\right\rangle = \left\langle \tilde{r}_{1,\boldsymbol{\theta}_{mle}}\psi'_{\boldsymbol{\theta}_{mle}}\right\rangle - \left\langle \tilde{r}_{0,\boldsymbol{\theta}_{mle}}\psi'_{\boldsymbol{\theta}_{mle}}\right\rangle = 0, \tag{15}$$

which is a moment matching with respect to the empirical distribution $\tilde{p} = \tilde{r}_{1,\boldsymbol{\theta}_{mle}}$ and the normalized model $\bar{q}_{\boldsymbol{\theta}} = \tilde{r}_{0,\boldsymbol{\theta}_{mle}}$. While the localized PS-divergence $S_{\alpha,\alpha'}$ is not defined for $(\alpha, \alpha') = (0, 1)$, comparison of (14) with (15) implies that behavior the estimator $\hat{\boldsymbol{\theta}}$ is expected to be similar to that of the MLE in the limit of $\alpha \to 1$ and $\alpha' \to 0$.

5.2 Relationship with α-Divergence

The α-divergence between two positive measures $f, g \in \mathcal{M}$ is defined as

$$D_\alpha(f, g) = \frac{1}{\alpha(1-\alpha)}\left\langle \alpha f + (1-\alpha)g - f^\alpha g^{1-\alpha}\right\rangle.$$

Note that $D_\alpha(f, g) \geq 0$ and 0 if and only if $f = g$, and the α-divergence reduces to $\mathrm{KL}(f, g)$ and $\mathrm{KL}(g, f)$ in the limit of $\alpha \to 1$ and 0, respectively. See Amari and Nagaoka (2000) for details of the α-divergences.

Remark 10 *The estimator defined by minimizing α-divergence $D_\alpha(\tilde{p}, \bar{q}_{\boldsymbol{\theta}})$ between the empirical distribution $\tilde{p}$ and normalized model $\bar{q}$ satisfies*

$$\frac{\partial D_\alpha(\tilde{p}, \bar{q}_{\boldsymbol{\theta}})}{\partial \boldsymbol{\theta}} \propto \left\langle \tilde{p}^\alpha q_{\boldsymbol{\theta}}^{1-\alpha}\left(\psi'_{\boldsymbol{\theta}} - \left\langle \bar{q}_{\boldsymbol{\theta}}\psi'_{\boldsymbol{\theta}}\right\rangle\right)\right\rangle = 0,$$

which requires the computation of the normalizing constant. The same holds when unnormalized models are used. Hence, naive application of the α-divergences does not resolve the computational issue.

Here, we assume that $\alpha, \alpha' \neq 0, 1$ and consider a trick to cancel out the term $\langle g\rangle$ by mixing two α-divergences as follows.

$$\begin{aligned} D_{\alpha,\alpha'}(f, g) =& D_\alpha(f, g) + \left(\frac{-\alpha'}{\alpha}\right) D_{\alpha'}(f, g) \\ =& \left\langle \left(\frac{1}{1-\alpha} - \frac{\alpha'}{\alpha(1-\alpha')}\right) f - \frac{1}{\alpha(1-\alpha)} f^\alpha g^{1-\alpha} + \frac{1}{\alpha(1-\alpha')} f^{\alpha'} g^{1-\alpha'}\right\rangle. \end{aligned}$$

Note that $D_{\alpha,\alpha'}(f, g) \geq 0$ is divergence when $\alpha\alpha' < 0$ holds, i.e., $D_{\alpha,\alpha'}(f, g) \geq 0$ and $D_{\alpha,\alpha'}(f, g) = 0$ if and only if $f = g$. Without loss of generality, we assume that $\alpha' < 0 < \alpha \neq 1$ holds for the parameter of $D_{\alpha,\alpha'}$.

Firstly, let us consider the estimator obtained by the minimizer of $D_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}})$,

$$\begin{aligned} \hat{\boldsymbol{\theta}} &= \operatorname*{argmin}_{\boldsymbol{\theta}} D_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}}) \\ &= \operatorname*{argmin}_{\boldsymbol{\theta}} \sum_{\boldsymbol{x}\in\mathcal{Z}} \frac{1}{\alpha}\left\{\frac{1}{1-\alpha'}\left(\frac{n_{\boldsymbol{x}}}{n}\right)^{\alpha'} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1-\alpha'} - \frac{1}{1-\alpha}\left(\frac{n_{\boldsymbol{x}}}{n}\right)^{\alpha} q_{\boldsymbol{\theta}}(\boldsymbol{x})^{1-\alpha}\right\}. \end{aligned} \tag{16}$$

Note that the summation in (16) includes only $\mathcal{Z}(\leq n)$ terms. Let $\bar{q}_{\boldsymbol{\theta}_0}(\boldsymbol{x})$ be the underlying distribution and $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ be the unnormalized model (2). Then the above estimator is not Fisher consistent, i.e.,

$$\frac{\partial D_{\alpha,\alpha'}(\bar{q}_{\boldsymbol{\theta}_0}, q_{\boldsymbol{\theta}})}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} \propto \left\langle \bar{q}_{\boldsymbol{\theta}_0}^{\alpha'} q_{\boldsymbol{\theta}_0}^{1-\alpha'}\psi'_{\boldsymbol{\theta}_0} - \bar{q}_{\boldsymbol{\theta}_0}^{\alpha} q_{\boldsymbol{\theta}_0}^{1-\alpha}\psi'_{\boldsymbol{\theta}_0}\right\rangle = \left(\langle q_{\boldsymbol{\theta}_0}\rangle^{-\alpha'} - \langle q_{\boldsymbol{\theta}_0}\rangle^{-\alpha}\right)\left\langle q_{\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle \neq 0.$$

Hence, the estimator associated with $D_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}})$ does not have suitable properties such as (asymptotic) unbiasedness and consistency, while computational cost is drastically reduced. Intuitively, this is because the divergence $D_{\alpha,\alpha'}$ satisfies the coincidence axiom, i.e., $D_{\alpha,\alpha'}(f, g) = 0$ leads to $f = g$.

The statistical consistency is recovered by employing a homogeneous divergence derived from $D_{\alpha,\alpha'}$. Let us consider the estimator defined by

$$(\hat{\boldsymbol{\theta}}, \hat{z}) = \operatorname*{argmin}_{\boldsymbol{\theta}, z>0} D_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}}/z), \tag{17}$$

where $\hat{z}$ is an estimator of the normalization term $Z_{\boldsymbol{\theta}} = \langle q_{\boldsymbol{\theta}} \rangle$.

Proposition 11 *Let $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ be the unnormalized model* (2)*. Then, the minimum solution of*

$$\min_{\boldsymbol{\theta}} \min_{z>0} D_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}}/z)$$

with respect to $\boldsymbol{\theta}$ is equal to that of

$$\operatorname{sgn}(\alpha-1) S_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}}),$$

where $S_{\alpha,\alpha'}$ is formally defined for (α, α') such that $\alpha' < 0 < \alpha \neq 1$.

Proof For a given $\boldsymbol{\theta}$, we observe that

$$\hat{z} = \operatorname*{argmin}_{z>0} D_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}}/z) = \left(\frac{\left\langle \tilde{p}^{\alpha'} q_{\boldsymbol{\theta}}^{1-\alpha'} \right\rangle}{\left\langle \tilde{p}^{\alpha} q_{\boldsymbol{\theta}}^{1-\alpha} \right\rangle} \right)^{\frac{1}{\alpha-\alpha'}}. \tag{18}$$

Note that computation of (18) requires only sample order $\mathcal{O}(n)$ calculation. Then, we have

$$\begin{aligned} \min_{z>0} D_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}}/z) &= \frac{\alpha-\alpha'}{\alpha(1-\alpha')(\alpha-1)} \left(\left\langle \tilde{p}^{\alpha} q_{\boldsymbol{\theta}}^{1-\alpha} \right\rangle^{\frac{1-\alpha'}{\alpha-\alpha'}} \left\langle \tilde{p}^{\alpha'} q_{\boldsymbol{\theta}}^{1-\alpha'} \right\rangle^{\frac{\alpha-1}{\alpha-\alpha'}} - 1 \right) \\ &= \frac{\alpha-\alpha'}{\alpha(1-\alpha')(\alpha-1)} \left(e^{S_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}})} - 1 \right). \end{aligned} \tag{19}$$

for $\alpha' < 0 < \alpha \neq 1$. ■

If $\alpha' < 0$ and $1 < \alpha$ hold, the estimator using a homogeneous variant of $D_{\alpha,\alpha'}$ is equivalent to the estimator associated with the localized PS-divergence $S_{\alpha,\alpha'}$. For α and α' such that $\alpha' < 0 < \alpha < 1$, one can verify that $\operatorname{sgn}(\alpha-1) S_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}})$ is not convex in the parameter $\boldsymbol{\theta}$ of the unnormalized exponential model $q_{\boldsymbol{\theta}}(\boldsymbol{x}) = \exp\{\boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{x})\}$, though $\operatorname{sgn}(\alpha-1) S_{\alpha,\alpha'}$ is still a homogeneous divergence. In any case, the mixture of α-divergences is expressed by $S_{\alpha,\alpha'}$.

From a viewpoint of the information geometry (Amari and Nagaoka, 2000), α-divergences induce the Fisher metric that is an information geometrical structure on statistical manifolds. In other word, the Hessian matrix of the α-divergences is nothing but the Fisher information matrix. This implies that the estimation based on the (mixture of) α-divergences is Fisher efficient and is an intuitive explanation of Theorem 7. The localized PS divergences, $S_{\alpha,\alpha',\gamma}$ and $S_{\alpha,\alpha'}$, and its extension $S_{\boldsymbol{\alpha},\boldsymbol{\delta}}$ in (12) can be interpreted as an extension of the α-divergences while keeping the Fisher efficiency.

We can estimate the normalization constant $Z_{\hat{\boldsymbol{\theta}}}$ by the optimal solution $\hat{z}$ of (18).

Theorem 12 *The asymptotic distribution of the estimator* (17) *is given as a degenerated multivariate normal distribution,*

$$\sqrt{n}\,(\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0,\, \log \hat{z} - \log Z_{\boldsymbol{\theta}_0}) \;\sim\; N(\mathbf{0}, V),$$

where V is written as

$$V = \begin{pmatrix} I_{\boldsymbol{\theta}_0}^{-1} & I_{\boldsymbol{\theta}_0}^{-1} (\log Z_{\boldsymbol{\theta}_0})' \\ (\log Z_{\boldsymbol{\theta}_0})'^T I_{\boldsymbol{\theta}_0}^{-1} & (\log Z_{\boldsymbol{\theta}_0})'^T I_{\boldsymbol{\theta}_0}^{-1} (\log Z_{\boldsymbol{\theta}_0})' \end{pmatrix}.$$

Proof is shown in Appendix E.

6. Experiments

We especially focus on a setting of $\bar{\alpha} = 1$, i.e., convexity of the risk function with the unnormalized model $\exp(\boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{x}))$ holds (Theorem 4), and examined the performance of the proposed estimator.

6.1 Fisher Efficiency

We investigate the Fisher efficiency of the estimator using the localized PS-divergence. In numerical experiments, training samples were generated from the Poisson distribution having the probability function

$$p_\theta(x) = \frac{e^{x\theta - e^\theta}}{x!}, \quad \theta \in \mathbb{R}$$

for $x = 0, 1, 2, \ldots$, where θ is the natural parameter. The usual parameter λ of the Poisson distribution is written as $\lambda = e^\theta$ that is equal to the expectation of x. Given the i.i.d. data $x_1, \ldots, x_n$, the MLE the parameter θ is given as $\log \bar{x}$ using the empirical mean $\bar{x} = \frac{1}{n}\sum_{i=1}^n x_i$.

We used the unnormalized model defined as

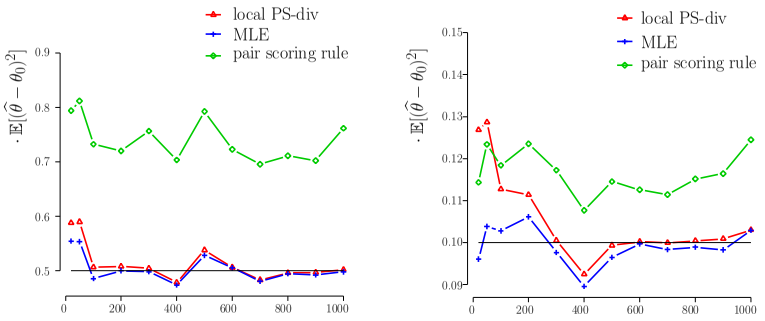
$$q_\theta(x) = \frac{e^{x\theta}}{x!}.$$

Theorem 4 ensures that the localized PS-divergence with $\bar{\alpha} = 1$ from the empirical distribution $\tilde{p}$ to the above unnormalized model, i.e., $S_{\alpha,\alpha'}(\tilde{p}, q_\theta)$, is convex in θ. In numerical experiments, the parameters of $S_{\alpha,\alpha'}$ was set to $\alpha = 1.1, \alpha' = 0.1$. In addition, the estimator using a pair scoring rule (Dawid et al., 2012),

$$\hat{\theta}_\kappa = \frac{\sum_{x\geq 0} \tilde{p}(x+1)(x+1)^{1+\kappa}}{\sum_{x\geq 0} \tilde{p}(x)(x+1)^\kappa},$$

was compared to the MLE and proposed method. Note that the above estimator with $\kappa = 0$ is nothing but the MLE. The estimator with $\kappa = 1$ was used.

Numerical results are presented in Figure 1. The horizontal axis is the number of sample size n and the vertical axis is the averaged mean square errors of the estimated parameter from the true parameter θ_0 multiplied by n, i.e., $n \cdot \mathbb{E}[(\hat{\theta} - \theta_0)^2]$. Numerically, the



$\theta_0 = \log(2)$ $\theta_0 = \log(10)$

Figure 1: Mean square errors of the estimator using localized PS-divergence, MLE, and pair scoring rule are shown. Horizontal axis is the sample size and vertical axis is mean square errors multiplied by the sample size. Left panel: $\theta_0 = \log(2)$. Right panel: $\theta_0 = \log(10)$.

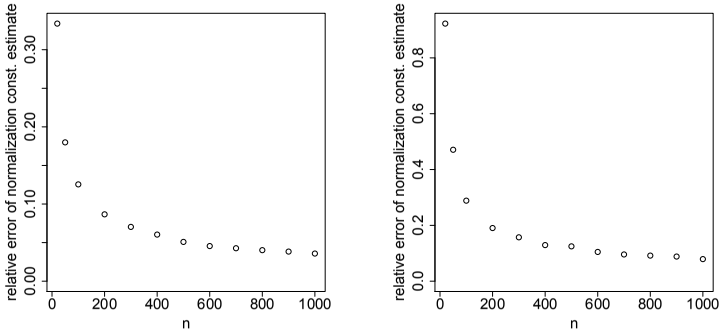
averaged mean square errors over 1000 repetitions were computed. The horizontal solid line denotes the Cramér-Rao bound. All the estimators are consistent, since the averaged mean square errors seems to be of the order $O(1/n)$. Numerically, we showed that the localized PS-divergence produces a Fisher efficient estimator, while the pair scoring rule does not achieve the Cramér-Rao bound.

When the variance of the data, i.e., e^{θ_0}, was large, the MSE of the proposed estimator for the small sample size became large. This result indicates that the regularization will be needed when a large-scale unnormalized model is used.

In addition to the parameter estimation, the normalization constant, $Z_\theta = e^{e^\theta}$, was also estimated using the equation (18). The results are presented in Fig. 2, which shows the averaged relative error of the estimator of the normalization constant, $|\hat{z} - Z_{\theta_0}|/Z_{\theta_0}$, over 1000 repetitions. Numerically, the convergence speed of the estimator (18) was approximately of the order $O(1/\sqrt{n})$.

6.2 Fully Visible Boltzmann Machine

In this subsection, we compared the proposed estimator with parameter settings $(\alpha, \alpha') = (1.01, 0.01), (1.01, -0.01), (2, -1)$, with the MLE and the ratio matching method (Hyvärinen, 2007). Note that the ratio matching method also does not require calculation of the normalization constant. In this experiment, we do not compare the proposed estimator with the pseudo-likelihood method and the contrastive divergence which also do not require the calculation of the normalization constant, because in (Hyvärinen, 2007), the ratio matching method has been numerically compared with the pseudo-likelihood method and the con-



$\theta_0 = \log(2)$ $\theta_0 = \log(10)$

Figure 2: Averaged relative errors of the estimator (18) for the normalization constant are shown. Horizontal axis is the sample size. Left panel: $\theta_0 = \log(2)$. Right panel: $\theta_0 = \log(10)$.

trastive divergence using the fully visible Boltzmann machine, and its result showed that all methods have almost equal performance. In addition, some theoretical properties of the pseudo-likelihood (and its extension, the composite likelihood) have been investigated in (Mardia et al., 2009; Kanamori, 2016), showing that the pseudo-likelihood is not Fisher efficient in general.

All methods were optimized with the *optim* function in R language (R Core Team, 2015). The dimension d of input was set to 10 and the synthetic dataset was randomly generated from the second order Boltzmann machine (Example 3) with a parameter $\boldsymbol{\theta}^* \sim \mathcal{N}(\mathbf{0}, I/d)$. We repeated comparison 50 times and observed averaged performance. Figure 3 (a) shows median of the root mean square errors (RMSEs) between $\boldsymbol{\theta}^*$ and $\hat{\boldsymbol{\theta}}$ of each method over 50 trials, against the number n of examples. We observe that the proposed estimator is comparable with the MLE as predicted by the Theorem 7, and the ratio matching is also comparable to the MLE under the setting. Figure 3 (b) shows a number of observed patterns in a dataset consists of n examples. Figure 3 (c) shows median of computational time of each method against n. The computational time of the MLE does not vary against n because the computational cost is dominated by the calculation of the normalization constant. Both the proposed estimator and the ratio matching method are significantly faster than the MLE.

We investigated performance of methods under an another situation, in which the true parameter distributes as $\boldsymbol{\theta}^* \sim \mathcal{N}(\mathbf{0}, I)$. Figure 4 (a) shows median of the root mean square errors (RMSEs) between $\boldsymbol{\theta}^*$ and $\hat{\boldsymbol{\theta}}$ of each method over 50 trials, against the number n of examples. We observe that the proposed estimator works well, but the MLE outperforms the proposed method contrary to the prediction of Theorem 7. This is because observed

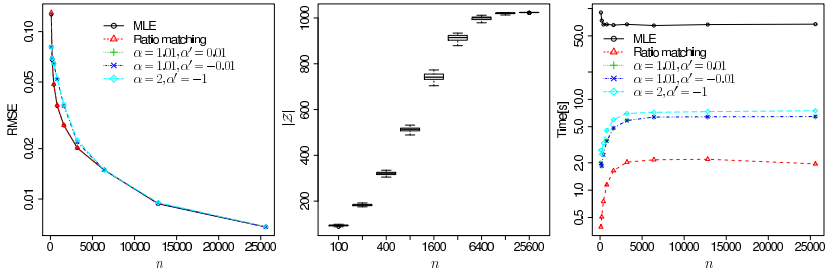


Figure 3: Results for datasets generated with a parameter $\boldsymbol{\theta}^* \sim \mathcal{N}(\mathbf{0}, I/d)$. (a) Median of RMSEs of each method against n, in log scale. (b) Box-whisker plot of number $|\mathcal{Z}|$ of unique patterns in the dataset $\mathcal{D}$ against n. (c) Median of computational time of each method against n, in log scale.

patterns were only a small portion of all possible patterns, as shown in Figure 4 (b). Even in such a case, the MLE can take all possible patterns ($2^{10} = 1024$) into account through the normalization term $Z_{\boldsymbol{\theta}}$ because the Taylor expansion of $\log Z_{\boldsymbol{\theta}}$ around $\boldsymbol{\theta} = \mathbf{0}$ which is approximated as

$$\log Z_{\boldsymbol{\theta}} \simeq d \log 2 + \frac{1}{2}||\boldsymbol{\theta}||^2$$

term behaves like a regularizer. On the other hand, the proposed method genuinely uses only the observed examples, and focuses on the restricted domain $\mathcal{Z}$ rather than the original domain $\mathcal{X}$, in which the asymptotic analysis would not be relevant in this case. Figure 4 (c) shows median of computational time of each method against n. Both the proposed estimator and the ratio matching method are significantly faster than the MLE. While the ratio matching method is faster than the proposed estimator, the RMSE of the proposed estimator is less than that of the ratio matching.

To overcome the degrade of performance of the proposed estimator caused by lack of example patterns, we consider a regularized version of the proposed estimator as,

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left\{ S_{\alpha,\alpha'}(\tilde{p}, q_{\boldsymbol{\theta}}) + \frac{\lambda}{2n}||\boldsymbol{\theta}||^2 \right\}. \tag{20}$$

Note that we can employ the l_1 regularizer to obtain a sparse estimator, rather than the l_2 regularizer. Ravikumar et al. (2010) theoretically investigated conditions for correctly selecting edges of the Boltzmann machine using the l_1-regularized logistic regression.

We investigated performance of the regularized estimator with the same dataset ($\boldsymbol{\theta}^* \sim \mathcal{N}(0, I)$). Figure 5 shows median of RMSEs and computational time of the MLE, the ratio matching, and the regularized estimator ($\alpha = 1.01, \alpha' = 0.01, \lambda = 0, 10^{-2}, 10^{-4}$). The Figure shows that performance of the proposed estimator is improved by the regularization term $||\boldsymbol{\theta}||^2$, which can be interpreted as an approximation of the normalization constant $\log Z_{\boldsymbol{\theta}}$. Note that the computational time of the regularized estimator is drastically reduced

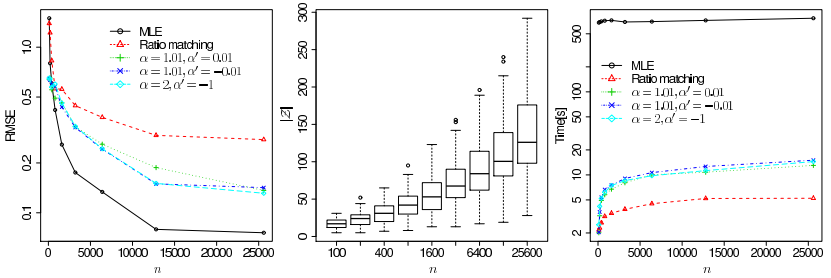


Figure 4: Results for datasets generated with a parameter $\boldsymbol{\theta}^* \sim \mathcal{N}(\mathbf{0}, I)$. (a) Median of RMSEs of each method against n, in log scale. (b) Box-whisker plot of number $|\mathcal{Z}|$ of unique patterns in the dataset $\mathcal{D}$ against n. (c) Median of computational time of each method against n, in log scale.

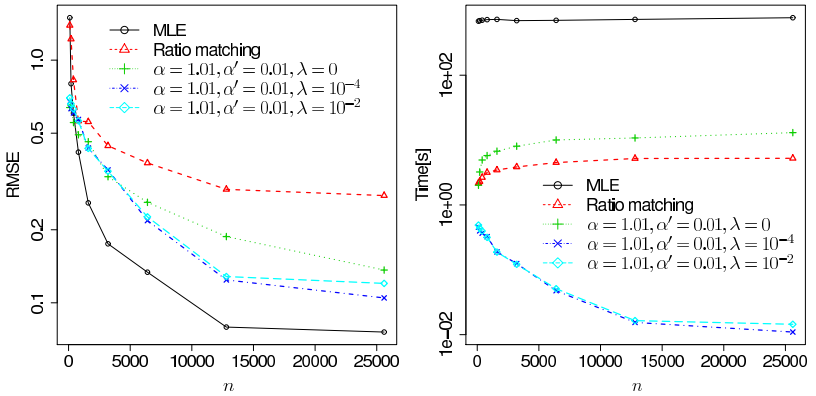


Figure 5: (a) Median of RMSE of the MLE, the ratio matching and the regularized estimator with $\alpha = 1.01, \alpha' = 0.01$, and $\lambda = 0, 10^{-2}, 10^{-4}$ against n, in log scale. (b) Median of computational time of each method against n, in log scale.

compared with that of estimator without regularization. This is because the cost function (20) becomes similar to a quadratic function and the condition number associated with the cost function is improved by the regularization, which influences required number of steps of the quasi-Newton method.

Also we investigated performance of the estimator (18) for the normalization constant $Z_{\boldsymbol{\theta}}$ and results for above two situations are shown in Figure 6. In the Figure 6, the averaged relative errors $|\hat{z} - Z_{\boldsymbol{\theta}_0}|/Z_{\boldsymbol{\theta}_0}$ of the proposed estimator ($(\alpha, \alpha') = (1.01, 0.01), (1.01, -0.01), (2, -1)$) are shown. We observe that the proposed estimator appropriately works.

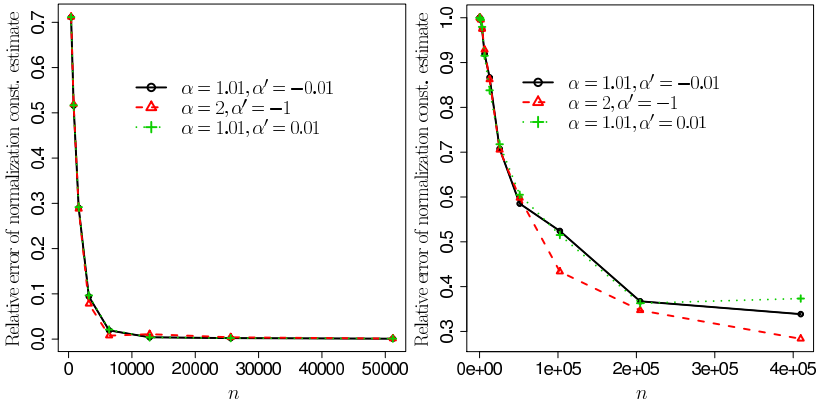


Figure 6: Median of relative errors of normalization constant estimates for two experiment scenarios: (a) $\boldsymbol{\theta}^* \sim \mathcal{N}(0, I/d)$. (b) $\boldsymbol{\theta}^* \sim \mathcal{N}(0, I)$.

6.3 Sample Complexity

We numerically investigated the sample complexity of the proposed estimator ($\alpha_1 = 1.01, \alpha' = 0.01$, without the regularization term), i.e., a number $N(\epsilon, \delta, d)$ of examples to attain

$$\Pr(\mathrm{KL}(\bar{q}_{\boldsymbol{\theta}_0}, \bar{q}_{\hat{\boldsymbol{\theta}}}) \geq \epsilon) < \delta$$

where ϵ and δ are arbitrary positive constants. For each dimension $d = 2, 3, \ldots, 21$, we generated a dataset containing $n = 50 \times 2^k (k = 1, \ldots, 9)$ examples from the fully visible Boltzmann machine and calculated the KL divergences $\mathrm{KL}(\bar{q}_{\boldsymbol{\theta}_0}, \bar{q}_{\hat{\boldsymbol{\theta}}})$. For two kinds of parameter settings, i.e., $\boldsymbol{\theta}_0 \sim \mathcal{N}(0, I/d)$ and $\boldsymbol{\theta}_0 \sim \mathcal{N}(0, I)$, we repeated these procedures 50 times and observed 50 values of KL divergence. Each panel in Figure 7 shows median of values of the KL divergence ($\delta = 0.5$) for each setting of the parameter, respectively. Both panels show that the proposed estimator requires approximately $2 \sim 3$ times as many examples examples against an increase of the dimensionality d at the same level ϵ of estimation error $\mathrm{KL}(\bar{q}_{\boldsymbol{\theta}_0}, \bar{q}_{\hat{\boldsymbol{\theta}}})$.

6.4 Boltzmann Machine with Hidden Variables

In this subsection, we applied the proposed estimator for the Boltzmann machine with hidden variables whose associated function is written as (3). The proposed estimator with parameter settings $(\alpha, \alpha') = (1.01, 0.01), (1.01, -0.01), (2, -1)$ was compared with the MLE. The dimension d_1 of observed variables was fixed to 10 and d_2 of hidden variables was set to 2, and the parameter $\boldsymbol{\theta}^*$ was generated as $\boldsymbol{\theta}^* \sim \mathcal{N}(\mathbf{0}, I)$ including parameters corresponding to hidden variables. Note that the Boltzmann machine with hidden variables is not identifiable and different values of the parameter do not necessarily generate different probability distributions, implying that estimators are influenced by local minimums. Then we

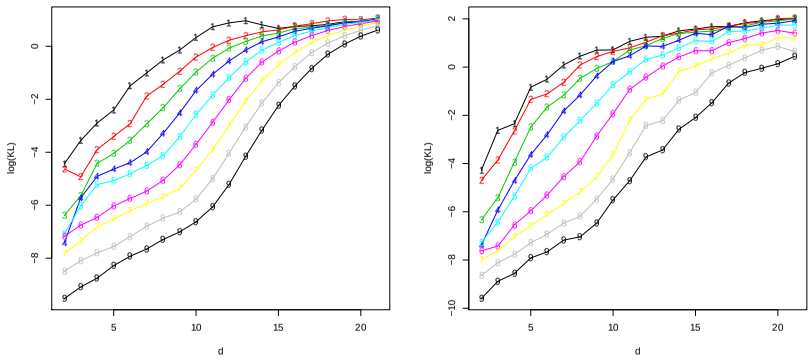


Figure 7: Median of KL divergences $\mathrm{KL}(\bar{q}_{\boldsymbol{\theta}_0}, \bar{q}_{\hat{\boldsymbol{\theta}}})$. Horizontal axis is dimension d, and vertical axis is median of KL divergences, in log-scale. Number k ($k = 1, \ldots, 9$) on a line corresponds to number $n = 50 \times 2^k$ of examples. (a) $\boldsymbol{\theta}_0 \sim \mathcal{N}(0, I/d)$. (b) $\boldsymbol{\theta}_0 \sim \mathcal{N}(0, I)$,

measured performance of each estimator by the averaged log-likelihood $\frac{1}{n}\sum_{i=1}^{n} \log \bar{q}_{\hat{\boldsymbol{\theta}}}(\boldsymbol{x}_i)$ rather than the RMSE of parameters. An initial value of the parameter was set by $\mathcal{N}(\mathbf{0}, I)$ and commonly used by all methods. We repeated the comparison 50 times and observed the averaged performance. Figure 8 (a) shows median of averaged log-likelihoods of each method for the training dataset over 50 trials, against the number n of example. We observe that the proposed estimator is comparable with the MLE when the number n of examples becomes large. Note that the averaged log-likelihood of MLE once decreases when n is small, and this is due to overfitting of the model. Figure 8 (b) shows median of averaged log-likelihoods of each method for test dataset consists of 10000 examples, over 50 trials. Figure 8 (c) shows median of computational time of each method against n, and we observe that the proposed estimator is significantly faster than the MLE.

In addition, we investigated performance of the regularized version (20) ($\alpha = 1.01, \alpha' = 0.01$ and $\lambda = 0, 10^{-2}, 10^{-4}$) and compared with the MLE. Figures 9 (a) and (b) show medians of averaged log-likelihoods of the MLE and the regularized estimator over 50 trials, for the training dataset and the test dataset, respectively. Figures imply that an appropriate regularization can improve performance of the estimator and also as the experiment in previous subsection, computational time of the regularized estimator is drastically reduced, in comparison with that of the estimator without regularization.

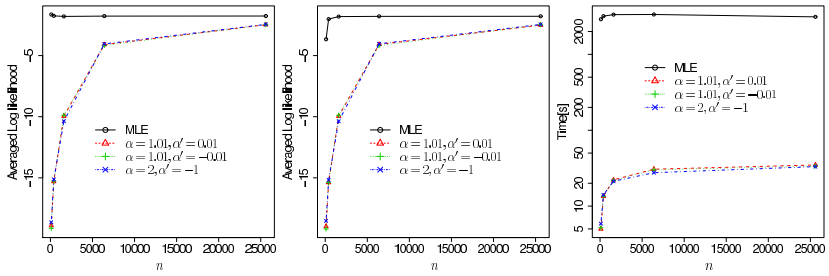


Figure 8: (a) Median of averaged log-likelihoods of each method against n. (b) Median of averaged log-likelihoods of each method calculated for test dataset against n. (c) Median of computational time of each method against n, in log scale.

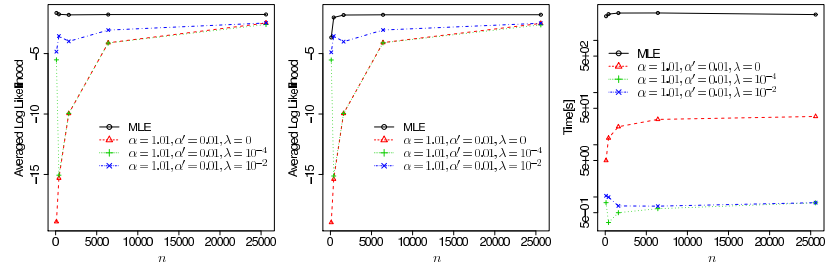


Figure 9: (a) Median of averaged log-likelihoods of the MLE and regularized estimator with $\lambda = 0, 10^{-2}, 10^{-4}$ against n. (b) Median of averaged log-likelihoods of each method calculated for test dataset against n. (c) Median of computational time of each method against n, in log scale.

7. Conclusions

We proposed a novel estimator for probabilistic models on discrete space, for which the normalization constant is infeasible to calculate. The proposed estimator is based on unnormalized models and an empirically localized PS-divergence having the homogeneous property and can be constructed without calculation of the normalization constant. We showed that homogeneous divergences with empirical localization allow the computation of the normalization constant to be avoided because of a weak coincidence axiom. The idea of empirical localization permits ignoring an unobserved domain on sample space, which can drastically reduce computational cost. We investigated statistical properties of the proposed estimator and revealed that the proposed estimator is asymptotically efficient and that its asymptotic distribution is equal to that of the maximum likelihood estimator (MLE). In addition, we showed a relationship between the empirically localized PS-divergence and a mixture of α-divergences. The Hessian matrix of the α-divergence is known to be equal

to the Fisher information matrix, which implies the Fisher efficiency of the proposed estimator. A difference between two divergences is that the mixture of α-divergence does not have homogeneous property. By utilizing this difference, we proposed an estimator for the normalization constant that requires only sample order $\mathcal{O}(n)$ calculation and is asymptotically consistent. We investigated the performance of the proposed estimator with various kinds of models on discrete sample space and showed that the proposed estimator performs comparably to the MLE, while required computational cost is drastically reduced.

A possible future direction for this work is application of the proposed framework for models on continuous space. While employment of the framework for continuous space is an interesting challenge, the empirical localization technique is difficult to apply to models on continuous space because of the power of the empirical distribution described by the delta function. Another direction is making the proposed estimator more robust. The PS-divergence used in this paper is well known to be robust against outlier noise (Kanamori and Fujisawa, 2015). The robustness of the PS-divergence can be controlled by the tuning parameter γ, which was fixed to a value ensuring convexity of the risk function in this paper. The robustification of the proposed estimator and its theoretical justification are important issues for data analysis application.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Numbers 25730018, 16K00051, 16K00044, 15H01678, and 15H03636 from MEXT, Japan.

Appendix A. Proof of Theorem 4

Some calculation yields

$$\frac{\partial^2 \log \left\langle p^{\alpha} q_{\boldsymbol{\theta}}^{1-\alpha} \right\rangle}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T} = (1-\alpha)^2 V_{r_{\alpha,\boldsymbol{\theta}}}[\boldsymbol{\phi}],$$

where $V_{r_{\alpha,\boldsymbol{\theta}}}[\boldsymbol{\phi}]$ is the covariance matrix of $\boldsymbol{\phi}(\boldsymbol{x})$ under the probability $r_{\alpha,\boldsymbol{\theta}}(\boldsymbol{x})$. Thus, the Hessian matrix of $S_{\alpha,\alpha',\gamma}(p, q_{\boldsymbol{\theta}})$ is written as

$$\frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T} S_{\alpha,\alpha',\gamma}(p, q_{\boldsymbol{\theta}}) = \frac{(1-\alpha)^2}{1+\gamma} V_{r_{\alpha,\boldsymbol{\theta}}}[\boldsymbol{\phi}] + \frac{\gamma(1-\alpha')^2}{1+\gamma} V_{r_{\alpha',\boldsymbol{\theta}}}[\boldsymbol{\phi}] - (1-\bar{\alpha})^2 V_{r_{\bar{\alpha},\boldsymbol{\theta}}}[\boldsymbol{\phi}].$$

The Hessian matrix is positive semidefinite if $\bar{\alpha} = 1$.

To prove the second part, we prove that there exists a distribution p, a model $q_{\boldsymbol{\theta}}$ and parameters α, α', γ such that the Hessian is not positive semidefinite, for a given $\bar{\alpha} \neq 1$. Suppose that $\mathcal{X} = \{+1, -1\}^d$ and the function $\boldsymbol{\phi}(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), \ldots, \phi_d(\boldsymbol{x})) \in \mathbb{R}^d$ is defined by $\phi_k(\boldsymbol{x}) = x_k$, $k = 1, \ldots, d$ for $\boldsymbol{x} = (x_1, \ldots, x_d) \in \mathcal{X}$. Let p be the uniform distribution on $\mathcal{X}$. The covariance matrix of $\boldsymbol{\phi}$ is the diagonal matrix given by

$$V_{r_{\alpha,\boldsymbol{\theta}}}[\boldsymbol{\phi}] = 4 \cdot \mathrm{diag}\left(\frac{1}{(e^{(1-\alpha)\theta_1} + e^{-(1-\alpha)\theta_1})^2}, \ldots, \frac{1}{(e^{(1-\alpha)\theta_d} + e^{-(1-\alpha)\theta_d})^2} \right).$$

Let δ be $\delta = 1/(1+\gamma)$, then $\delta \in (0,1)$ holds for $\gamma > 0$. We define

$$f(z;\theta) = \frac{(1-z)^2}{(e^{(1-z)\theta} + e^{-(1-z)\theta})^2}, \quad z, \theta \in \mathbb{R}.$$

Then, the i-th diagonal element of the Hessian matrix is expressed by

$$\Delta = \delta \cdot f(\alpha;\theta_i) + (1-\delta) \cdot f(\alpha';\theta_i) - f(\delta\alpha + (1-\delta)\alpha';\theta_i)$$

up to a positive constant. Our task is to find the parameter α, α', δ such that $\bar{\alpha} = \delta\alpha + (1-\delta)\alpha'$ and $\Delta < 0$ hold. The function f satisfies the following properties.

(a) $f(z;\theta) \geq 0$ and $f(z;\theta) = 0 \Leftrightarrow z = 1$.

(b) $f(1+\varepsilon;\theta) = f(1-\varepsilon;\theta) = f(1+\varepsilon;-\theta)$ for $\varepsilon \geq 0, \theta \in \mathbb{R}$.

(c) $\lim_{z\to\pm\infty} f(z;\theta) = 0$ holds for $\theta \neq 0$.

Let θ be a fixed non-zero real number. Since $\bar{\alpha} \neq 1$, $f(\bar{\alpha};\theta) > 0$ holds. Due to the properties (b) and (c), any sufficiently large $\varepsilon > 0$ satisfies $f(1-\varepsilon;\theta) = f(1+\varepsilon;\theta) < f(\bar{\alpha};\theta)$. Define $\alpha = 1+\varepsilon$ and $\alpha' = 1-\varepsilon$. By choosing $\delta \in (0,1)$ such that $\bar{\alpha} = \delta\alpha + (1-\delta)\alpha'$, we have $\Delta < 0$.

Appendix B. Proof of Proposition 5

Suppose that $\bar{\alpha} > 1$. Due to (a), (c) in Appendix A and the continuity of $f(z,\theta)$ at $z = 1$, there exists α and α' satisfying $1 < \alpha' < \bar{\alpha} < \alpha$ such that both $f(\alpha';\theta)$ and $f(\alpha;\theta)$ are less than $f(\bar{\alpha};\theta)$, where θ is a non-zero constant. Then, $\Delta < 0$ holds for $\delta \in (0,1)$ such that $\bar{\alpha} = \delta\alpha + (1-\delta)\alpha'$. We prove the case of $0 < \bar{\alpha} < 1$. For a sufficiently large θ, we have

$$\frac{f(0;\theta)}{f(\bar{\alpha};\theta)} = O\left(\frac{e^{-2\bar{\alpha}\theta}}{(1-\bar{\alpha})^2}\right) \to 0 \ (\theta \to \infty).$$

Hence, the continuity of f ensures that there exist a sufficiently large θ and a small positive α' such that $0 < \alpha' < \bar{\alpha}$ and $f(\alpha';\theta) < f(\bar{\alpha};\theta)$ hold. The property (c) in the Appendix A ensures that there exists a sufficiently large $\alpha > 1$ satisfying $f(\alpha;\theta) < f(\bar{\alpha};\theta)$. Again, $\Delta < 0$ holds for δ such that $\bar{\alpha} = \delta\alpha + (1-\delta)\alpha'$.

Appendix C. Generalized Hölder's inequality

This section includes the proof and the equality condition of the generalized Hölder's inequality. The inequality (11) of $L = 2$ is nothing but the standard Hölder's inequality. The inequality (11) of $L = 3$ is proved by using the standard one as follows:

$$\left\langle f_1^{\delta_1} f_2^{\delta_2} f_3^{\delta_3} \right\rangle = \left\langle (f_1^{\frac{\delta_1}{1-\delta_3}} f_2^{\frac{\delta_2}{1-\delta_3}})^{1-\delta_3} f_3^{\delta_3} \right\rangle \leq \left\langle f_1^{\frac{\delta_1}{1-\delta_3}} f_2^{\frac{\delta_2}{1-\delta_3}} \right\rangle^{1-\delta_3} \langle f_3 \rangle^{\delta_3} \leq \langle f_1 \rangle^{\delta_1} \langle f_2 \rangle^{\delta_2} \langle f_3 \rangle^{\delta_3},$$

where $\delta_1 + \delta_2 + \delta_3 = 1$ and $\delta_i > 0$ for $i = 1,2,3$ are assumed. The third inequality becomes equality if and only if f_1 and f_2 are linearly dependent, and the second inequality becomes

equality if and only if $f_1^{\frac{\delta_1}{1-\delta_3}} f_2^{\frac{\delta_2}{1-\delta_3}}$ and f_3 are linearly dependent. As a result, (11) of $L = 3$ becomes an equality if and only if there exists a function $g \in \mathcal{M}$ such that all f_1, f_2, f_3 are proportional to g. In the same way, we can prove the generalized Hölder's inequality of any natural number L.

Appendix D. Proof of Theorem 7

We show the asymptotic distribution of the estimator defined from the localized PS-divergence $S_{\alpha,\alpha',\gamma}$. Along the same line, one can prove the asymptotic property of the estimator defined from the general localized PS-divergence $S_{\boldsymbol{\alpha},\boldsymbol{\delta}}$.

Let us assume that the empirical distribution is written as

$$\tilde{p}(\boldsymbol{x}) = \bar{q}_{\boldsymbol{\theta}_0}(\boldsymbol{x}) + \epsilon \cdot s(\boldsymbol{x}),$$

where ϵ is a small positive number and $s(\boldsymbol{x})$ satisfies $\langle s \rangle = 0$. Note that $r_{\alpha,\boldsymbol{\theta}_0}(\boldsymbol{x}) = \bar{q}_{\boldsymbol{\theta}_0}(\boldsymbol{x})$. By expanding an equilibrium condition of the estimator (13) around $\boldsymbol{\theta} = \boldsymbol{\theta}_0$, we obtain

$$\begin{aligned}
0 &= \frac{\partial}{\partial \boldsymbol{\theta}} S_{\alpha,\alpha',\gamma}(\tilde{p}, q_{\boldsymbol{\theta}})\Big|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}} \\
&= \frac{\partial}{\partial \boldsymbol{\theta}} S_{\alpha,\alpha',\gamma}(\tilde{p}, q_{\boldsymbol{\theta}})\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} + \frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T} S_{\alpha,\alpha',\gamma}(\tilde{p}, q_{\boldsymbol{\theta}})\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} (\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0) + \mathcal{O}(||\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0||^2) \\
&= \left\{ \frac{1-\alpha}{1+\gamma} \left\langle \tilde{r}_{\alpha,\boldsymbol{\theta}_0} \psi'_{\boldsymbol{\theta}_0} \right\rangle + \frac{\gamma(1-\alpha')}{1+\gamma} \left\langle \tilde{r}_{\alpha',\boldsymbol{\theta}_0} \psi'_{\boldsymbol{\theta}_0} \right\rangle - (1-\bar{\alpha}) \left\langle \tilde{r}_{\bar{\alpha},\boldsymbol{\theta}_0} \psi'_{\boldsymbol{\theta}_0} \right\rangle \right\} \\
&\quad + \left\{ \frac{(1-\alpha)^2}{1+\gamma} V_{\tilde{r}_{\alpha,\boldsymbol{\theta}_0}}[\psi'_{\boldsymbol{\theta}_0}] + \frac{\gamma(1-\alpha')^2}{1+\gamma} V_{\tilde{r}_{\alpha',\boldsymbol{\theta}_0}}[\psi'_{\boldsymbol{\theta}_0}] - (1-\bar{\alpha})^2 V_{\tilde{r}_{\bar{\alpha},\boldsymbol{\theta}_0}}[\psi'_{\boldsymbol{\theta}_0}] \right. \\
&\quad \left. + \frac{1-\alpha}{1+\gamma} \left\langle \tilde{r}_{\alpha,\boldsymbol{\theta}_0} \psi''_{\boldsymbol{\theta}_0} \right\rangle + \frac{\gamma(1-\alpha')}{1+\gamma} \left\langle \tilde{r}_{\alpha',\boldsymbol{\theta}_0} \psi''_{\boldsymbol{\theta}_0} \right\rangle - (1-\bar{\alpha}) \left\langle \tilde{r}_{\bar{\alpha},\boldsymbol{\theta}_0} \psi''_{\boldsymbol{\theta}_0} \right\rangle \right\} (\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0) + \mathcal{O}(||\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0||^2),
\end{aligned}$$

where $\psi'_{\boldsymbol{\theta}_0}$ and $\psi''_{\boldsymbol{\theta}_0}$ are the gradient vector and the Hessian matrix of $\psi_{\boldsymbol{\theta}_0}(\boldsymbol{x})$ with respect to the parameter $\hat{\boldsymbol{\theta}}$. By the delta method (Van der Vaart, 1998), we observe that

$$\begin{aligned}
&(\langle \tilde{r}_{\alpha,\boldsymbol{\theta}_0} \psi'_{\boldsymbol{\theta}_0} \rangle - \langle r_{\alpha,\boldsymbol{\theta}_0} \psi'_{\boldsymbol{\theta}_0} \rangle) \\
&= \alpha\epsilon \frac{\left\langle \bar{q}_{\boldsymbol{\theta}_0}^{\alpha-1} q_{\boldsymbol{\theta}_0}^{1-\alpha} \psi'_{\boldsymbol{\theta}_0} s \right\rangle \left\langle \bar{q}_{\boldsymbol{\theta}_0}^{\alpha} q_{\boldsymbol{\theta}_0}^{1-\alpha} \right\rangle - \left\langle \bar{q}_{\boldsymbol{\theta}_0}^{\alpha} q_{\boldsymbol{\theta}_0}^{1-\alpha} \psi'_{\boldsymbol{\theta}_0} \right\rangle \left\langle \bar{q}_{\boldsymbol{\theta}_0}^{\alpha-1} q_{\boldsymbol{\theta}_0}^{1-\alpha} s \right\rangle}{\left\langle \bar{q}_{\boldsymbol{\theta}_0}^{\alpha} q_{\boldsymbol{\theta}_0}^{1-\alpha} \right\rangle^2} + O(\epsilon^2) \\
&= \alpha\epsilon \left(\langle \psi'_{\boldsymbol{\theta}_0} s \rangle - \langle \bar{q}_{\boldsymbol{\theta}_0} \psi'_{\boldsymbol{\theta}_0} \rangle \langle s \rangle \right) + O(\epsilon^2) \\
&= \alpha\epsilon \langle \psi'_{\boldsymbol{\theta}_0} s \rangle + O(\epsilon^2).
\end{aligned}$$

Then we have

$$\begin{aligned}
&\frac{\partial}{\partial\boldsymbol{\theta}}S_{\alpha,\alpha',\gamma}(\tilde{p},q_{\boldsymbol{\theta}})\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} - \frac{\partial}{\partial\boldsymbol{\theta}}S_{\alpha,\alpha',\gamma}(p,q_{\boldsymbol{\theta}})\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0}\\
&= \frac{1-\alpha}{1+\gamma}\left(\left\langle \tilde{r}_{\alpha,\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle - \left\langle r_{\alpha,\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle\right) + \frac{\gamma(1-\alpha')}{1+\gamma}\left(\left\langle \tilde{r}_{\alpha',\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle - \left\langle r_{\alpha',\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle\right)\\
&\quad - (1-\bar{\alpha})\left(\left\langle \tilde{r}_{\bar{\alpha},\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle - \left\langle r_{\bar{\alpha},\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle\right)\\
&= \left\{\frac{1-\alpha}{1+\gamma}\left\langle \tilde{r}_{\alpha,\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle + \frac{\gamma(1-\alpha')}{1+\gamma}\left\langle \tilde{r}_{\alpha',\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle - (1-\bar{\alpha})\left\langle \tilde{r}_{\bar{\alpha},\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle\right\}\\
&\quad - \left\{1 - \frac{\alpha+\gamma\alpha'}{1+\gamma} - (1-\bar{\alpha})\right\}\left\langle \bar{q}_{\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle\\
&= \left(\alpha\frac{1-\alpha}{1+\gamma} + \alpha'\frac{\gamma(1-\alpha')}{1+\gamma} - (1-\bar{\alpha})\bar{\alpha}\right)\left\langle \psi'_{\boldsymbol{\theta}_0}\epsilon s\right\rangle + O(\epsilon^2)\\
&= -\frac{\gamma}{(1+\gamma)^2}(\alpha-\alpha')^2\left\langle \psi'_{\boldsymbol{\theta}_0}(\tilde{p}-\bar{q}_{\boldsymbol{\theta}_0})\right\rangle + O(\epsilon^2).
\end{aligned}$$

From the central limit theorem,

$$\sqrt{n}\left\langle \psi'_{\boldsymbol{\theta}_0}(\tilde{p}-\bar{q}_{\boldsymbol{\theta}_0})\right\rangle = \frac{1}{\sqrt{n}}\sum_{i=1}^{n}\left(\psi'_{\boldsymbol{\theta}_0}(\boldsymbol{x}_i) - \left\langle \bar{q}_{\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle\right)$$

asymptotically follows the normal distribution with mean $\mathbf{0}$ and variance $V_{\bar{q}_{\boldsymbol{\theta}_0}}[\psi'_{\boldsymbol{\theta}_0}] = I_{\boldsymbol{\theta}_0}$. Also from the law of large numbers, we have

$$\begin{aligned}
&\frac{(1-\alpha)^2}{1+\gamma}V_{\tilde{r}_\alpha}[\psi'_{\boldsymbol{\theta}_0}] + \frac{\gamma(1-\alpha')^2}{1+\gamma}V_{\tilde{r}_{\alpha'}}[\psi'_{\boldsymbol{\theta}_0}] - (1-\bar{\alpha})^2 V_{\tilde{r}_{\bar{\alpha}}}[\psi'_{\boldsymbol{\theta}_0}] \to \frac{\gamma}{(1+\gamma)^2}(\alpha-\alpha')^2 I_{\boldsymbol{\theta}_0},\\
&\quad \frac{1-\alpha}{1+\gamma}\left\langle \tilde{r}_{\alpha,\boldsymbol{\theta}_0}\psi''_{\boldsymbol{\theta}_0}\right\rangle + \frac{\gamma(1-\alpha')}{1+\gamma}\left\langle \tilde{r}_{\alpha',\boldsymbol{\theta}_0}\psi''_{\boldsymbol{\theta}_0}\right\rangle - (1-\bar{\alpha})\left\langle \tilde{r}_{\bar{\alpha},\boldsymbol{\theta}_0}\psi''_{\boldsymbol{\theta}_0}\right\rangle\\
&\to \left(1 - \frac{\alpha+\gamma\alpha'}{1+\gamma} - (1-\bar{\alpha})\right)\left\langle \bar{q}_{\boldsymbol{\theta}_0}\psi''_{\boldsymbol{\theta}_0}\right\rangle = 0
\end{aligned}$$

in the limit of $n\to\infty$. Taking the probabilistic error in the law of large numbers into account, we obtain the equality

$$\sqrt{n}\left(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_0\right) = I_{\boldsymbol{\theta}_0}^{-1}\left\{\frac{1}{\sqrt{n}}\sum_{i=1}^{n}\left(\psi'_{\boldsymbol{\theta}_0}(\boldsymbol{x}_i) - \left\langle \bar{q}_{\boldsymbol{\theta}_0}\psi'_{\boldsymbol{\theta}_0}\right\rangle\right)\right\} + o_p(1)$$

Consequently, the asymptotic distribution of the estimator is given as

$$\sqrt{n}\left(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_0\right) \sim \mathcal{N}(\mathbf{0}, I_{\boldsymbol{\theta}_0}^{-1}).$$

Appendix E. Proof of Theorem 12

Let us consider the asymptotic property of $\hat{z}$. The logarithm of the estimator $\widehat{z}$ is expressed as

$$\log\widehat{z} = \frac{1}{\alpha-\alpha'}\log\left\langle \tilde{p}^{\alpha'}q_{\hat{\boldsymbol{\theta}}}^{1-\alpha'}\right\rangle - \frac{1}{\alpha-\alpha'}\log\left\langle \tilde{p}^{\alpha}q_{\hat{\boldsymbol{\theta}}}^{1-\alpha}\right\rangle.$$

In the same way as the calculation in Appendix D, we have

$$\log\left\langle p^{\alpha}q_{\hat{\theta}}^{1-\alpha}\right\rangle = (1-\alpha)\log Z_{\boldsymbol{\theta}_0} + (1-\alpha)(\log Z_{\boldsymbol{\theta}_0})'(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_0) + o_p(\|\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_0\|).$$

From the above equation, we have

$$\sqrt{n}(\log\hat{z} - \log Z_{\boldsymbol{\theta}_0}) = \sqrt{n}(\log Z_{\boldsymbol{\theta}_0})'^{T}(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_0) + o_p(1).$$

Therefore, the asymptotic variance of of $\log(\hat{z})$ is given by

$$n\cdot V[\log(\hat{z})] \longrightarrow (\log Z_{\boldsymbol{\theta}_0})'^{T} I_{\boldsymbol{\theta}_0}^{-1}(\log Z_{\boldsymbol{\theta}_0})'.$$

We show the correlation between $\hat{\boldsymbol{\theta}}$ and $\log\hat{z}$. The asymptotic expansion above yields

$$n\cdot(\log\widehat{z} - (\log Z_{\boldsymbol{\theta}_0}))(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_0) = (\sqrt{n}(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_0))(\sqrt{n}(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_0))^{T}(\log Z_{\boldsymbol{\theta}_0})' + o_p(1)$$

and we have

$$n\cdot\mathbb{E}[(\log\hat{z} - \log Z_{\boldsymbol{\theta}_0})(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_0)] \longrightarrow I_{\boldsymbol{\theta}_0}^{-1}(\log Z_{\boldsymbol{\theta}_0})'.$$

Therefore, the asymptotic distribution of $\sqrt{n}(\hat{\boldsymbol{\theta}}-\boldsymbol{\theta}_0, \log\hat{z} - \log Z_{\boldsymbol{\theta}_0})$ is given as the multivariate normal distribution with mean zero and the following asymptotic variance

$$\begin{aligned}
n\cdot V[(\hat{\boldsymbol{\theta}}, \log\hat{z})] &\longrightarrow \begin{pmatrix} I_{\boldsymbol{\theta}_0}^{-1} & I_{\boldsymbol{\theta}_0}^{-1}(\log Z_{\boldsymbol{\theta}_0})'\\ (\log Z_{\boldsymbol{\theta}_0})'^{T}I_{\boldsymbol{\theta}_0}^{-1} & (\log Z_{\boldsymbol{\theta}_0})'^{T}I_{\boldsymbol{\theta}_0}^{-1}(\log Z_{\boldsymbol{\theta}_0})' \end{pmatrix}\\
&= \begin{pmatrix} E\\ (\log Z_{\boldsymbol{\theta}_0})'^{T}\end{pmatrix} I_{\boldsymbol{\theta}_0}^{-1}\begin{pmatrix} E & (\log Z_{\boldsymbol{\theta}_0})'\end{pmatrix},
\end{aligned}$$

where E is the identity matrix. Note that the distribution of the estimator $(\hat{\boldsymbol{\theta}}, \hat{z})$ is asymptotically degenerated.

References

D. H. Ackley, G. E. Hinton, and T. J. Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985.

S. Amari and H. Nagaoka. *Methods of Information Geometry*, volume 191 of *Translations of Mathematical Monographs*. Oxford University Press, 2000.

Shun-ichi Amari, Koji Kurata, and Hiroshi Nagaoka. Information geometry of boltzmann machines. *Neural Networks, IEEE Transactions on*, 3(2):260–271, 1992.

A. P. Dawid, S. Lauritzen, and M. Parry. Proper local scoring rules on discrete sample spaces. *The Annals of Statistics*, 40(1):593–608, 2012.

H. Fujisawa and S. Eguchi. Robust parameter estimation with a small bias against heavy contamination. *Journal of Multivariate Analysis*, 99(9):2053–2081, 2008.

I. J. Good. Comment on "measuring information and uncertainty," by R. J. Buehler. In V. P. Godambe and D. A. Sprott, editors, *Foundations of Statistical Inference*, page 337339, Toronto: Holt, Rinehart and Winston, 1971.

M. Gutmann and J. Hirayama. Bregman divergence as general framework to estimate unnormalized statistical models. *arXiv preprint arXiv:1202.3727*, 2012.

G. E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8):1771–1800, 2002.

G. E Hinton. A practical guide to training restricted boltzmann machines. *Momentum*, 9 (1):926, 2010.

G. E. Hinton and R. R. Salakhutdinov. A better way to pretrain deep Boltzmann machines. In *Advances in Neural Information Processing Systems*, pages 2447–2455, 2012.

G. E. Hinton and T. J. Sejnowski. Learning and relearning in Boltzmann machines. *MIT Press, Cambridge, Mass*, 1:282–317, 1986.

A. Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–708, 2005.

A. Hyvärinen. Some extensions of score matching. *Computational statistics & data analysis*, 51(5):2499–2512, 2007.

T. Kanamori and H. Fujisawa. Robust estimation under heavy contamination using unnormalized models. *Biometrika*, 102(3):559–572, 2015.

T. Kanamori. Efficiency bound of local z-estimators on discrete sample spaces. *Entropy*, 18 (7):273, 2016.

K. V. Mardia, J. T. Kent, G. Hughes, and C. C. Taylor. Maximum likelihood estimation using composite likelihoods for closed exponential families. *Biometrika*, 96(4):975–982, 2009.

M. Opper and D. Saad, editors. *Advanced Mean Field Methods: Theory and Practice*. MIT Press, Cambridge, MA, 2001.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015. URL `http://www.R-project.org/`.

P. Ravikumar, M. J. Wainwright, J. D. Lafferty. High-dimensional ising model selection using 1-regularized logistic regression. *The Annals of Statistics*, 38(3):1287–1319, 2010.

T. J. Sejnowski. Higher-order Boltzmann machines. In *American Institute of Physics Conference Series*, volume 151, pages 398–403, 1986.

T. Takenouchi. A novel parameter estimation method for boltzmann machines. *Neural computation*, 27(11):2423–2446, 2015.

T. Takenouchi and T. Kanamori. Empirical localization of homogeneous divergences on discrete sample spaces. In *Advances in Neural Information Processing Systems*, pages 820–828, 2015.

A. W. Van der Vaart. *Asymptotic Statistics*. Cambridge University Press, 1998.

Journal of Machine Learning Research 18 (2017) 1-59 Submitted 1/16; Revised 4/17; Published 7/17

Density Estimation in Infinite Dimensional Exponential Families

Bharath Sriperumbudur bks18@psu.edu
Department of Statistics, Pennsylvania State University
University Park, PA 16802, USA.

Kenji Fukumizu fukumizu@ism.ac.jp
The Institute of Statistical Mathematics
10-3 Midoricho, Tachikawa, Tokyo 190-8562 Japan.

Arthur Gretton arthur.gretton@gmail.com
ORCID 0000-0003-3169-7624
Gatsby Computational Neuroscience Unit, University College London
Sainsbury Wellcome Centre, 25 Howland Street, London W1T 4JG, UK

Aapo Hyvärinen a.hyvarinen@ucl.ac.uk
Gatsby Computational Neuroscience Unit, University College London
Sainsbury Wellcome Centre, 25 Howland Street, London W1T 4JG, UK

Revant Kumar rkumar74@gatech.edu
College of Computing, Georgia Institute of Technology
801 Atlantic Drive, Atlanta, GA 30332, USA.

Editor: Ingo Steinwart

Abstract

In this paper, we consider an infinite dimensional exponential family $\mathcal{P}$ of probability densities, which are parametrized by functions in a reproducing kernel Hilbert space $\mathcal{H}$, and show it to be quite rich in the sense that a broad class of densities on $\mathbb{R}^d$ can be approximated arbitrarily well in Kullback-Leibler (KL) divergence by elements in $\mathcal{P}$. Motivated by this approximation property, the paper addresses the question of estimating an unknown density p_0 through an element in $\mathcal{P}$. Standard techniques like maximum likelihood estimation (MLE) or pseudo MLE (based on the method of sieves), which are based on minimizing the KL divergence between p_0 and $\mathcal{P}$, do not yield practically useful estimators because of their inability to efficiently handle the log-partition function. We propose an estimator $\hat{p}_n$ based on minimizing the *Fisher divergence*, $J(p_0\|p)$ between p_0 and $p \in \mathcal{P}$, which involves solving a simple finite-dimensional linear system. When $p_0 \in \mathcal{P}$, we show that the proposed estimator is consistent, and provide a convergence rate of $n^{-\min\left\{\frac{2}{3}, \frac{2\beta+1}{2\beta+2}\right\}}$ in Fisher divergence under the smoothness assumption that $\log p_0 \in \mathcal{R}(C^\beta)$ for some $\beta \geq 0$, where C is a certain Hilbert-Schmidt operator on $\mathcal{H}$ and $\mathcal{R}(C^\beta)$ denotes the image of C^β. We also investigate the misspecified case of $p_0 \notin \mathcal{P}$ and show that $J(p_0\|\hat{p}_n) \to \inf_{p\in\mathcal{P}} J(p_0\|p)$ as $n \to \infty$, and provide a rate for this convergence under a similar smoothness condition as above. Through numerical simulations we demonstrate that the proposed estimator outperforms the non-parametric kernel density estimator, and that the advantage of the proposed estimator grows as d increases.

©2017 Bharath Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Aapo Hyvärinen and Revant Kumar.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-011.html.

Keywords: density estimation, exponential family, Fisher divergence, kernel density estimator, maximum likelihood, interpolation space, inverse problem, reproducing kernel Hilbert space, Tikhonov regularization, score matching

1. Introduction

Exponential families are among the most important classes of parametric models studied in statistics, and include many common distributions such as the normal, exponential, gamma, and Poisson. In its "natural form", the family generated by a probability density q_0 (defined over $\Omega \subseteq \mathbb{R}^d$) and *sufficient statistic*, $T : \Omega \to \mathbb{R}^m$ is defined as

$$\mathscr{P}_{\text{fin}} := \left\{ p_\theta(x) = q_0(x) e^{\theta^T T(x) - A(\theta)},\, x \in \Omega \,:\, \theta \in \Theta \subset \mathbb{R}^m \right\} \tag{1}$$

where $A(\theta) := \log \int_\Omega e^{\theta^T T(x)} q_0(x)\, dx$ is the cumulant generating function (also called the log-partition function), $\Theta \subset \{\theta \in \mathbb{R}^m : A(\theta) < \infty\}$ is the *natural parameter space* and θ is a finite-dimensional vector called the *natural parameter*. Exponential families have a number of properties that make them extremely useful for statistical analysis (see Brown, 1986 for more details).

In this paper, we consider an infinite dimensional generalization (Canu and Smola, 2005; Fukumizu, 2009) of (1),

$$\mathcal{P} = \left\{ p_f(x) = e^{f(x) - A(f)} q_0(x),\, x \in \Omega : f \in \mathcal{F} \right\},$$

where the function space $\mathcal{F}$ is defined as

$$\mathcal{F} = \left\{ f \in \mathcal{H} : e^{A(f)} < \infty \right\}, \text{ with } A(f) := \log \int_\Omega e^{f(x)} q_0(x)\, dx$$

being the cumulant generating function, and $(\mathcal{H}, \langle \cdot, \cdot \rangle_{\mathcal{H}})$ a reproducing kernel Hilbert space (RKHS) (Aronszajn, 1950) with k as its reproducing kernel. While various generalizations are possible for different choices of $\mathcal{F}$ (e.g., an Orlicz space as in Pistone and Sempi, 1995), the connection of $\mathcal{P}$ to the natural exponential family in (1) is particularly enlightening when $\mathcal{H}$ is an RKHS. This is due to the reproducing property of the kernel, $f(x) = \langle f, k(x, \cdot) \rangle_{\mathcal{H}}$, through which $k(x, \cdot)$ takes the role of the sufficient statistic. In fact, it can be shown (see Section 3 and Example 1 for more details) that every $\mathscr{P}_{\text{fin}}$ is generated by $\mathcal{P}$ induced by a finite dimensional RKHS $\mathcal{H}$, and therefore the family $\mathcal{P}$ with $\mathcal{H}$ being an infinite dimensional RKHS is a natural infinite dimensional generalization of $\mathscr{P}_{\text{fin}}$. Furthermore, this generalization is particularly interesting as in contrast to $\mathscr{P}_{\text{fin}}$, it can be shown that $\mathcal{P}$ is a rich class of densities (depending on the choice of k and therefore $\mathcal{H}$) that can approximate a broad class of probability densities arbitrarily well (see Propositions 1, 13 and Corollary 2). This generalization is not only of theoretical interest, but also has implications for statistical and machine learning applications. For example, in Bayesian non-parametric density estimation, the densities in $\mathcal{P}$ are chosen as prior distributions on a collection of probability densities (e.g., see van der Vaart and van Zanten, 2008). $\mathcal{P}$ has also found applications in nonparametric hypothesis testing (Gretton et al., 2012; Fukumizu et al., 2008) and dimensionality reduction (Fukumizu et al., 2004, 2009) through the mean and

covariance operators, which are obtained as the first and second Fréchet derivatives of $A(f)$ (see Fukumizu, 2009, Section 1.2.3). Recently, the infinite dimensional exponential family, $\mathcal{P}$ has been used to develop a gradient-free adaptive MCMC algorithm based on Hamiltonian Monte Carlo (Strathmann et al., 2015) and also has been used in the context of learning the structure of graphical models (Sun et al., 2015).

Motivated by the richness of the infinite dimensional generalization and its statistical applications, it is of interest to model densities by $\mathcal{P}$, and therefore the goal of this paper is to estimate unknown densities by elements in $\mathcal{P}$ when $\mathcal{H}$ is an infinite dimensional RKHS. Formally, given i.i.d. random samples $(X_a)_{a=1}^n$ drawn from an unknown density p_0, the goal is to estimate p_0 through $\mathcal{P}$. Throughout the paper, we refer to case of $p_0 \in \mathcal{P}$ as *well-specified*, in contrast to the *misspecified* case where $p_0 \notin \mathcal{P}$. The setting is useful because $\mathcal{P}$ is a rich class of densities that can approximate a broad class of probability densities arbitrarily well, hence it may be widely used in place of non-parametric density estimation methods (e.g., kernel density estimation (KDE)). In fact, through numerical simulations, we show in Section 6 that estimating p_0 through $\mathcal{P}$ performs better than KDE, and that the advantage of the proposed estimator grows with increasing dimensionality.

In the finite-dimensional case where $\theta \in \Theta \subset \mathbb{R}^m$, estimating p_θ through maximum likelihood (ML) leads to solving elegant likelihood equations (Brown, 1986, Chapter 5). However, in the infinite dimensional case (assuming $p_0 \in \mathcal{P}$), as in many non-parametric estimation methods, a straightforward extension of maximum likelihood estimation (MLE) suffers from the problem of ill-posedness (Fukumizu, 2009, Section 1.3.1). To address this problem, Fukumizu (2009) proposed a method of sieves involving pseudo-MLE by restricting the infinite dimensional manifold $\mathcal{P}$ to a series of finite-dimensional submanifolds, which enlarge as the sample size increases, i.e., $p_{\hat{f}^{(l)}}$ is the density estimator with

$$\hat{f}^{(l)} = \arg\max_{f \in \mathcal{F}^{(l)}} \frac{1}{n} \sum_{a=1}^{n} f(X_a) - A(f), \tag{2}$$

where $\mathcal{F}^{(l)} = \{f \in \mathcal{H}^{(l)} : e^{A(f)} < \infty\}$ and $(\mathcal{H}^{(l)})_{l=1}^{\infty}$ is a sequence of finite-dimensional subspaces of $\mathcal{H}$ such that $\mathcal{H}^{(l)} \subset \mathcal{H}^{(l+1)}$ for all $l \in \mathbb{N}$. While the consistency of $p_{\hat{f}^{(l)}}$ is proved in Kullback-Leibler (KL) divergence (Fukumizu, 2009, Theorem 6), the method suffers from many drawbacks that are both theoretical and computational in nature. On the theoretical front, the consistency in Fukumizu (2009, Theorem 6) is established by assuming a decay rate on the eigenvalues of the covariance operator (see (A-2) and the discussion in Section 1.4 of Fukumizu (2009) for details), which is usually difficult to check in practice. Moreover, it is not clear which classes of RKHS should be used to obtain a consistent estimator (Fukumizu, 2009, (A-1)) and the paper does not provide any discussion about the convergence rates. On the practical side, the estimator is not attractive as it can be quite difficult to construct the sequence $(\mathcal{H}^{(l)})_{l=1}^{\infty}$ that satisfies the assumptions in Fukumizu (2009, Theorem 6). In fact, the impracticality of the estimator, $\hat{f}^{(l)}$ is accentuated by the difficulty in efficiently handling $A(f)$ (though it can be approximated by numerical integration).

A related work was carried out by Barron and Sheu (1991)—also see references therein—where the goal is to estimate a density, p_0 by approximating its logarithm as an expansion in terms of basis functions, such as polynomials, splines or trigonometric series. Similar to

Fukumizu (2009), Barron and Sheu proposed the ML estimator $p_{\hat{f}_m}$, where

$$\hat{f}_m = \arg\max_{f \in \mathcal{F}_m} \frac{1}{n} \sum_{a=1}^{n} f(X_a) - A(f)$$

and $\mathcal{F}_m$ is the linear space of dimension m spanned by the chosen basis functions. Under the assumption that $\log p_0$ has square-integrable derivatives up to order r, they showed that $KL(p_0 \| p_{\hat{f}_m}) = O_{p_0}(n^{-2r/(2r+1)})$ with $m = n^{1/(2r+1)}$ for each of the approximating families, where $KL(p\|q) = \int p(x) \log(p(x)/q(x))\, dx$ is the KL divergence between p and q. Similar work was carried out by Gu and Qiu (1993), who assumed that $\log p_0$ lies in an RKHS, and proposed an estimator based on penalized MLE, with consistency and rates established in Jensen-Shannon divergence. Though these results are theoretically interesting, these estimators are obtained via a procedure similar to that in Fukumizu (2009), and therefore suffers from the practical drawbacks discussed above.

The discussion so far shows that the MLE approach to learning $p_0 \in \mathcal{P}$ results in estimators that are of limited practical interest. To alleviate this, one can treat the problem of estimating $p_0 \in \mathcal{P}$ in a completely non-parametric fashion by using KDE, which is well-studied (Tsybakov, 2009, Chapter 1) and easy to implement. This approach ignores the structure of $\mathcal{P}$, however, and is known to perform poorly for moderate to large d (Wasserman, 2006, Section 6.5) (see also Section 6 of this paper).

1.1 Score Matching and Fisher Divergence

To counter the disadvantages of KDE and pseudo/penalized-MLE, in this paper, we propose to use the *score matching method* introduced by Hyvärinen (2005, 2007). While MLE is based on minimizing the KL divergence, the score matching method involves minimizing the *Fisher divergence* (also called the Fisher information distance; see Definition 1.13 in Johnson (2004)) between two continuously differentiable densities, p and q on an open set $\Omega \subseteq \mathbb{R}^d$, given as

$$J(p\|q) = \frac{1}{2} \int_\Omega p(x) \left\| \nabla \log p(x) - \nabla \log q(x) \right\|_2^2 dx, \tag{3}$$

where $\nabla \log p(x) = (\partial_1 \log p(x), \ldots, \partial_d \log p(x))$ with $\partial_i \log p(x) := \frac{\partial}{\partial x_i} \log p(x)$. Fisher divergence is closely related to the KL divergence through de Bruijn's identity (Johnson, 2004, Appendix C) and it can be shown that $KL(p\|q) = \int_0^\infty J(p_t\|q_t)\, dt$, where $p_t = p * N(0, tI_d)$, $q_t = q * N(0, tI_d)$, $*$ denotes the convolution, and $N(0, tI_d)$ denotes a normal distribution on $\mathbb{R}^d$ with mean zero and diagonal covariance with $t > 0$ (see Proposition B.1 for a precise statement; also see Theorem 1 in Lyu, 2009). Moreover, convergence in Fisher divergence is a stronger form of convergence than that in KL, total variation and Hellinger distances (see Lemmas E.2 & E.3 in Johnson, 2004 and Corollary 5.1 in Ley and Swan, 2013).

To understand the advantages associated with the score matching method, let us consider the problem of density estimation where the data generating distribution (say p_0) belongs to $\mathscr{P}_{\text{fin}}$ in (1). In other words, given random samples $(X_a)_{a=1}^n$ drawn i.i.d. from $p_0 := p_{\theta_0}$, the goal is to estimate θ_0 as $\hat{\theta}_n$, and use $p_{\hat{\theta}_n}$ as an estimator of p_0. While the

MLE approach is well-studied and enjoys nice statistical properties in asymptopia (i.e., asymptotically unbiased, efficient, and normally distributed), the computation of $\hat{\theta}_n$ can be intractable in many situations as discussed above. In particular, this is the case for $p_\theta(x) = \frac{r_\theta(x)}{A(\theta)}$ where $r_\theta \geq 0$ for all $\theta \in \Theta$, $A(\theta) = \int_\Omega r_\theta(x)\,dx$, and the functional form of r is known (as a function of θ and x); yet we do not know how to easily compute A, which is often analytically intractable. In this setting (which is exactly the setting of this paper), assuming p_θ to be differentiable (w.r.t. x), and $\int_\Omega p_0(x)\|\nabla \log p_\theta(x)\|_2^2\,dx < \infty, \forall\, \theta \in \Theta$, $J(p_0\|p_\theta) =: J(\theta)$ in (3) reduces to

$$J(\theta) = \sum_{i=1}^{d} \int_\Omega p_0(x) \left(\frac{1}{2} \left(\partial_i \log p_\theta(x)\right)^2 + \partial_i^2 \log p_\theta(x) \right) dx + \frac{1}{2} \int_\Omega p_0(x) \left\|\nabla \log p_0(x)\right\|_2^2 dx, \tag{4}$$

through integration by parts (see Hyvärinen, 2005, Theorem 1), under appropriate regularity conditions on p_0 and p_θ for all $\theta \in \Theta$. Here $\partial_i^2 \log p_\theta(x) := \frac{\partial^2}{\partial x_i^2} \log p_\theta(x)$. The main advantage of the objective in (3) (and also (4)) is that when it is applied to the situation discussed above where $p_\theta(x) = \frac{r_\theta(x)}{A(\theta)}$, $J(\theta)$ is independent of $A(\theta)$, and an estimate of θ_0 can be obtained by simply minimizing the empirical counterpart of $J(\theta)$, given by

$$J_n(\theta) := \frac{1}{n} \sum_{a=1}^{n} \sum_{i=1}^{d} \left(\frac{1}{2} \left(\partial_i \log p_\theta(X_a)\right)^2 + \partial_i^2 \log p_\theta(X_a) \right) + \frac{1}{2} \int_\Omega p_0(x) \left\|\nabla \log p_0(x)\right\|_2^2 dx.$$

Since $J_n(\theta)$ is also independent of $A(\theta)$, $\hat{\theta}_n = \arg\min_{\theta\in\Theta} J_n(\theta)$ may be easily computable, unlike the MLE. We would like to highlight that while the score matching approach may have computational advantages over MLE, it only estimates p_θ up to the scaling factor $A(\theta)$, and therefore requires the approximation or computation of $A(\theta)$ through numerical integration to estimate p_θ. Note that this issue (of computing $A(\theta)$ through numerical integration) exists even with MLE, but not with KDE. In score matching, however, numerical integration is needed only once, while MLE would typically require a functional form of the log-partition function which is approximated through numerical integration at every step of an iterative optimization algorithm (for example, see (2)), thus leading to major computational savings. An important application that does not require the computation of $A(\theta)$ is in finding modes of the distribution, which has recently become very popular in image processing (Comaniciu and Meer, 2002), and has already been investigated in the score matching framework (Sasaki et al., 2014). Similarly, in sampling methods such as sequential Monte Carlo (Doucet et al., 2001), it is often the case that the evaluation of unnormalized densities is sufficient to calculate required importance weights.

1.2 Contributions

(i) We present an estimate of $p_0 \in \mathcal{P}$ in the well-specified case through the minimization of Fisher divergence, in Section 4. First, we show that estimating $p_0 := p_{f_0}$ using the score matching method reduces to estimating f_0 by solving a simple finite-dimensional linear system (Theorems 4 and 5). Hyvärinen (2007) obtained a similar result for $\mathscr{P}_{\text{fin}}$ where the estimator is obtained by solving a linear system, which in the case of Gaussian family

matches the MLE (Hyvärinen, 2005). The estimator obtained in the infinite dimensional case is not a simple extension of its finite-dimensional counterpart, however, as the former requires an appropriate regularizer (we use $\|\cdot\|_{\mathcal{H}}^2$) to make the problem well-posed. We would like to highlight that to the best of our knowledge, the proposed estimator is the first practically computable estimator of p_0 with consistency guarantees (see below).

(ii) In contrast to Hyvärinen (2007) where no guarantees on consistency or convergence rates are provided for the density estimator in $\mathscr{P}_{\text{fin}}$, we establish in Theorem 6 the consistency and rates of convergence for the proposed estimator of f_0, and use these to prove consistency and rates of convergence for the corresponding plug-in estimator of p_0 (Theorems 7 and B.2), even when $\mathcal{H}$ is infinite dimensional. Furthermore, while the estimator of f_0 (and therefore p_0) is obtained by minimizing the Fisher divergence, the resultant density estimator is also shown to be consistent in KL divergence (and therefore in Hellinger and total-variation distances) and we provide convergence rates in all these distances.

Formally, we show that the proposed estimator $\hat{f}_n$ is converges as

$$\|f_0 - \hat{f}_n\|_{\mathcal{H}} = O_{p_0}(n^{-\alpha}),\ KL(p_0\|p_{\hat{f}_n}) = O_{p_0}(n^{-2\alpha}) \text{ and } J(p_0\|p_{\hat{f}_n}) = O_{p_0}\left(n^{-\min\left\{\frac{2}{3}, \frac{2\beta+1}{2\beta+2}\right\}}\right)$$

if $f_0 \in \mathcal{R}(C^\beta)$ for some $\beta > 0$, where $\mathcal{R}(A)$ denotes the range or image of an operator A, $\alpha = \min\{\frac{1}{4}, \frac{\beta}{2\beta+2}\}$, and $C := \sum_{i=1}^d \int_\Omega \partial_i k(x,\cdot) \otimes \partial_i k(x,\cdot)\, p_0(x)\,dx$ is a Hilbert-Schmidt operator on $\mathcal{H}$ (see Theorem 4) with k being the reproducing kernel and $\otimes$ denoting the tensor product. When $\mathcal{H}$ is a finite-dimensional RKHS, we show that the estimator enjoys parametric rates of convergence, i.e.,

$$\|f_0 - \hat{f}_n\|_{\mathcal{H}} = O_{p_0}(n^{-1/2}),\ KL(p_0\|p_{\hat{f}_n}) = O_{p_0}(n^{-1}) \text{ and } J(p_0\|p_{\hat{f}_n}) = O_{p_0}(n^{-1}).$$

Note that the convergence rates are obtained under a non-classical smoothness assumption on f_0, namely that it lies in the image of certain fractional power of C, which reduces to a more classical assumption if we choose k to be a Matérn kernel (see Section 2 for its definition), as it induces a Sobolev space. In Section 4.2, we discuss in detail the smoothness assumption on f_0 for the Gaussian (Example 2) and Matérn (Example 3) kernels. Another interesting point to observe is that unlike in the classical function estimation methods (e.g., kernel density estimation and regression), the rates presented above for the proposed estimator tend to saturate for $\beta > 1$ ($\beta > \frac{1}{2}$ w.r.t. J), with the best rate attained at $\beta = 1$ ($\beta = \frac{1}{2}$ w.r.t. J), which means the smoothness of f_0 is not fully captured by the estimator. Such a saturation behavior is well-studied in the inverse problem literature (Engl et al., 1996) where it has been attributed to the choice of regularizer. In Section 4.3, we discuss alternative regularization strategies using ideas from Bauer et al. (2007), which covers non-parametric least squares regression: we show that for appropriately chosen regularizers, the above mentioned rates hold for any $\beta > 0$, and do not saturate for the aforementioned ranges of β (see Theorem 9).

(iii) In Section 5, we study the problem of density estimation in the misspecified setting, i.e., $p_0 \notin \mathcal{P}$, which is not addressed in Hyvärinen (2007) and Fukumizu (2009). Using a more sophisticated analysis than in the well-specified case, we show in Theorem 12 that $J(p_0\|p_{\hat{f}_n}) \to \inf_{p\in\mathcal{P}} J(p_0\|p)$ as $n \to \infty$. Under an appropriate smoothness assumption

on $\log \frac{p_0}{q_0}$ (see the statement of Theorem 12 for details), we show that $J(p_0 \| p_{\hat{f}_n}) \to 0$ as $n \to \infty$ along with a rate for this convergence, even though $p_0 \notin \mathcal{P}$. However, unlike in the well-specified case, where the consistency is obtained not only in J but also in other distances, we obtain convergence only in J for the misspecified case. Note that while Barron and Sheu (1991) considered the estimation of p_0 in the misspecified setting, the results are restricted to the approximating families consisting of polynomials, splines, or trigonometric series. Our results are more general, as they hold for abstract RKHSs.

(iv) In Section 6, we present preliminary numerical results comparing the proposed estimator with KDE in estimating a Gaussian and mixture of Gaussians, with the goal of empirically evaluating performance as d gets large for a fixed sample size. In these two estimation problems, we show that the proposed estimator outperforms KDE, and the advantage grows as d increases. Inspired by this preliminary empirical investigation, our proposed estimator (or computationally efficient approximations) has been used by Strathmann et al. (2015) in a gradient-free adaptive MCMC sampler, and by Sun et al. (2015) for graphical model structure learning. These applications demonstrate the practicality and performance of the proposed estimator.

Finally, we would like to make clear that our principal goal is not to construct density estimators that improve uniformly upon KDE, but to provide a novel flexible modeling technique for approximating an unknown density by a rich parametric family of densities, with the parameter being infinite dimensional, in contrast to the classical approach of finite dimensional approximation.

Various notations and definitions that are used throughout the paper are collected in Section 2. The proofs of the results are provided in Section 8, along with some supplementary results in an appendix.

2. Definitions & Notation

We introduce the notation used throughout the paper. Define $[d] := \{1, \ldots, d\}$. For $a := (a_1, \ldots, a_d) \in \mathbb{R}^d$ and $b := (b_1, \ldots, b_d) \in \mathbb{R}^d$, $\|a\|_2 := \sqrt{\sum_{i=1}^d a_i^2}$ and $\langle a, b \rangle := \sum_{i=1}^d a_i b_i$. For $a, b > 0$, we write $a \lesssim b$ if $a \leq \gamma b$ for some positive universal constant γ. For a topological space $\mathcal{X}$, $C(\mathcal{X})$ (*resp.* $C_b(\mathcal{X})$) denotes the space of all continuous (*resp.* bounded continuous) functions on $\mathcal{X}$. For a locally compact Hausdorff space $\mathcal{X}$, $f \in C(\mathcal{X})$ is said to *vanish at infinity* if for every $\epsilon > 0$ the set $\{x : |f(x)| \geq \epsilon\}$ is compact. The class of all continuous f on $\mathcal{X}$ which vanish at infinity is denoted as $C_0(\mathcal{X})$. For open $\mathcal{X} \subset \mathbb{R}^d$, $C^1(\mathcal{X})$ denotes the space of continuously differentiable functions on $\mathcal{X}$. For $f \in C_b(\mathcal{X})$, $\|f\|_\infty := \sup_{x \in \mathcal{X}} |f(x)|$ denotes the supremum norm of f. $M_b(\mathcal{X})$ denotes the set of all finite Borel measures on $\mathcal{X}$. For $\mu \in M_b(\mathcal{X})$, $L^r(\mathcal{X}, \mu)$ denotes the Banach space of r-power ($r \geq 1$) μ-integrable functions. For $\mathcal{X} \subset \mathbb{R}^d$, we will use $L^r(\mathcal{X})$ for $L^r(\mathcal{X}, \mu)$ if μ is a Lebesgue measure on $\mathcal{X}$. For $f \in L^p(\mathcal{X}, \mu)$, $\|f\|_{L^r(\mathcal{X},\mu)} := \left(\int_{\mathcal{X}} |f|^r \, d\mu\right)^{1/r}$ denotes the L^r-norm of f for $1 \leq r < \infty$ and we denote it as $\|\cdot\|_{L^r(\mathcal{X})}$ if $\mathcal{X} \subset \mathbb{R}^d$ and μ is the Lebesgue measure. The convolution $f * g$ of two measurable functions f and g on $\mathbb{R}^d$ is defined as

$$(f * g)(x) := \int_{\mathbb{R}^d} f(y) g(x - y) \, dy,$$

provided the integral exists for all $x \in \mathbb{R}^d$. The Fourier transform of $f \in L^1(\mathbb{R}^d)$ is defined as

$$f^\wedge(y) = (2\pi)^{-d/2} \int_{\mathbb{R}^d} f(x) \, e^{-i\langle y, x \rangle} \, dx$$

where i denotes the imaginary unit $\sqrt{-1}$.

In the following, for the sake of completeness and simplicity, we present definitions restricted to Hilbert spaces. Let H_1 and H_2 be abstract Hilbert spaces. A map $S : H_1 \to H_2$ is called a *linear operator* if it satisfies $S(\alpha x) = \alpha S x$ and $S(x + x') = Sx + Sx'$ for all $\alpha \in \mathbb{R}$ and $x, x' \in H_1$, where $Sx := S(x)$. A linear operator S is said to be *bounded*, i.e., the image SB_{H_1} of B_{H_1} under S is bounded if and only if there exists a constant $c \in [0, \infty)$ such that for all $x \in H_1$ we have $\|Sx\|_{H_2} \leq c \|x\|_{H_1}$, where $B_{H_1} := \{x \in H_1 : \|x\|_{H_1} \leq 1\}$. In this case, the *operator norm* of S is defined as $\|S\| := \sup\{\|Sx\|_{H_2} : x \in B_{H_1}\}$. Define $\mathcal{L}(H_1, H_2)$ be the space of bounded linear operators from H_1 to H_2. $S \in \mathcal{L}(H_1, H_2)$ is said to be *compact* if $\overline{SB_{H_1}}$ is a compact subset in H_2. The *adjoint operator* $S^* : H_2 \to H_1$ of $S \in \mathcal{L}(H_1, H_2)$ is defined by $\langle x, S^* y \rangle_{H_1} = \langle Sx, y \rangle_{H_2}$, $x \in H_1, y \in H_2$. $S \in \mathcal{L}(H) := \mathcal{L}(H, H)$ is called *self-adjoint* if $S^* = S$ and is called *positive* if $\langle Sx, x \rangle_H \geq 0$ for all $x \in H$. $\alpha \in \mathbb{R}$ is called an *eigenvalue* of $S \in \mathcal{L}(H)$ if there exists an $x \neq 0$ such that $Sx = \alpha x$ and such an x is called the *eigenvector* of S and α. For compact, positive, self-adjoint $S \in \mathcal{L}(H)$, $S^r : H \to H$, $r \geq 0$ is called a *fractional power* of S and $S^{1/2}$ is the *square root* of S, which we write as $\sqrt{S} := S^{1/2}$. An operator $S \in \mathcal{L}(H_1, H_2)$ is *Hilbert-Schmidt* if $\|S\|_{HS} := (\sum_{j \in J} \|Se_j\|_{H_2}^2)^{1/2} < \infty$ where $(e_j)_{j \in J}$ is an arbitrary orthonormal basis of separable Hilbert space H_1. $S \in \mathcal{L}(H_1, H_2)$ is said to be of *trace class* if $\sum_{j \in J} \langle (S^* S)^{1/2} e_j, e_j \rangle_{H_1} < \infty$. For $x \in H_1$ and $y \in H_2$, $x \otimes y$ is an element of the tensor product space $H_1 \otimes H_2$ which can also be seen as an operator from H_2 to H_1 as $(x \otimes y) z = x \langle y, z \rangle_{H_2}$ for any $z \in H_2$. $\mathcal{R}(S)$ denotes the *range space (or image)* of S.

A real-valued symmetric function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a positive definite (pd) kernel if, for all $n \in \mathbb{N}$, $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ and all $x_1, \ldots, x_n \in \mathcal{X}$, we have $\sum_{i,j=1}^n \alpha_i \alpha_j k(x_i, x_j) \geq 0$. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, $(x, y) \mapsto k(x, y)$ is a *reproducing kernel* of the Hilbert space $(\mathcal{H}_k, \langle \cdot, \cdot \rangle_{\mathcal{H}_k})$ of functions if and only if (i) $\forall\, y \in \mathcal{X}$, $k(y, \cdot) \in \mathcal{H}_k$ and (ii) $\forall\, y \in \mathcal{X}$, $\forall\, f \in \mathcal{H}_k$, $\langle f, k(y, \cdot) \rangle_{\mathcal{H}_k} = f(y)$ hold. If such a k exists, then $\mathcal{H}_k$ is called a *reproducing kernel Hilbert space*. Since $\langle k(x, \cdot), k(y, \cdot) \rangle_{\mathcal{H}_k} = k(x, y)$, $\forall\, x, y \in \mathcal{X}$, it is easy to show that every reproducing kernel (r.k.) k is symmetric and positive definite. Some examples of kernels that appear throughout the paper are: *Gaussian kernel*, $k(x, y) = \exp(-\sigma \|x - y\|_2^2)$, $x, y \in \mathbb{R}^d$, $\sigma > 0$ that induces the following *Gaussian RKHS*,

$$\mathcal{H}_k = \mathcal{H}_\sigma := \left\{ f \in L^2(\mathbb{R}^d) \cap C(\mathbb{R}^d) \, : \, \int |f^\wedge(\omega)|^2 e^{\|\omega\|_2^2 / 4\sigma} \, d\omega < \infty \right\},$$

the *inverse multiquadric kernel*, $k(x, y) = (1 + \|\frac{x - y}{c}\|_2^2)^{-\beta}$, $x, y \in \mathbb{R}^d$, $\beta > 0$, $c \in (0, \infty)$ and the *Matérn kernel*, $k(x, y) = \frac{2^{1-\beta}}{\Gamma(\beta)} \|x - y\|_2^{\beta - d/2} \mathfrak{K}_{d/2 - \beta}(\|x - y\|_2)$, $x, y \in \mathbb{R}^d$, $\beta > d/2$ that induces the Sobolev space, H_2^β,

$$\mathcal{H}_k = H_2^\beta := \left\{ f \in L^2(\mathbb{R}^d) \cap C(\mathbb{R}^d) \, : \, \int (1 + \|\omega\|_2^2)^\beta |f^\wedge(\omega)|^2 \, d\omega < \infty \right\},$$

where Γ is the Gamma function, and $\mathfrak{K}_v$ is the modified Bessel function of the third kind of order v (v controls the smoothness of k).

For any real-valued function f defined on open $\mathcal{X} \subset \mathbb{R}^d$, f is said to be m-times continuously differentiable if for $\alpha \in \mathbb{N}_0^d$ with $|\alpha| := \sum_{i=1}^d \alpha_i \leq m$, $\partial^\alpha f(x) = \partial_1^{\alpha_1} \dots \partial_d^{\alpha_d} f(x) = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \dots \partial x_d^{\alpha_d}} f(x)$ exists. A kernel k is said to be m-times continuously differentiable if $\partial^{\alpha,\alpha} k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ exists and is continuous for all $\alpha \in \mathbb{N}_0^d$ with $|\alpha| \leq m$ where $\partial^{\alpha,\alpha} := \partial_1^{\alpha_1} \dots \partial_d^{\alpha_d} \partial_{1+d}^{\alpha_1} \dots \partial_{2d}^{\alpha_d}$. Corollary 4.36 in Steinwart and Christmann (2008) and Theorem 1 in Zhou (2008) state that if $\partial^{\alpha,\alpha} k$ exists and is continuous, then $\partial^\alpha k(x, \cdot) = \partial_1^{\alpha_1} \dots \partial_d^{\alpha_d} k(x, \cdot) = \frac{\partial^{|\alpha|}}{\partial x_1^{\alpha_1} \dots \partial x_d^{\alpha_d}} k((x_1, \dots, x_d), \cdot) \in \mathcal{H}_k$ with $x = (x_1, \dots, x_d)$ and for every $f \in \mathcal{H}_k$, we have $\partial^\alpha f(x) = \langle \partial^\alpha k(x, \cdot), f \rangle_{\mathcal{H}_k}$ and $\partial^{\alpha,\alpha} k(x, x') = \langle \partial^\alpha k(x, \cdot), \partial^\alpha k(x', \cdot) \rangle_{\mathcal{H}_k}$.

Given two probability densities, p and q on $\Omega \subset \mathbb{R}^d$, the Kullback-Leibler divergence (KL) and Hellinger distance (h) are defined as $KL(p\|q) = \int p(x) \log \frac{p(x)}{q(x)} dx$ and $h(p, q) = \|\sqrt{p} - \sqrt{q}\|_{L^2(\Omega)}$ respectively. We refer to $\|p - q\|_{L^1(\Omega)}$ as the total variation (TV) distance between p and q.

3. Approximation of Densities by $\mathcal{P}$

In this section, we first show that every finite dimensional exponential family, $\mathscr{P}_{\text{fin}}$ is generated by the family $\mathcal{P}$ induced by a finite dimensional RKHS, which naturally leads to the infinite dimensional generalization of $\mathscr{P}_{\text{fin}}$ when $\mathcal{H}$ is an infinite dimensional RKHS. Next, we investigate the approximation properties of $\mathcal{P}$ in Proposition 1 and Corollary 2 when $\mathcal{H}$ is an infinite dimensional RKHS.

Let us consider a r-parameter exponential family, $\mathscr{P}_{\text{fin}}$ with sufficient statistic $T(x) := (T_1(x), \dots, T_r(x))$ and construct a Hilbert space, $\mathcal{H} = \text{span}\{T_1(x), \dots, T_r(x)\}$. It is easy to verify that $\mathcal{P}$ induced by $\mathcal{H}$ is exactly the same as $\mathscr{P}_{\text{fin}}$ since any $f \in \mathcal{H}$ can be written as $f(x) = \sum_{i=1}^r \theta_i T_i(x)$ for some $(\theta_i)_{i=1}^r \subset \mathbb{R}$. In fact, by defining the inner product between $f = \sum_{i=1}^r \theta_i T_i$ and $g = \sum_{i=1}^r \gamma_i T_i$ as $\langle f, g \rangle_{\mathcal{H}} := \sum_{i=1}^r \theta_i \gamma_i$, it follows that $\mathcal{H}$ is an RKHS with the r.k. $k(x, y) = \langle T(x), T(y) \rangle_{\mathbb{R}^r}$ since $\langle f, k(x, \cdot) \rangle_{\mathcal{H}} = \sum_{i=1}^r \theta_i T_i(x) = f(x)$. Based on this equivalence between $\mathscr{P}_{\text{fin}}$ and $\mathcal{P}$ induced by a finite dimensional RKHS, it is therefore clear that $\mathcal{P}$ induced by a infinite dimensional RKHS is a strict generalization to $\mathscr{P}_{\text{fin}}$ with $k(\cdot, x)$ playing the role of a sufficient statistic.

Example 1 *The following are some popular examples of probability distributions that belong to $\mathscr{P}_{\text{fin}}$. Here we show the corresponding RKHSs $(\mathcal{H}, k)$ that generate these distributions. In some of these examples, we choose $q_0(x) = 1$ and ignore the fact that q_0 is a probability distribution as assumed in the definition of $\mathcal{P}$.*

Exponential: $\Omega = \mathbb{R}_{++} := \mathbb{R}_+ \backslash \{0\}$, $k(x, y) = xy$.

Normal: $\Omega = \mathbb{R}$, $k(x, y) = xy + x^2 y^2$.

Beta: $\Omega = (0, 1)$, $k(x, y) = \log x \log y + \log(1 - x) \log(1 - y)$.

Gamma: $\Omega = \mathbb{R}_{++}$, $k(x, y) = \log x \log y + xy$.

Inverse Gaussian: $\Omega = \mathbb{R}_{++}$, $k(x, y) = xy + \frac{1}{xy}$.

Poisson: $\Omega = \mathbb{N} \cup \{0\}$, $k(x, y) = xy$, $q_0(x) = (x!\, e)^{-1}$.

Binomial: $\Omega = \{0, \dots, m\}$, $k(x, y) = xy$, $q_0(x) = 2^{-m} \binom{m}{c}$.

While Example 1 shows that all popular probability distributions are contained in $\mathcal{P}$ for an appropriate choice of finite-dimensional $\mathcal{H}$, it is of interest to understand the richness of $\mathcal{P}$ (i.e., what class of distributions can be approximated arbitrarily well by $\mathcal{P}$?) when $\mathcal{H}$ is an infinite dimensional RKHS. This is addressed by the following result, which is proved in Section 8.1.

Proposition 1 *Define*

$$\mathcal{P}_0 := \left\{ \pi_f(x) = e^{f(x) - A(f)} q_0(x),\ x \in \Omega : f \in C_0(\Omega) \right\}$$

where $\Omega \subseteq \mathbb{R}^d$ is locally compact Hausdorff. Suppose $k(x, \cdot) \in C_0(\Omega)$, $\forall\, x \in \Omega$ and

$$\int \int k(x, y)\, d\mu(x)\, d\mu(y) > 0,\ \forall\, \mu \in M_b(\Omega) \backslash \{0\}. \tag{5}$$

Then $\mathcal{P}$ is dense in $\mathcal{P}_0$ w.r.t. Kullback-Leibler divergence, total variation (L^1 norm) and Hellinger distances. In addition, if $q_0 \in L^1(\Omega) \cap L^r(\Omega)$ for some $1 < r \leq \infty$, then $\mathcal{P}$ is also dense in $\mathcal{P}_0$ w.r.t. L^r norm.

A sufficient condition for $\Omega \subseteq \mathbb{R}^d$ to be locally compact Hausdorff is that it is either open or closed. Condition (5) is equivalent to k being c_0-universal (Sriperumbudur et al., 2011, p. 2396). If $k(x, y) = \psi(x - y)$, $x, y \in \Omega = \mathbb{R}^d$ where $\psi \in C_b(\mathbb{R}^d) \cap L^1(\mathbb{R}^d)$, then (5) can be shown to be equivalent to $\text{supp}(\psi^\wedge) = \mathbb{R}^d$ (Sriperumbudur et al., 2011, Proposition 5). Examples of kernels that satisfy the conditions in Proposition 1 include the Gaussian, Matérn and inverse multiquadrics. In fact, any compactly supported non-zero $\psi \in C_b(\mathbb{R}^d)$ satisfies the assumptions in Proposition 1 as $\text{supp}(\psi^\wedge) = \mathbb{R}^d$ (Sriperumbudur et al., 2010, Corollary 10). Though $\mathcal{P}_0$ is still a parametric family of densities indexed by a Banach space (here $C_0(\Omega)$), the following corollary (proved in Section 8.2) to Proposition 1 shows that a broad class of continuous densities are contained in $\mathcal{P}_0$ and therefore can be approximated arbitrarily well in L^r norm ($1 \leq r \leq \infty$), Hellinger distance, and KL divergence by $\mathcal{P}$.

Corollary 2 *Let $q_0 \in C(\Omega)$ be a probability density such that $q_0(x) > 0$ for all $x \in \Omega$, where $\Omega \subseteq \mathbb{R}^d$ is locally compact Hausdorff. Suppose there exists a constant ℓ such that for any $\epsilon > 0$, $\exists\, R > 0$ that satisfies $|\frac{p(x)}{q_0(x)} - \ell| \leq \epsilon$ for any x with $\|x\|_2 > R$. Define*

$$\mathcal{P}_c := \left\{ p \in C(\Omega) : \int_\Omega p(x)\, dx = 1, p(x) \geq 0,\ \forall\, x \in \Omega \text{ and } \frac{p}{q_0} - \ell \in C_0(\Omega) \right\}.$$

Suppose $k(x, \cdot) \in C_0(\Omega)$, $\forall\, x \in \Omega$ and (5) holds. Then $\mathcal{P}$ is dense in $\mathcal{P}_c$ w.r.t. KL divergence, TV and Hellinger distances. Moreover, if $q_0 \in L^1(\Omega) \cap L^r(\Omega)$ for some $1 < r \leq \infty$, then $\mathcal{P}$ is also dense in $\mathcal{P}_c$ w.r.t. L^r norm.

By choosing Ω to be compact and q_0 to be a uniform distribution on Ω, Corollary 2 reduces to an easily interpretable result that any continuous density p_0 on Ω can be approximated arbitrarily well by densities in $\mathcal{P}$ in KL, Hellinger and L^r ($1 \leq r \leq \infty$) distances.

Similar to the results so far, an approximation result for $\mathcal{P}$ can also be obtained w.r.t. Fisher divergence (see Proposition 13). Since this result is heavily based on the notions and results developed in Section 5, we defer its presentation until that section. Briefly, this result states that if $\mathcal{H}$ is sufficiently rich (i.e., dense in an appropriate class of functions), then any $p \in C^1(\Omega)$ with $J(p\|q_0) < \infty$ can be approximated arbitrarily well by elements in $\mathcal{P}$ w.r.t. Fisher divergence, where $q_0 \in C^1(\Omega)$.

4. Density Estimation in $\mathcal{P}$: Well-specified Case

In this section, we present our score matching estimator for an unknown density $p_0 := p_{f_0} \in \mathcal{P}$ (well-specified case) from i.i.d. random samples $(X_a)_{a=1}^n$ drawn from it. This involves choosing the minimizer of the (empirical) Fisher divergence between p_0 and $p_f \in \mathcal{P}$ as the estimator, $\hat{f}$ which we show in Theorem 5 to be obtained by solving a simple finite-dimensional linear system. In contrast, we would like to remind the reader that the MLE is infeasible in practice due to the difficulty in handling $A(f)$. The consistency and convergence rates of $\hat{f} \in \mathcal{F}$ and the plug-in estimator $p_{\hat{f}}$ are provided in Section 4.1 (see Theorems 6 and 7). Before we proceed, we list the assumptions on p_0, q_0 and $\mathcal{H}$ that we need in our analysis.

(A) Ω is a non-empty open subset of $\mathbb{R}^d$ with a piecewise smooth boundary $\partial\Omega := \overline{\Omega}\backslash\Omega$, where $\overline{\Omega}$ denotes the closure of Ω.

(B) p_0 is continuously extendible to $\overline{\Omega}$. k is twice continuously differentiable on $\Omega \times \Omega$ with continuous extension of $\partial^{\alpha,\alpha}k$ to $\overline{\Omega} \times \overline{\Omega}$ for $|\alpha| \leq 2$.

(C) $\partial_i\partial_{i+d}k(x,x)p_0(x) = 0$ for $x \in \partial\Omega$ and $\sqrt{\partial_i\partial_{i+d}k(x,x)}p_0(x) = o(\|x\|_2^{1-d})$ as $x \in \Omega$, $\|x\|_2 \to \infty$ for all $i \in [d]$.

(D) *(ε-Integrability)* For some $\varepsilon \geq 1$ and $\forall\, i \in [d]$, $\partial_i\partial_{i+d}k(x,x)$, $\sqrt{\partial_i^2\partial_{i+d}^2k(x,x)}$ and $\sqrt{\partial_i\partial_{i+d}k(x,x)}\partial_i \log q_0(x) \in L^\varepsilon(\Omega, p_0)$, where $q_0 \in C^1(\Omega)$.

Remark 3 (i) *Ω being a subset of $\mathbb{R}^d$ along with k being continuous ensures that $\mathcal{H}$ is separable (Steinwart and Christmann, 2008, Lemma 4.33). The twice differentiability of k ensures that every $f \in \mathcal{H}$ is twice continuously differentiable (Steinwart and Christmann, 2008, Corollary 4.36).* **(C)** *ensures that J in (3) is equivalent to the one in (4) through integration by parts on Ω (see Corollary 7.6.2 in Duistermaat and Kolk, 2004 for integration by parts on bounded subsets of $\mathbb{R}^d$ which can be extended to unbounded Ω through a truncation and limiting argument) for densities in $\mathcal{P}$. In particular,* **(C)** *ensures that $\int_\Omega \partial_i f(x)\partial_i p_0(x)\,dx = -\int_\Omega \partial_i^2 f(x)p_0(x)\,dx$ for all $f \in \mathcal{H}$ and $i \in [d]$, which will be critical to prove the representation in Theorem 4(ii), upon which rest of the results depend. The decay condition in* **(C)** *can be weakened to $\sqrt{\partial_i\partial_{i+d}k(x,x)}p_0(x) = o(\|x\|_2^{1-\overline{d}})$ as $x \in \Omega$, $\|x\|_2 \to \infty$ for all $i \in [d]$ if Ω is a (possibly unbounded) box where $\overline{d} = \#\{i \in [d] | (a_i, b_i)$ is unbounded$\}$.*

(ii) *When $\varepsilon = 1$, the first condition in* **(D)** *ensures that $J(p_0\|p_f) < \infty$ for any $p_f \in \mathcal{P}$. The other two conditions ensure the validity of the alternate representation for $J(p_0\|p_f)$ in (4) which will be useful in constructing estimators of p_0 (see Theorem 4). Examples of kernels that satisfy* **(D)** *are the Gaussian, Matérn (with $\beta > \max\{2, d/2\}$), and inverse multiquadric kernels, for which it is easy to show that there exists q_0 that satisfies* **(D)**.

(iii) (Identifiability) *The above list of assumptions do not include the identifiability condition that ensures $p_{f_1} = p_{f_2}$ if and only if $f_1 = f_2$. It is clear that if constant functions are included in $\mathcal{H}$, i.e., $1 \in \mathcal{H}$, then $p_f = p_{f+c}$ for any $c \in \mathbb{R}$. On the other hand, it can be shown that if $1 \notin \mathcal{H}$ and $\operatorname{supp}(q_0) = \Omega$, then $p_{f_1} = p_{f_2} \Leftrightarrow f_1 = f_2$. A sufficient condition for $1 \notin \mathcal{H}$ is $k \in C_0(\Omega \times \Omega)$. We do not explicitly impose the identifiability condition as a part of our blanket assumptions because the assumptions under which consistency and rates are obtained in Theorem 7 automatically ensure identifiability.*

Under these assumptions, the following result—proved in Section 8.3—shows that the problem of estimating p_0 through the minimization of Fisher divergence reduces to the problem of estimating f_0 through a weighted least squares minimization in $\mathcal{H}$ (see parts (i) and (ii)). This motivates the minimization of the regularized empirical weighted least squares (see part (iv)) to obtain an estimator $f_{\lambda,n}$ of f_0, which is then used to construct the plug-in estimate $p_{f_{\lambda,n}}$ of p_0.

Theorem 4 *Suppose* **(A)**–**(D)** *hold with $\varepsilon = 1$. Then $J(p_0\|p_f) < \infty$ for all $f \in \mathcal{F}$. In addition, the following hold.*

(i) For all $f \in \mathcal{F}$,

$$J(f) := J(p_0\|p_f) = \frac{1}{2}\langle f - f_0, C(f - f_0)\rangle_{\mathcal{H}}, \tag{6}$$

where $C : \mathcal{H} \to \mathcal{H}$, $C := \int_\Omega p_0(x)\sum_{i=1}^d \partial_i k(x,\cdot) \otimes \partial_i k(x,\cdot)\,dx$ is a trace-class positive operator with

$$Cf = \int_\Omega p_0(x)\sum_{i=1}^d \partial_i k(x,\cdot)\partial_i f(x)\,dx.$$

(ii) Alternatively,

$$J(f) = \frac{1}{2}\langle f, Cf\rangle_{\mathcal{H}} + \langle f, \xi\rangle_{\mathcal{H}} + J(p_0\|q_0)$$

where

$$\xi := \int_\Omega p_0(x)\sum_{i=1}^d \left(\partial_i k(x,\cdot)\partial_i \log q_0(x) + \partial_i^2 k(x,\cdot)\right)\,dx \in \mathcal{H}$$

and f_0 satisfies $Cf_0 = -\xi$.

(iii) For any $\lambda > 0$, a unique minimizer f_λ of $J_\lambda(f) := J(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2$ over $\mathcal{H}$ exists and is given by

$$f_\lambda = -(C + \lambda I)^{-1}\xi = (C + \lambda I)^{-1}Cf_0.$$

(iv) ***(Estimator of f_0)*** *Given samples $(X_a)_{a=1}^n$ drawn i.i.d. from p_0, for any $\lambda > 0$, the unique minimizer $f_{\lambda,n}$ of $\hat{J}_\lambda(f) := \hat{J}(f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2$ over $\mathcal{H}$ exists and is given by*

$$f_{\lambda,n} = -(\hat{C} + \lambda I)^{-1}\hat{\xi},$$

where $\hat{J}(f) := \frac{1}{2}\langle f, \hat{C}f\rangle_{\mathcal{H}} + \langle f, \hat{\xi}\rangle_{\mathcal{H}} + J(p_0\|q_0)$, $\hat{C} := \frac{1}{n}\sum_{a=1}^{n}\sum_{i=1}^{d} \partial_i k(X_a,\cdot)\otimes\partial_i k(X_a,\cdot)$ *and*

$$\hat{\xi} := \frac{1}{n}\sum_{a=1}^{n}\sum_{i=1}^{d}\left(\partial_i k(X_a,\cdot)\partial_i \log q_0(X_a) + \partial_i^2 k(X_a,\cdot)\right).$$

An advantage of the alternate formulation of $J(f)$ in Theorem 4(ii) over (6) is that it provides a simple way to obtain an empirical estimate of $J(f)$—by replacing C and ξ by their empirical estimators, $\hat{C}$ and $\hat{\xi}$ respectively—from finite samples drawn i.i.d. from p_0, which is then used to obtain an estimator of f_0. Note that the empirical estimate of $J(f)$, i.e., $\hat{J}(f)$ depends only on $\hat{C}$ and $\hat{\xi}$ which in turn depend on the known quantities, k and q_0, and therefore $f_{\lambda,n}$ in Theorem 4(iv) should in principle be computable. In practice, however, it is not easy to compute the expression for $f_{\lambda,n} = -(\hat{C}+\lambda I)^{-1}\hat{\xi}$ as it involves solving an infinite dimensional linear system. In Theorem 5 (proved in Section 8.4), we provide an alternative expression for $f_{\lambda,n}$ as a solution of a simple finite-dimensional linear system (see (7) and (8)), using the general representer theorem (see Theorem A.2). It is interesting to note that while the solution to $J(f)$ in Theorem 4(ii) is obtained by solving a non-linear system, $Cf_0 = -\xi$ (the system is non-linear as C depends on p_0 which in turn depends on f_0), its estimator $f_{\lambda,n}$ proposed in Theorem 4, is obtained by solving a simple linear system. In addition, we would like to highlight the fact that the proposed estimator, $f_{\lambda,n}$ is precisely the Tikhonov regularized solution (which is well-studied in the theory of linear inverse problems) to the ill-posed linear system $\hat{C}f = -\hat{\xi}$. We further discuss the choice of regularizer in Section 4.3 using ideas from the inverse problem literature.

An important remark we would like to make about Theorem 4 is that though $J(f)$ in (6) is valid only for $f \in \mathcal{F}$, as it is obtained from $J(p_0\|p_f)$ where $p_0, p_f \in \mathcal{P}$, the expression $\langle f-f_0, C(f-f_0)\rangle_{\mathcal{H}}$ is valid for any $f \in \mathcal{H}$, as it is finite under the assumption that **(D)** holds with $\varepsilon = 1$. Therefore, in Theorem 4(iii, iv), f_λ and $f_{\lambda,n}$ are obtained by minimizing J_λ and $\hat{J}_\lambda$ over $\mathcal{H}$ instead of over $\mathcal{F}$, as the latter does not yield a nice expression (unlike f_λ and $f_{\lambda,n}$, respectively). However, there is no guarantee that $f_{\lambda,n} \in \mathcal{F}$, and so the density estimator $p_{f_{\lambda,n}}$ may not be valid. While this is not an issue when studying the convergence of $\|f_{\lambda,n} - f_0\|_{\mathcal{H}}$ (see Theorem 6), the convergence of $p_{f_{\lambda,n}}$ to p_0 (in various distances) needs to be handled slightly differently depending on whether the kernel is bounded or not (see Theorems 7 and B.2). Note that when the kernel is bounded, we obtain $\mathcal{F} = \mathcal{H}$, which implies $p_{f_{\lambda,n}}$ is valid.

Theorem 5 (Computation of $f_{\lambda,n}$) *Let* $f_{\lambda,n} = \arg\inf_{f\in\mathcal{H}} \hat{J}_\lambda(f)$, *where* $\hat{J}_\lambda(f)$ *is defined in Theorem 4(iv) and* $\lambda > 0$. *Then*

$$f_{\lambda,n} = -\frac{\hat{\xi}}{\lambda} + \sum_{a=1}^{n}\sum_{i=1}^{d}\beta_{(a-1)d+i}\partial_i k(X_a,\cdot), \tag{7}$$

where $\hat{\xi}$ *is defined in Theorem 4(iv) and* $\boldsymbol{\beta} = (\beta_{(a-1)d+i})_{a,i}$ *is obtained by solving*

$$(\boldsymbol{G} + n\lambda I)\,\boldsymbol{\beta} = \frac{1}{\lambda}\boldsymbol{h} \tag{8}$$

with $(\boldsymbol{G})_{(a-1)d+i,(b-1)d+j} = \partial_i\partial_{j+d}k(X_a,X_b)$ *and*

$$(\boldsymbol{h})_{(a-1)d+i} = \langle\hat{\xi}, \partial_i k(X_a,\cdot)\rangle_{\mathcal{H}} = \frac{1}{n}\sum_{b=1}^{n}\sum_{j=1}^{d}\partial_i\partial_{j+d}^2 k(X_a,X_b) + \partial_i\partial_{j+d}k(X_a,X_b)\partial_j \log q_0(X_b).$$

We would like to highlight that though $f_{\lambda,n}$ requires solving a simple linear system in (8), it can still be computationally intensive when d and n are large as $\boldsymbol{G}$ is a $nd\times nd$ matrix. This is still a better scenario than that of MLE, however, since computationally efficient methods exist to solve large linear systems such as (8), whereas MLE can be intractable due to the difficulty in handling the log-partition function (though it can be approximated). On the other hand, MLE is statistically well-understood, with consistency and convergence rates established in general for the problem of density estimation (van de Geer, 2000) and in particular for the problem at hand (Fukumizu, 2009). In order to ensure that $f_{\lambda,n}$ and $p_{f_{\lambda,n}}$ are statistically useful, in the following section, we investigate their consistency and convergence rates under some smoothness conditions on f_0.

4.1 Consistency and Rate of Convergence

In this section, we prove the consistency of $f_{\lambda,n}$ (see Theorem 6(i)) and $p_{f_{\lambda,n}}$ (see Theorems 7 and B.2). Under the smoothness assumption that $f_0 \in \mathcal{R}(C^\beta)$ for some $\beta > 0$, we present convergence rates for $f_{\lambda,n}$ and $p_{f_{\lambda,n}}$ in Theorems 6(ii), 7 and B.2. In reference to the following results, for simplicity we suppress the dependence of λ on n by defining $\lambda := \lambda_n$ where $(\lambda_n)_{n\in\mathbb{N}} \subset (0,\infty)$.

Theorem 6 (Consistency and convergence rates for $f_{\lambda,n}$) *Suppose* **(A)**–**(D)** *with* $\varepsilon = 2$ *hold.*

(i) If $f_0 \in \overline{\mathcal{R}(C)}$, *then* $\|f_{\lambda,n} - f_0\|_{\mathcal{H}} \xrightarrow{p_0} 0$ *as* $\lambda\to 0$, $\lambda\sqrt{n}\to\infty$ *and* $n\to\infty$.

(ii) If $f_0 \in \mathcal{R}(C^\beta)$ *for some* $\beta > 0$, *then for* $\lambda = n^{-\max\left\{\frac{1}{4},\frac{1}{2(\beta+1)}\right\}}$,

$$\|f_{\lambda,n} - f_0\|_{\mathcal{H}} = O_{p_0}\left(n^{-\min\left\{\frac{1}{4},\frac{\beta}{2(\beta+1)}\right\}}\right) \text{ as } n\to\infty.$$

(iii) If $\|C^{-1}\| < \infty$, *then for* $\lambda = n^{-\frac{1}{2}}$, $\|f_{\lambda,n} - f_0\|_{\mathcal{H}} = O_{p_0}(n^{-1/2})$ *as* $n\to\infty$.

Remark *(i)* While Theorem 6 (proved in Section 8.5) provides an asymptotic behavior for $\|f_{\lambda,n} - f_0\|_{\mathcal{H}}$ under conditions that depend on p_0 (and are therefore not easy to check in practice), a non-asymptotic bound on $\|f_{\lambda,n} - f_0\|_{\mathcal{H}}$ that holds for all $n \geq 1$ can be obtained under stronger assumptions through an application of Bernstein's inequality in separable Hilbert spaces. For the sake of simplicity, we provided asymptotic results which are obtained through an application of Chebyshev's inequality.

(ii) The proof of Theorem 6(i) involves decomposing $\|f_{\lambda,n} - f_0\|_{\mathcal{H}}$ into an estimation error part, $\mathcal{E}(\lambda,n) := \|f_{\lambda,n} - f_\lambda\|_{\mathcal{H}}$, and an approximation error part, $\mathcal{A}_0(\lambda) := \|f_\lambda - f_0\|_{\mathcal{H}}$, where $f_\lambda = (C+\lambda I)^{-1}Cf_0$. While $\mathcal{E}(\lambda,n)\to 0$ as $\lambda\to 0$, $\lambda\sqrt{n}\to\infty$ and $n\to\infty$ without any assumptions on f_0 (see the proof in Section 8.5 for details), it is not reasonable to expect

$\mathcal{A}_0(\lambda) \to 0$ as $\lambda \to 0$ without assuming $f_0 \in \overline{\mathcal{R}(C)}$. This is because, if f_0 lies in the null space of C, then f_λ is zero irrespective of λ and therefore cannot approximate f_0.

(iii) The condition $f_0 \in \overline{\mathcal{R}(C)}$ is difficult to check in practice as it depends on p_0 (which in turn depends on f_0). However, since the null space of C is just constant functions if the kernel is bounded and $supp(q_0) = \Omega$ (see Lemma 14 in Section 8.6 for details), assuming $1 \notin \mathcal{H}$ yields that $\overline{\mathcal{R}(C)} = \mathcal{H}$ and therefore consistency can be attained under conditions that are easy to impose in practice. As mentioned in Remark 3(iii), the condition $1 \notin \mathcal{H}$ ensures identifiability and a sufficient condition for it to hold is $k \in C_0(\Omega \times \Omega)$, which is satisfied by Gaussian, Matérn and inverse multiquadric kernels.

(iv) It is well known that convergence rates are possible only if the quantity of interest (here f_0) satisfies some additional conditions. In function estimation, this additional condition is classically imposed by assuming f_0 to be sufficiently smooth, e.g., f_0 lies in a Sobolev space of certain smoothness. By contrast, the smoothness condition in Theorem 6(ii) is imposed in an indirect manner by assuming $f_0 \in \mathcal{R}(C^\beta)$ for some $\beta > 0$—so that the results hold for abstract RKHSs and not just Sobolev spaces—which then provides a rate, with the best rate being $n^{-1/4}$ that is attained when $\beta \geq 1$. While such a condition has already been used in various works (Caponnetto and Vito, 2007; Smale and Zhou, 2007; Fukumizu et al., 2013) in the context of non-parametric least squares regression, we explore it in more detail in Proposition 8, and Examples 2 and 3. Note that this condition is common in the inverse problem theory (see Engl, Hanke, and Neubauer, 1996), and it naturally arises here through the connection of $f_{\lambda,n}$ being a Tikhonov regularized solution to the ill-posed linear system $\hat{C}f = -\hat{\xi}$. An interesting observation about the rate is that it does not improve with increasing β (for $\beta > 1$), in contrast to the classical results in function estimation (e.g., kernel density estimation and kernel regression) where the rate improves with increasing smoothness. This issue is discussed in detail in Section 4.3.

(v) Since $\|C^{-1}\| < \infty$ only if $\mathcal{H}$ is finite-dimensional, we recover the parametric rate of $n^{-1/2}$ in a finite-dimensional situation with an automatic choice for λ as $n^{-1/2}$.While Theorem 6 provides statistical guarantees for parameter convergence, the question of primary interest is the convergence of $p_{f_{\lambda,n}}$ to p_0. This is guaranteed by the following result, which is proved in Section 8.6.

Theorem 7 (Consistency and rates for $p_{f_{\lambda,n}}$) *Suppose* **(A)**–**(D)** *with* $\varepsilon = 2$ *hold and* $\|k\|_\infty := \sup_{x\in\Omega} k(x,x) < \infty$. *Assume* $\operatorname{supp}(q_0) = \Omega$. *Then the following hold:*

(i) For any $1 < r \leq \infty$ *with* $q_0 \in L^1(\Omega) \cap L^r(\Omega)$,

$$\|p_{f_{\lambda,n}} - p_0\|_{L^r(\Omega)} \to 0,\ h(p_{f_{\lambda,n}}, p_0) \to 0,\ KL(p_0 \| p_{f_{\lambda,n}}) \to 0 \ \text{ as } \lambda\sqrt{n} \to \infty,\ \lambda \to 0 \text{ and } n \to \infty.$$

In addition, if $f_0 \in \mathcal{R}(C^\beta)$ *for some* $\beta > 0$, *then for* $\lambda = n^{-\max\left\{\frac{1}{4}, \frac{1}{2(\beta+1)}\right\}}$,

$$\|p_{f_{\lambda,n}} - p_0\|_{L^r(\Omega)} = O_{p_0}(\theta_n),\ h(p_0, p_{f_{\lambda,n}}) = O_{p_0}(\theta_n),\ KL(p_0 \| p_{f_{\lambda,n}}) = O_{p_0}(\theta_n^2)$$

as $n \to \infty$ *where* $\theta_n := n^{-\min\left\{\frac{1}{4}, \frac{\beta}{2(\beta+1)}\right\}}$.

(ii) $J(p_0 \| p_{f_{\lambda,n}}) \to 0$ *as* $\lambda n \to \infty$, $\lambda \to 0$ *and* $n \to \infty$. *In addition, if* $f_0 \in \mathcal{R}(C^\beta)$ *for some* $\beta \geq 0$, *then for* $\lambda = n^{-\max\left\{\frac{1}{3}, \frac{1}{2(\beta+1)}\right\}}$,

$$J(p_0 \| p_{f_{\lambda,n}}) = O_{p_0}\left(n^{-\min\left\{\frac{2}{3}, \frac{2\beta+1}{2(\beta+1)}\right\}}\right) \ \text{ as } n \to \infty.$$

(iii) If $\|C^{-1}\| < \infty$, *then* $\theta_n = n^{-\frac{1}{2}}$ *and* $J(p_0 \| p_{f_{\lambda,n}}) = O_{p_0}(n^{-1})$ *with* $\lambda = n^{-\frac{1}{2}}$.

Remark *(i)* Comparing the results of Theorem 6(i) and Theorem 7(i) (for L^r, Hellinger and KL divergence), we would like to highlight that while the conditions on λ and n match in both the cases, the latter does not require $f_0 \in \overline{\mathcal{R}(C)}$ to ensure consistency. While $f_0 \in \overline{\mathcal{R}(C)}$ can be imposed in Theorem 7 to attain consistency, we replaced this condition with $supp(q_0) = \Omega$—a simple and easy condition to work with—which along with the boundedness of the kernel ensures that for any $f_0 \in \mathcal{H}$, there exists $\tilde{f}_0 \in \overline{\mathcal{R}(C)}$ such that $p_{\tilde{f}_0} = p_0$ (see Lemma 14).

(ii) In contrast to the results in L^r, Hellinger and KL divergence, consistency in J can be obtained with λ converging to zero at a rate faster than in these results. In addition, one can obtain rates in J with $\beta = 0$, i.e., no smoothness assumption on f_0, while no rates are possible in other distances (the latter might also be an artifact of the proof technique, as these results are obtained through an application of Theorem 6(ii) in Lemma A.1) which is due to the fact that the convergence in these other distances is based on the convergence of $\|f_{\lambda,n} - f_0\|_{\mathcal{H}}$, which in turn involves convergence of $\mathcal{A}_0(\lambda) := \|f_\lambda - f_0\|_{\mathcal{H}}$ to zero while the convergence in J is controlled by $\mathcal{A}_{\frac{1}{2}}(\lambda) := \|\sqrt{C}(f_\lambda - f_0)\|_{\mathcal{H}}$ which can be shown to behave as $O(\sqrt{\lambda})$ as $\lambda \to 0$, without requiring any assumptions on f_0 (see Proposition A.3). Indeed, as a further consequence, the rate of convergence in J is faster than in other distances.

(iii) An interesting aspect in Theorem 7 is that $p_{f_{\lambda,n}}$ is consistent in various distances such as L^r, Hellinger and KL, despite being obtained by minimizing a different loss function, i.e., J. However, we will see in Section 5 that such nice results are difficult to obtain in the misspecified case, where consistency and rates are provided only in J.

While Theorem 7 addresses the case of bounded kernels, the case of unbounded kernels requires a technical modification. The reason for this modification, as alluded to in the discussion following Theorem 4, is due to the fact that $f_{\lambda,n}$ may not be in $\mathcal{F}$ when k is unbounded, and therefore the corresponding density estimator, $p_{f_{\lambda,n}}$ may not be well-defined. In order to keep the main ideas intact, we discuss the unbounded case in detail in Section B.2 in Appendix B.

4.2 Range Space Assumption

While Theorems 6 and 7 are satisfactory from the point of view of consistency, we believe the presented rates are possibly not minimax optimal since these rates are valid for any RKHS that satisfies the conditions **(A)**–**(D)** and does not capture the smoothness of k (and therefore the corresponding $\mathcal{H}$). In other words, the rates presented in Theorems 6 and 7 should depend on the decay rate of the eigenvalues of C which in turn effectively captures the smoothness of $\mathcal{H}$. However, we are not able to obtain such a result—see the remark following the proof of Theorem 6 for a discussion. While these rates do not reflect

the intrinsic smoothness of $\mathcal{H}$, they are obtained under the smoothness assumption, i.e., *range space condition* that $f_0 \in \mathcal{R}(C^\beta)$ for some $\beta > 0$. This condition is quite different from the classical smoothness conditions that appear in non-parametric function estimation. While the range space assumption has been made in various earlier works (e.g., Caponnetto and Vito (2007); Smale and Zhou (2007); Fukumizu et al. (2013) in the context of non-parametric least square regression), in the following, we investigate the implicit smoothness assumptions that it makes on f_0 in our context. To this end, first it is easy to show (see the proof of Proposition B.3 in Section B.3) that

$$\mathcal{R}(C^\beta) = \left\{ \sum_{i\in I} c_i \phi_i \,:\, \sum_{i\in I} c_i^2 \alpha_i^{-2\beta} < \infty \right\}, \tag{9}$$

where $(\alpha_i)_{i\in I}$ are the positive eigenvalues of C, $(\phi_i)_{i\in I}$ are the corresponding eigenvectors that form an orthonormal basis for $\mathcal{R}(C)$, and I is an index set which is either finite (if $\mathcal{H}$ is finite-dimensional) or $I = \mathbb{N}$ with $\lim_{i\to\infty} \alpha_i = 0$ (if $\mathcal{H}$ is infinite dimensional). From (9) it is clear that larger the value of β, the faster is the decay of the Fourier coefficients $(c_i)_{i\in I}$, which in turn implies that the functions in $\mathcal{R}(C^\beta)$ are smoother. Using (9), an interpretation can be provided for $\mathcal{R}(C^\beta)$ ($\beta > 0$ and $\beta \notin \mathbb{N}$) as interpolation spaces (see Section A.5 for the definition of interpolation spaces) between $\mathcal{R}(C^{\lceil\beta\rceil})$ and $\mathcal{R}(C^{\lfloor\beta\rfloor})$ where $\mathcal{R}(C^0) := \mathcal{H}$ (see Proposition B.3 for details). While it is not completely straightforward to obtain a sufficient condition for $f_0 \in \mathcal{R}(C^\beta)$, $\beta \in \mathbb{N}$, the following result provides a necessary condition for $f_0 \in \mathcal{R}(C)$ (and therefore a necessary condition for $f_0 \in \mathcal{R}(C^\beta)$, $\forall\, \beta > 1$) for translation invariant kernels on $\Omega = \mathbb{R}^d$, whose proof is presented in Section 8.7.

Proposition 8 (Necessary condition) *Suppose $\psi, \phi \in C_b(\mathbb{R}^d) \cap L^1(\mathbb{R}^d)$ are positive definite functions on $\mathbb{R}^d$ with Fourier transforms $\psi^\wedge$ and $\phi^\wedge$ respectively. Let $\mathcal{H}$ and $\mathcal{G}$ be the RKHSs associated with $k(x,y) = \psi(x-y)$ and $l(x,y) = \phi(x-y), x, y \in \mathbb{R}^d$ respectively. For $1 \le r \le 2$, suppose the following hold:*

(i) $\int_{\mathbb{R}^d} \|\omega\|_2^2 \psi^\wedge(\omega)\, d\omega < \infty$; (ii) $\left\| \frac{\phi^\wedge}{\psi^\wedge} \right\|_\infty < \infty$; (iii) $\frac{\|\cdot\|_2^2 (\psi^\wedge)^2}{\phi^\wedge} \in L^{\frac{r}{2-r}}(\mathbb{R}^d)$; (iv) $q_0 \in L^r(\mathbb{R}^d)$.

Then $f_0 \in \mathcal{R}(C)$ implies $f_0 \in \mathcal{G} \subset \mathcal{H}$.

In the following, we apply the above result in two examples involving Gaussian and Matérn kernels to get insights into the range space assumption.

Example 2 (Gaussian kernel) *Let $\psi(x) = e^{-\sigma\|x\|^2}$ with $\mathcal{H}_\sigma$ as its corresponding RKHS (see Section 2 for its definition). By Proposition 8, it is easy to verify that $f_0 \in \mathcal{R}(C)$ implies $f_0 \in \mathcal{H}_\alpha \subset \mathcal{H}_\sigma$ for $\frac{\sigma}{2} < \alpha \le \sigma$. Since $\mathcal{H}_\beta \subset \mathcal{H}_\gamma$ for $\beta < \gamma$ (i.e., Gaussian RKHSs are nested), $f_0 \in \mathcal{R}(C)$ ensures that f_0 lies in $\mathcal{H}_{\frac{\sigma}{2}+\epsilon}$ for arbitrary small $\epsilon > 0$.*

Example 3 (Matérn kernel) *Let $\psi(x) = \frac{2^{1-s}}{\Gamma(s)} \|x\|_2^{s-\frac{d}{2}} \mathfrak{K}_{d/2-s}(\|x\|_2)$, $x \in \mathbb{R}^d$ with $H_2^s(\mathbb{R}^d)$ as its corresponding RKHS (see Section 2 for its definition) where $s > \frac{d}{2}$. By Proposition 8, we have that for $q_0 \in L^1(\mathbb{R}^d)$, if $f_0 \in \mathcal{R}(C)$, then $f_0 \in H_2^\alpha(\mathbb{R}^d) \subset H_2^s(\mathbb{R}^d)$ for $1 + \frac{d}{2} < s \le \alpha < 2s - 1 - \frac{d}{2}$. Since $H_2^\delta(\mathbb{R}^d) \subset H_2^\gamma(\mathbb{R}^d)$ for $\gamma < \delta$ (i.e., Sobolev spaces are nested), this means f_0 lies in $H_2^{2s-1-\frac{d}{2}-\epsilon}(\mathbb{R}^d)$ for arbitrarily small $\epsilon > 0$, i.e., f_0 has at least $2s - 1 - \lceil \frac{d}{2} \rceil$ weak-derivatives. By the minimax theory (Tsybakov, 2009, Chapter 2), it is well known that for any $\alpha > \delta \ge 0$,*

$$\inf_{\hat{f}_n} \sup_{f_0 \in H_2^\alpha(\mathbb{R}^d)} \|\hat{f}_n - f_0\|_{H_2^\delta(\mathbb{R}^d)} \asymp n^{-\frac{\alpha-\delta}{2(\alpha-\delta)+d}}, \tag{10}$$

where the infimum is taken over all possible estimators. Here $a_n \asymp b_n$ means that for any two sequences $a_n, b_n > 0$, a_n/b_n is bounded away from zero and infinity as $n \to \infty$. Suppose $f_0 \notin H_2^\alpha(\mathbb{R}^d)$ for $\alpha \ge 2s - 1 - \frac{d}{2}$, which means $f_0 \in H_2^{2s-1-\frac{d}{2}-\epsilon}(\mathbb{R}^d)$ for arbitrarily small $\epsilon > 0$. This implies that the rate of $n^{-1/4}$ obtained in Theorem 6 is minimax optimal if $\mathcal{H}$ is chosen to be $H_2^{1+d+\epsilon}(\mathbb{R}^d)$ (i.e., choose $\alpha = 2s - 1 - \frac{d}{2} - \epsilon$ and $\delta = s$ in (10) and solve for s by equating the exponent in the r.h.s. of (10) to $-\frac{1}{4}$). Similarly, it can be shown that if $q_0 \in L^2(\mathbb{R}^d)$, then the rate of $n^{-1/4}$ in Theorem 6 is minimax optimal if $\mathcal{H}$ is chosen to be $H_2^{1+\frac{d}{2}+\epsilon}(\mathbb{R}^d)$. This example also explains away the dimension independence of the rate provided by Theorem 6 by showing that the dimension effect is captured in the relative smoothness of f_0 w.r.t. $\mathcal{H}$.

While Example 3 provides some understanding about the minimax optimality of $f_{\lambda,n}$ under additional assumptions on f_0, the problem is not completely resolved. In the following section, however, we show that the rate in Theorem 6 is not optimal for $\beta > 1$, and that improved rates can be obtained by choosing the regularizer appropriately.

4.3 Choice of Regularizer

We understand from the characterization of $\mathcal{R}(C^\beta)$ in (9) that larger β values yield smoother functions in $\mathcal{H}$. However, the smoothness of $f_0 \in \mathcal{R}(C^\beta)$ for $\beta > 1$ is not captured in the rates in Theorem 6(ii), where the rate saturates at $\beta = 1$ providing the best possible rate of $n^{-1/4}$ (irrespective of the size of β). This is unsatisfactory on the part of the estimator, as it does not effectively capture the smoothness of f_0, i.e., the estimator is not adaptive to the smoothness of f_0. We remind the reader that the estimator $f_{\lambda,n}$ is obtained by minimizing the regularized empirical Fisher divergence (see Theorem 4(iv)) yielding $f_{\lambda,n} = -(\hat{C} + \lambda I)^{-1}\hat{\xi}$, which can be seen as a heuristic to solve the (non-linear) inverse problem $Cf_0 = -\xi$ (see Theorem 4(ii)) from finite samples, by replacing C and ξ with their empirical counterparts. This heuristic, which ensures that the finite sample inverse problem is well-posed, is popular in inverse problem literature under the name of Tikhonov regularization (Engl et al., 1996, Chapter 5). Note that Tikhonov regularization helps to make the ill-posed inverse problem a well-posed one by approximating α^{-1} by $(\alpha + \lambda)^{-1}$, $\lambda > 0$, where α^{-1} appears as the inverse of the eigenvalues of C while computing C^{-1}. In other words, if $\hat{C}$ is invertible, then an estimate of f_0 can be obtained as $\hat{f}_n = -\hat{C}^{-1}\hat{\xi}$, i.e., $\hat{f}_n = -\sum_{i\in I} \frac{\langle \hat{\xi}, \hat{\phi}_i \rangle_{\mathcal{H}}}{\hat{\alpha}_i} \hat{\phi}_i$, where $(\hat{\alpha}_i)_{i\in I}$ and $(\hat{\phi}_i)_{i\in I}$ are the eigenvalues and eigenvectors of $\hat{C}$ respectively. However, $\hat{C}$ being a rank n operator defined on $\mathcal{H}$ (which can be infinite dimensional) is not invertible and therefore the regularized estimator is constructed as $f_{\lambda,n} = -g_\lambda(\hat{C})\hat{\xi}$ where $g_\lambda(\hat{C})$ is defined through functional calculus (see Engl, Hanke, and

Neubauer, 1996, Section 2.3) as

$$g_\lambda(\hat{C}) = \sum_{i \in I} g_\lambda(\hat{\alpha}_i) \langle \cdot, \hat{\phi}_i \rangle_{\mathcal{H}} \hat{\phi}_i$$

with $g_\lambda : \mathbb{R}_+ \to \mathbb{R}$ and $g_\lambda(\alpha) := (\alpha + \lambda)^{-1}$. Since the Tikhonov regularization is well-known to saturate (as explained above)—see Engl et al. (1996, Sections 4.2 and 5.1) for details—, better approximations to α^{-1} have been used in the inverse problems literature to improve the rates by using g_λ other than $(\cdot + \lambda)^{-1}$ where $g_\lambda(\alpha) \to \alpha^{-1}$ as $\lambda \to 0$. In the statistical context, Rosasco et al. (2005) and Bauer et al. (2007) have used the ideas from Engl et al. (1996) in non-parametric regression for learning a square integrable function from finite samples through regularization in RKHS. In the following, we use these ideas to construct an alternate estimator for f_0 (and therefore for p_0) that appropriately captures the smoothness of f_0 by providing a better convergence rate when $\beta > 1$. To this end, we need the following assumption—quoted from Engl et al. (1996, Theorems 4.1–4.3 and Corollary 4.4) and Bauer et al. (2007, Definition 1)—that is standard in the theory of inverse problems.

(E) There exists finite positive constants A_g, B_g, C_g, η_0 and $(\gamma_\eta)_{\eta \in (0,\eta_0]}$ (all independent of $\lambda > 0$) such that $g_\lambda : [0, \chi] \to \mathbb{R}$ satisfies:

(a) $\sup_{\alpha \in \mathcal{D}} |\alpha g_\lambda(\alpha)| \leq A_g$, (b) $\sup_{\alpha \in \mathcal{D}} |g_\lambda(\alpha)| \leq \frac{B_g}{\lambda}$, (c) $\sup_{\alpha \in \mathcal{D}} |1 - \alpha g_\lambda(\alpha)| \leq C_g$ and (d) $\sup_{\alpha \in \mathcal{D}} |1 - \alpha g_\lambda(\alpha)| \alpha^\eta \leq \gamma_\eta \lambda^\eta$, $\forall\, \eta \in (0, \eta_0]$ where $\mathcal{D} := [0, \chi]$ and $\chi := d \sup_{x \in \Omega, i \in [d]} \partial_i \partial_{i+d} k(x, x) < \infty$.

The constant η_0 is called the *qualification* of g_λ which is what determines the point of saturation of g_λ. We show in Theorem 9 that if g_λ has a finite qualification, then the resultant estimator cannot fully exploit the smoothness of f_0 and therefore the rate of convergence will suffer for $\beta > \eta_0$. Given g_λ that satisfies **(E)**, we construct our estimator of f_0 as

$$f_{g,\lambda,n} = -g_\lambda(\hat{C})\hat{\xi}.$$

Note that the above estimator can be obtained by using the data dependent regularizer, $\frac{1}{2}\langle f, ((g_\lambda(\hat{C}))^{-1} - \hat{C}) f \rangle_{\mathcal{H}}$ in the minimization of $\hat{J}(f)$ defined in Theorem 4(iv), i.e.,

$$f_{g,\lambda,n} = \arg \inf_{f \in \mathcal{H}} \hat{J}(f) + \frac{1}{2}\langle f, ((g_\lambda(\hat{C}))^{-1} - \hat{C}) f \rangle_{\mathcal{H}}.$$

However, unlike $f_{\lambda,n}$ for which a simple form is available in Theorem 5 by solving a linear system, we are not able to obtain such a nice expression for $f_{g,\lambda,n}$. The following result (proved in Section 8.8) presents an analog of Theorems 6 and 7 for the new estimators, $f_{g,\lambda,n}$ and $p_{f_{g,\lambda,n}}$.

Theorem 9 (Consistency and convergence rates for $f_{g,\lambda,n}$ and $p_{f_{g,\lambda,n}}$) *Suppose* **(A)**–**(E)** *hold with* $\varepsilon = 2$.

(i) If $f_0 \in \mathcal{R}(C^\beta)$ *for some* $\beta > 0$, *then for any* $\lambda \geq n^{-1/2}$,

$$\|f_{g,\lambda,n} - f_0\|_{\mathcal{H}} = O_{p_0}(\theta_n),$$

where $\theta_n := n^{-\min\left\{\frac{\beta}{2(\beta+1)}, \frac{\eta_0}{2(\eta_0+1)}\right\}}$ *with* $\lambda = n^{-\max\left\{\frac{1}{2(\beta+1)}, \frac{1}{2(\eta_0+1)}\right\}}$. *In addition, if* $\|k\|_\infty < \infty$, *then for any* $1 < r \leq \infty$ *with* $q_0 \in L^1(\Omega) \cap L^r(\Omega)$,

$$\|p_{f_{g,\lambda,n}} - p_0\|_{L^r(\Omega)} = O_{p_0}(\theta_n),\ h(p_0, p_{f_{g,\lambda,n}}) = O_{p_0}(\theta_n) \ \text{ and } \ KL(p_0 \| p_{f_{g,\lambda,n}}) = O_{p_0}(\theta_n^2).$$

(ii) If $f_0 \in \mathcal{R}(C^\beta)$ *for some* $\beta \geq 0$, *then for any* $\lambda \geq n^{-1/2}$,

$$J(p_0 \| p_{f_{g,\lambda,n}}) = O_{p_0}\left(n^{-\frac{\min\{2\beta+1, 2\eta_0\}}{\min\{2\beta+2, 2\eta_0+1\}}}\right)$$

with $\lambda = n^{-\frac{1}{\min\{2\beta+2, 2\eta_0+1\}}}$.

(iii) If $\|C^{-1}\| < \infty$, *then for any* $\lambda \geq n^{-1/2}$,

$$\|f_{g,\lambda,n} - f_0\|_{\mathcal{H}} = O_{p_0}(\theta_n) \ \text{ and } \ J(p_0 \| p_{f_{g,\lambda,n}}) = O_{p_0}(\theta_n^2)$$

with $\theta_n = n^{-\frac{1}{2}}$ *and* $\lambda = n^{-\frac{1}{\min\{2, 2\eta_0\}}}$.

Theorem 9 shows that if g_λ has infinite qualification, then smoothness of f_0 is fully captured in the rates and as $\beta \to \infty$, we attain $O_{p_0}(n^{-1/2})$ rate for $\|f_{g,\lambda,n} - f_0\|_{\mathcal{H}}$ in contrast to $n^{-1/4}$ (similar improved rates are also obtained for $p_{f_{g,\lambda,n}}$ in various distances) in Theorem 6. In the following example, we present two choices of g_λ that improve on Tikhonov regularization. We refer the reader to Rosasco et al. (2005, Section 3.1) for more examples of g_λ.

Example 4 (Choices of g_λ) *(i) Tikhonov regularization involves* $g_\lambda(\alpha) = (\alpha + \lambda)^{-1}$ *for which it is easy to verify that* $\eta_0 = 1$ *and therefore the rates saturate at* $\beta = 1$, *leading to the results in Theorems 6 and 7.*

(ii) Showalter's method and spectral cut-off use

$$g_\lambda(\alpha) = \frac{1 - e^{-\alpha/\lambda}}{\alpha} \ \text{ and } \ g_\lambda(\alpha) = \begin{cases} \frac{1}{\alpha}, & \alpha \geq \lambda \\ 0, & \alpha < \lambda \end{cases}$$

respectively for which it is easy to verify that $\eta_0 = +\infty$ *(see Engl, Hanke, and Neubauer, 1996, Examples 4.7 & 4.8 for details) and therefore improved rates are obtained for* $\beta > 1$ *in Theorem 9 compared to that of Tikhonov regularization.*

5. Density Estimation in $\mathcal{P}$: Misspecified Case

In this section, we analyze the misspecified case where $p_0 \notin \mathcal{P}$, which is a more reasonable case than the well-specified one, as in practice it is not easy to check whether $p_0 \in \mathcal{P}$. To this end, we consider the same estimator $p_{f_{\lambda,n}}$ as considered in the well-specified case where $f_{\lambda,n}$ is obtained from Theorem 5. The following result shows that $J(p_0 \| p_{f_{\lambda,n}}) \to \inf_{p \in \mathcal{P}} J(p_0 \| p)$ as $\lambda \to 0$, $\lambda n \to \infty$ and $n \to \infty$ under the assumption that there exists $f^* \in \mathcal{F}$ such that $J(p_0 \| p_{f^*}) = \inf_{p \in \mathcal{P}} J(p_0 \| p)$. We present the result for bounded kernels although it can be easily extended to unbounded kernels as in Theorem B.2. Also, the presented result for Tikhonov regularization extends easily to $p_{f_{g,\lambda,n}}$ using the ideas in the proof of Theorem 9. Note that unlike in the well-specified case where convergence in other distances can be shown even though the estimator is constructed from J, it is difficult to show such a result in the misspecified case.

Theorem 10 *Let p_0, $q_0 \in C^1(\Omega)$ be probability densities such that $J(p_0\|q_0) < \infty$ where Ω satisfies* **(A)**. *Assume that* **(B)**, **(C)** *and* **(D)** *with $\varepsilon = 2$ hold. Suppose $\|k\|_\infty < \infty$, $\operatorname{supp}(q_0) = \Omega$ and there exists $f^* \in \mathcal{F}$ such that*

$$J(p_0\|p_{f^*}) = \inf_{p\in\mathcal{P}} J(p_0\|p).$$

Then for an estimator $p_{f_{\lambda,n}}$ constructed from random samples $(X_a)_{a=1}^n$ drawn i.i.d. from p_0, where $f_{\lambda,n}$ is defined in (7)—also see Theorem 4(iv)—with $\lambda > 0$, we have

$$J(p_0\|p_{f_{\lambda,n}}) \to \inf_{p\in\mathcal{P}} J(p_0\|p) \;\text{ as } \lambda \to 0,\ \lambda n \to \infty \text{ and } n \to \infty.$$

In addition, if $f^* \in \mathcal{R}(C^\beta)$ for some $\beta \geq 0$, then

$$\sqrt{J(p_0\|p_{f_{\lambda,n}})} \leq \sqrt{\inf_{p\in\mathcal{P}} J(p_0\|p)} + O_{p_0}\left(n^{-\min\left\{\frac{1}{3},\frac{2\beta+1}{4(\beta+1)}\right\}}\right)$$

with $\lambda = n^{-\max\left\{\frac{1}{3},\frac{1}{2(\beta+1)}\right\}}$. If $\|C^{-1}\| < \infty$, then for $\lambda = n^{-\frac{1}{2}}$,

$$\sqrt{J(p_0\|p_{f_{\lambda,n}})} \leq \sqrt{\inf_{p\in\mathcal{P}} J(p_0\|p)} + O_{p_0}(n^{-1/2}).$$

with $\lambda = n^{-\frac{1}{2}}$.

While the above result is useful and interesting, the assumption about the existence of f^* is quite restrictive. This is because if p_0 (which is not in $\mathcal{P}$) belongs to a family $\mathcal{Q}$ where $\mathcal{P}$ is dense in $\mathcal{Q}$ w.r.t. J, then there is no $f \in \mathcal{H}$ that attains the infimum, i.e., f^* does not exist and therefore the proof technique employed in Theorem 10 will fail. In the following, we present a result (Theorem 12) that does not require the existence of f^* but attains the same result as in Theorem 10, but requiring a more complicated proof. Before we present Theorem 12, we need to introduce some notation.

To this end, let us return to the objective function under consideration,

$$J(p_0\|p_f) = \frac{1}{2}\int_\Omega p_0(x)\left\|\nabla \log\frac{p_0}{p_f}\right\|_2^2 dx = \frac{1}{2}\int_\Omega p_0(x)\sum_{i=1}^d (\partial_i f_\star - \partial_i f)^2\, dx,$$

where $f_\star = \log\frac{p_0}{q_0}$ and $p_0 \notin \mathcal{P}$. Define

$$\mathcal{W}_2(\Omega, p_0) := \left\{f \in C^1(\Omega) \,:\, \partial^\alpha f \in L^2(\Omega, p_0),\, \forall\, |\alpha| = 1\right\}.$$

This is a reasonable class of functions to consider as under the condition $J(p_0\|q_0) < \infty$, it is clear that $f_\star \in \mathcal{W}_2(\Omega, p_0)$. Endowed with a semi-norm,

$$\|f\|_{\mathcal{W}_2}^2 := \sum_{|\alpha|=1} \|\partial^\alpha f\|_{L^2(\Omega,p_0)}^2,$$

$\mathcal{W}_2(\Omega, p_0)$ is a vector space of functions, from which a normed space can be constructed as follows. Let us define $f, f' \in \mathcal{W}_2(\Omega, p_0)$ to be equivalent, i.e., $f \sim f'$, if $\|f - f'\|_{\mathcal{W}_2} = 0$.

In other words, $f \sim f'$ if and only if f and f' differ by a constant p_0-almost everywhere. Now define the quotient space $\mathcal{W}_2^\sim(\Omega, p_0) := \{[f]_\sim : f \in \mathcal{W}_2(\Omega, p_0)\}$ where $[f]_\sim := \{f' \in \mathcal{W}_2(\Omega, p_0) : f \sim f'\}$ denotes the equivalence class of f. Defining $\|[f]_\sim\|_{\mathcal{W}_2^\sim} := \|f\|_{\mathcal{W}_2}$, it is easy to verify that $\|\cdot\|_{\mathcal{W}_2^\sim}$ defines a norm on $\mathcal{W}_2^\sim(p_0)$. In addition, endowing the following bilinear form on $\mathcal{W}_2^\sim(\Omega, p_0)$

$$\langle [f]_\sim, [g]_\sim\rangle_{\mathcal{W}_2^\sim} := \int_\Omega p_0(x)\sum_{|\alpha|=1}(\partial^\alpha f)(x)(\partial^\alpha g)(x)\, dx$$

makes it a pre-Hilbert space. Let $W_2(\Omega, p_0)$ be the Hilbert space obtained by completion of $\mathcal{W}_2^\sim(\Omega, p_0)$. As shown in Proposition 11 below, under some assumptions, a continuous mapping $I_k : \mathcal{H} \to W_2(\Omega, p_0)$, $f \mapsto [f]_\sim$ can be defined, which is injective modulo constant functions. Since addition of a constant does not contribute to p_f, the space $W_2(\Omega, p_0)$ can be regarded as a parameter space extended from $\mathcal{H}$. In addition to I_k, Proposition 11 (proved in Section 8.10) describes the adjoint of I_k and relevant self-adjoint operators, which will be useful in analyzing $p_{f_{\lambda,n}}$ in Theorem 12.

Proposition 11 *Let $\operatorname{supp}(q_0) = \Omega$ where $\Omega \subset \mathbb{R}^d$ is non-empty and open. Suppose k satisfies* **(B)** *and $\partial_i\partial_{i+d}k(x,x) \in L^1(\Omega, p_0)$ for all $i \in [d]$. Then $I_k : \mathcal{H} \to W_2(\Omega, p_0)$, $f \mapsto [f]_\sim$ defines a continuous mapping with the null space $\mathcal{H} \cap \mathbb{R}$. The adjoint of I_k is $S_k : W_2(\Omega, p_0) \to \mathcal{H}$ whose restriction to $\mathcal{W}_2^\sim(\Omega, p_0)$ is given by*

$$S_k[h]_\sim(y) = \int_\Omega \sum_{i=1}^d \partial_i k(x,y)\partial_i h(x)\, p_0(x)\, dx, \qquad [h]_\sim \in \mathcal{W}_2^\sim(\Omega, p_0),\ y \in \Omega.$$

In addition, I_k and S_k are Hilbert-Schmidt and therefore compact. Also, $E_k := S_kI_k$ and $T_k := I_kS_k$ are compact, positive and self-adjoint operators on $\mathcal{H}$ and $W_2(\Omega, p_0)$ respectively where

$$E_k g(y) = \int_\Omega \sum_{i=1}^d \partial_i k(x,y)\partial_i g(x) p_0(x)\, dx, \qquad g \in \mathcal{H},\ y \in \Omega$$

and the restriction of T_k to $\mathcal{W}_2^\sim(\Omega, p_0)$ is given by

$$T_k[h]_\sim = \left[\int_\Omega \sum_{i=1}^d \partial_i k(x,\cdot)\partial_i h(x)\, p_0(x)\, dx\right]_\sim, \qquad [h]_\sim \in \mathcal{W}_2^\sim(\Omega, p_0).$$

Note that for $[h]_\sim \in \mathcal{W}_2^\sim(\Omega, p_0)$, the derivatives $\partial_i h$ do not depend on the choice of a representative element almost surely w.r.t. p_0, and thus the above integrals are well defined. Having constructed $W_2(\Omega, p_0)$, it is clear that $J(p_0\|p_f) = \frac{1}{2}\|[f_\star]_\sim - I_k f\|_{W_2}^2$, which means estimating p_0 is equivalent to estimating $f_\star \in W_2(\Omega, p_0)$ by $f \in \mathcal{F}$. With all these preparations, we are now ready to present a result (proved in Section 8.11) on consistency and convergence rate for $p_{f_{\lambda,n}}$ without assuming the existence of f^*.

Theorem 12 *Let p_0, $q_0 \in C^1(\Omega)$ be probability densities such that $J(p_0\|q_0) < \infty$. Assume that* **(A)**-**(D)** *hold with $\varepsilon = 2$ and $\chi := d\sup_{x\in\Omega, i\in[d]} \partial_i\partial_{i+d}k(x,x) < \infty$. Then the following*

hold.

(i) As $\lambda \to 0$, $\lambda n \to \infty$ *and* $n \to \infty$, $J(p_0\|p_{f_{\lambda,n}}) \to \inf_{p\in\mathcal{P}} J(p_0\|p)$.

(ii) Define $f_\star := \log \frac{p_0}{q_0}$. *If* $[f_\star]_\sim \in \overline{\mathcal{R}(T_k)}$, *then*

$$J(p_0\|p_{f_{\lambda,n}}) \to 0 \;\; as \;\; \lambda \to 0,\, \lambda n \to \infty \; and \; n \to \infty.$$

In addition, if $[f_\star]_\sim \in \mathcal{R}(T_k^\beta)$ *for some* $\beta > 0$, *then for* $\lambda = n^{-\max\left\{\frac{1}{3}, \frac{1}{2\beta+1}\right\}}$

$$J(p_0\|p_{f_{\lambda,n}}) = O_{p_0}\left(n^{-\min\left\{\frac{2}{3}, \frac{2\beta}{2\beta+1}\right\}}\right)$$

. (iii) If $\|E_k^{-1}\| < \infty$ *and* $\|T_k^{-1}\| < \infty$, *then* $J(p_0\|p_{f_{\lambda,n}}) = O_{p_0}\left(n^{-1}\right)$ *with* $\lambda = n^{-\frac{1}{2}}$.

Remark *(i)* The result in Theorem 12(ii) is particularly interesting as it shows that $[f_\star]_\sim \in W_2(\Omega, p_0)\backslash I_k(\mathcal{H})$ can be consistently estimated by $f_{\lambda,n} \in \mathcal{H}$, which in turn implies that certain $p_0 \notin \mathcal{P}$ can be consistently estimated by $p_{f_{\lambda,n}} \in \mathcal{P}$. In particular, if S_k is injective, then $I_k(\mathcal{H})$ is dense in $W_2(\Omega, p_0)$ w.r.t. $\|\cdot\|_{W_2}$, which implies $\inf_{p\in\mathcal{P}} J(p_0\|p) = 0$ though there does not exist $f^* \in \mathcal{H}$ for which $J(p_0\|p_{f^*}) = 0$. While Theorem 10 cannot handle this situation, (i) and (ii) in Theorem 12 coincide showing that $p_0 \notin \mathcal{P}$ can be consistently estimated by $p_{f_{\lambda,n}} \in \mathcal{P}$. While the question of when $I_k(\mathcal{H})$ is dense in $W_2(\Omega, p_0)$ is open, we refer the reader to Section B.4 for a related discussion.

(ii) Replicating the proof of Theorem 4.6 in Steinwart and Scovel (2012), it is easy to show that for all $0 < \gamma < 1$, $\mathcal{R}(T_k^{\gamma/2}) = [W_2(\Omega, p_0), I_k(\mathcal{H})]_{\gamma,2}$, where the r.h.s. is an interpolation space obtained through the real interpolation of $W_2(\Omega, p_0)$ and $I_k(\mathcal{H})$ (see Section A.5 for the notation and definition). Here $I_k(\mathcal{H})$ is endowed with the Hilbert space structure by $I_k(\mathcal{H}) \cong \mathcal{H}/\mathcal{H} \cap \mathbb{R}$. This interpolation space interpretation means that, for $\beta \geq \frac{1}{2}$, $\mathcal{R}(T_k^\beta) \subset \mathcal{H}$ modulo constant functions. It is nice to note that the rates in Theorem 12(ii) for $\beta \geq \frac{1}{2}$ match with the rates in Theorem 7 (i.e., the well-specified case) w.r.t. J for $0 \leq \beta \leq \frac{1}{2}$. We highlight the fact that $\beta = 0$ corresponds to $\mathcal{H}$ in Theorem 7 whereas $\beta = \frac{1}{2}$ corresponds to $\mathcal{H}$ in Theorem 12(ii) and therefore the range of comparison is for $\beta \geq \frac{1}{2}$ in Theorem 12(ii) versus $0 \leq \beta \leq \frac{1}{2}$ in Theorem 7. In contrast, Theorem 10 is very limited as it only provides a rate for the convergence of $J(p_0||p_{f_{\lambda,n}})$ to $\inf_{p\in\mathcal{P}} J(p_0||p)$ assuming that f^* is sufficiently smooth.

Based on the observation (i) in the above remark that $\inf_{p\in\mathcal{P}} J(p_0\|p) = 0$ if $I_k(\mathcal{H})$ is dense in $W_2(\Omega, p_0)$ w.r.t. $\|\cdot\|_{W_2}$, it is possible to obtain an approximation result for $\mathcal{P}$ (similar to those discussed in Section 3) w.r.t. Fisher divergence as shown below, whose proof is provided in Section 8.12.

Proposition 13 *Suppose* $\Omega \subset \mathbb{R}^d$ *is non-empty and open. Let* $q_0 \in C^1(\Omega)$ *be a probability density and*

$$\mathcal{P}_{\text{FD}} := \left\{p \in C^1(\Omega) : \int_\Omega p(x)\,dx = 1, p(x) \geq 0,\, \forall\, x \in \Omega \text{ and } J(p\|q_0) < \infty\right\}.$$

For any $p \in \mathcal{P}_{\text{FD}}$, *if* $I_k(\mathcal{H})$ *is dense in* $W_2(\Omega, p)$ *w.r.t.* $\|\cdot\|_{W_2}$, *then for every* $\epsilon > 0$, *there exists* $\tilde{p} \in \mathcal{P}$ *such that* $J(p\|\tilde{p}) \leq \epsilon$.

6. Numerical Simulations

We have proposed an estimator of p_0 that is obtained by minimizing the regularized empirical Fisher divergence and presented its consistency along with convergence rates. As discussed in Section 1, however one can simply ignore the structure of $\mathcal{P}$ and estimate p_0 in a completely non-parametric fashion, for example using the kernel density estimator (KDE). In fact, consistency and convergence rates of KDE are also well-studied (Tsybakov, 2009, Chapter 1) and the kernel density estimator is very simple to compute—requiring only $O(n)$ computations—compared to the proposed estimator, which is obtained by solving a linear system of size $nd \times nd$. This raises questions about the applicability of the proposed estimator in practice, though it is very well known that KDE performs poorly for moderate to large d (Wasserman, 2006, Section 6.5). In this section, we numerically demonstrate that the proposed score matching estimator performs significantly better than the KDE, and in particular, that the advantage with the proposed estimator grows as d gets large. Note further that the maximum likelihood approach of Barron and Sheu (1991) and Fukumizu (2009) does not yield estimators that are practically feasible, and therefore to the best of our knowledge, the proposed estimator is the only viable estimator for estimating densities through $\mathcal{P}$.

In the following, we consider two simple scenarios of estimating a multivariate normal and mixture of normals using the proposed estimator and demonstrate the superior performance of the proposed estimator over KDE. Inspired by this preliminary empirical investigation, recently, the proposed estimator has been explored in two concrete applications of gradient-free adaptive MCMC sampler (Strathmann et al., 2015) and graphical model structure learning (Sun et al., 2015) where the superiority of working with the infinite dimensional family is demonstrated. We would like to again highlight that the goal of this work is not to construct density estimators that improve upon KDE but to provide a novel modeling technique of approximating an unknown density by a rich parametric family of densities with the parameter being infinite dimensional in contrast to the classical approach of finite dimensional approximation.

We consider the problems of estimating a standard normal distribution on $\mathbb{R}^d$, $N(0, I_d)$ and mixture of Gaussians,

$$p_0(x) = \frac{1}{2}\phi_d(x; \alpha\mathbf{1}_n, I_d) + \frac{1}{2}\phi_d(x; \beta\mathbf{1}_n, I_d)$$

through the score matching approach and KDE, and compare their estimation accuracies. Here $\phi_d(x; \mu, \Sigma)$ is the p.d.f. of $N(\mu, \Sigma I_d)$. By choosing the kernel, $k(x, y) = \exp(-\frac{\|x-y\|_2^2}{2\sigma^2}) + r(x^T y + c)^2$, which is a Gaussian plus polynomial of degree 2, it is easy to verify that Gaussian distributions lie in $\mathcal{P}$, and therefore the first problem considers the well-specified case while the second problem deals with the misspecified case. In our simulations, we chose $r = 0.1$, $c = 0.5$, $\alpha = 4$ and $\beta = -4$. The base measure of the exponential family is $N(0, 10^2 I_d)$. The bandwidth parameter σ is chosen by cross-validation (CV) of the objective function $\hat{J}_\lambda$ (see Theorem 4(iv)) within the parameter set $\{0.1, 0.2, 0.4, 0.6, 0.8, 1, 1.2, 1.4, 1.6\} \times \sigma_*$, where σ_* is the median of pairwise distances of data, and the regularization parameter λ is set as $\lambda = 0.1 \times n^{-1/3}$ with sample size n. For KDE, the Gaussian kernel is used for the smoothing kernel, and the bandwidth parameter is chosen by CV from

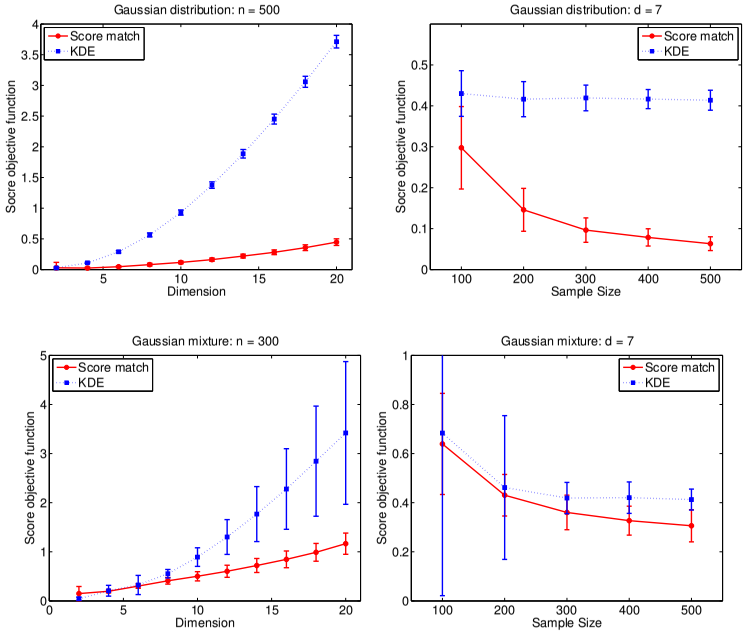


Figure 1: Experimental comparisons with the score objective function: proposed method and kernel density estimator

$\{0.02, 0.04, 0.06, 0.08, 0.1, 0.2, 0.4, 0.6, 0.8, 1.0\} \times \sigma_*$; where for both the methods, 5-fold CV is applied. Since it is difficult to accurately estimate the normalization constant in the proposed method, we use two methods to evaluate the accuracy of estimation. One is the objective function for the score matching method,

$$\tilde{J}(p) = \sum_{i=1}^{d} \int_{\Omega} \left(\frac{1}{2} \left| \partial_i \log p(x) \right|^2 + \partial_i^2 \log p(x) \right) p_0(x) dx,$$

and the other is correlation of the estimator with the true density function,

$$\mathrm{Cor}(p, p_0) := \frac{\mathbb{E}_R[p(X) p_0(X)]}{\sqrt{\mathbb{E}_R[p(X)^2] \mathbb{E}_R[p_0(X)^2]}},$$

where R is a probability distribution. For R, we use the empirical distribution based on 10000 random samples drawn i.i.d. from $p_0(x)$.

Figures 1 and 2 show the score objective function ($\tilde{J}(p)$) and the correlation ($\mathrm{Cor}(p, p_0)$) (along with their standard deviation as error bars) of the proposed estimator and KDE for the tasks of estimating a Gaussian and a mixture of Gaussians, for different sample sizes

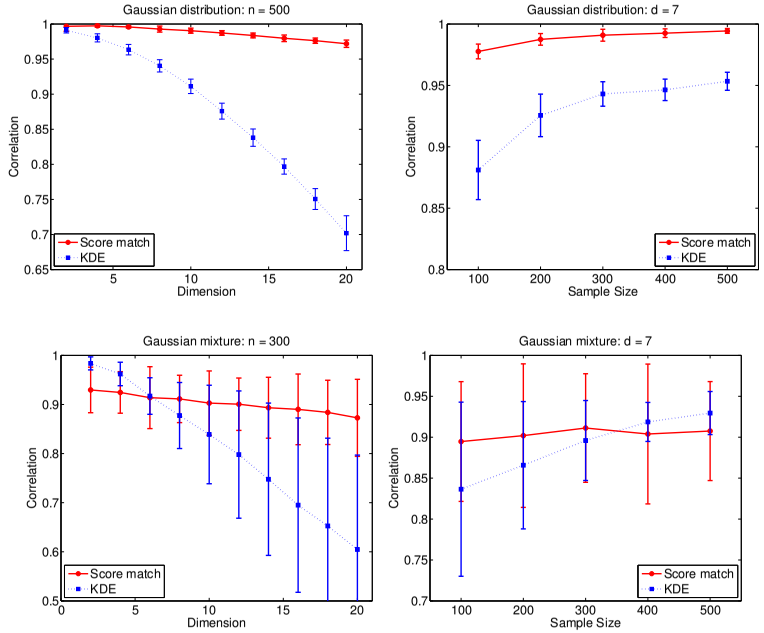


Figure 2: Experimental comparisons with the correlation: proposed method and kernel density estimator

(n) and dimensions (d). From the figures, we see that the proposed estimator outperforms (i.e., lower function values) KDE in all the cases except the low dimensional cases ($(n, d) = (500, 2)$ for the Gaussian, and $(n, d) = (300, 2), (300, 4)$ for the Gaussian mixture). In the case of the correlation measure, the score matching method yields better results (i.e., higher correlation) besides in the Gaussian mixture cases of $d = 2, 4, 6$ (Fig.2, lower-left) and some cases of $d = 7$ (lower-right). The proposed method shows an increased advantage over KDE as the dimensionality increases, thereby demonstrating the advantage of the proposed estimator for high dimensional data.

7. Summary & Discussion

We have considered an infinite dimensional generalization, $\mathcal{P}$, of the finite-dimensional exponential family, where the densities are indexed by functions in a reproducing kernel Hilbert space (RKHS), $\mathcal{H}$. We showed that $\mathcal{P}$ is a rich object that can approximate a large class of probability densities arbitrarily well in Kullback-Leibler divergence, and addressed the main question of estimating an unknown density, p_0 from finite samples drawn i.i.d. from it, in well-specified ($p_0 \in \mathcal{P}$) and misspecified ($p_0 \notin \mathcal{P}$) settings. We proposed a density

estimator based on minimizing the regularized version of the empirical Fisher divergence, which results in solving a simple finite-dimensional linear system. Our estimator provides a computationally efficient alternative to maximum likelihood based estimators, which suffer from the computational intractability of the log-partition function. The proposed estimator is also shown to empirically outperform the classical kernel density estimator, with advantage increasing as the dimension of the space increases. In addition to these computational and empirical results, we have established the consistency and convergence rates under certain smoothness assumptions (e.g., $\log p_0 \in \mathcal{R}(C^\beta)$) for both well-specified and misspecified scenarios.

Three important questions still remain open in this work which we intend to address in our future work. First, the assumption $\log p_0 \in \mathcal{R}(C^\beta)$ is not well understood. Though we presented a necessary condition for this assumption (with $\beta = 1$) to hold for bounded continuous translation invariant kernels on $\mathbb{R}^d$, obtaining a sufficient condition can throw light on the minimax optimality of the proposed estimator. Another alternative is to directly study the minimax optimality of the rates for $0 < \beta \leq 1$ (for $\beta > 1$, we showed that the above mentioned rates can be improved by an appropriate choice of the regularizer) by obtaining minimax lower bounds under the source condition $\log p_0 \in \mathcal{R}(C^\beta)$ and the eigenvalue decay rate of C, using the ideas in DeVore et al. (2004). Second, the proposed estimator depends on the regularization parameter, which in turn depends on the smoothness scale β. Since β is not known in practice, it is therefore of interest to construct estimators that are adaptive to unknown β. Third, since the proposed estimator is computationally expensive as it involves solving a linear system of size $nd \times nd$, it is important to study either alternate estimators or efficient implementations of the proposed estimator to improve the applicability of the method.

8. Proofs

We provide proofs of the results presented in Sections 3–5.

8.1 Proof of Proposition 1

Sriperumbudur et al. (2011, Proposition 5) showed that $\mathcal{H}$ is dense in $C_0(\Omega)$ w.r.t. uniform norm if and only if k satisfies (5). Therefore, the denseness in L^1, KL and Hellinger distances follow trivially from Lemma A.1. For L^r norm ($r > 1$), the denseness follows by using the bound $\|p_f - p_g\|_{L^r(\Omega)} \leq 2e^{2\|f-g\|_\infty} e^{2\|f\|_\infty} \|f - g\|_\infty \|q_0\|_{L^r(\Omega)}$ obtained from Lemma A.1(i) with $f \in C_0(\Omega)$ and $g \in \mathcal{H}$. ■

8.2 Proof of Corollary 2

For any $p \in \mathcal{P}_c$, define $p_\delta := \frac{p+\delta q_0}{1+\delta}$. Note that $p_\delta(x) > 0$ for all $x \in \Omega$ and $\|p - p_\delta\|_{L^r(\Omega)} = \frac{\delta \|p-q_0\|_{L^r(\Omega)}}{1+\delta}$, implying that $\lim_{\delta\to 0} \|p - p_\delta\|_{L^r(\Omega)} = 0$ for any $1 \leq r \leq \infty$. This means, for any $\epsilon > 0$, $\exists \delta_\epsilon > 0$ such that for any $0 < \theta < \delta_\epsilon$, we have $\|p - p_\theta\|_{L^r(\Omega)} \leq \epsilon$, where $p_\theta(x) > 0$ for all $x \in \Omega$.

Define $f := \log \frac{p_\theta}{q_0} - c_\theta$ where $c_\theta := \log \frac{\ell+\theta}{1+\theta}$. It is clear that $f \in C(\Omega)$ since $p, q \in C(\Omega)$. Fix any $\eta > 0$ and define

$$A := \{x : f(x) \geq \eta\} = \left\{ x : \frac{p(x)}{q_0(x)} - \ell \geq (\ell + \theta)\left(e^\eta - 1\right) \right\}.$$

Since $\frac{p}{q_0} - \ell \in C_0(\Omega)$, it is clear that A is compact and so $f \in C_0(\Omega)$. Also, it is easy to verify that $p_\theta = e^{f - A(f)} q_0$ which implies $p_\theta \in \mathcal{P}_0$, where $\mathcal{P}_0$ is defined in Proposition 1. This means, for any $\epsilon > 0$, there exists $p_g \in \mathcal{P}$ such that $\|p_\theta - p_g\|_{L^r(\Omega)} \leq \epsilon$ under the assumption that $q_0 \in L^1(\Omega) \cap L^r(\Omega)$. Therefore $\|p - p_g\|_{L^r(\Omega)} \leq 2\epsilon$ for any $1 \leq r \leq \infty$, which proves the denseness of $\mathcal{P}$ in $\mathcal{P}_c$ w.r.t. L^r norm for any $1 \leq r \leq \infty$. Since $h(p, q) \leq \sqrt{\|p - q\|_{L^1(\Omega)}}$ for any probability densities p, q, the denseness in Hellinger distance follows.

We now prove the denseness in KL divergence by noting that

$$\begin{aligned} KL(p\|p_\delta) = \int_{\{p>0\}} p \log \frac{p + p\delta}{p + q_0\delta}\, dx &\leq \int_{\{p>0\}} p \left(\frac{p + p\delta}{p + q_0\delta} - 1 \right) dx \\ &= \delta \int_{p>0} (p - q_0) \frac{p}{p + q_0\delta}\, dx \leq \delta \|p - q_0\|_{L^1(\Omega)} \leq 2\delta, \end{aligned}$$

which implies $\lim_{\delta\to 0} KL(p\|p_\delta) = 0$. This implies, for any $\epsilon > 0$, $\exists \delta_\epsilon > 0$ such that for any $0 < \theta < \delta_\epsilon$, $KL(p\|p_\theta) \leq \epsilon$. Arguing as above, we have $p_\theta \in \mathcal{P}_0$, i.e., there exists $f \in C_0(\Omega)$ such that $p_\theta = \frac{e^f q_0}{\int e^f q_0\, dx}$. Since $\mathcal{H}$ is dense in $C_0(\Omega)$, for any $f \in C_0(\Omega)$ and any $\epsilon > 0$, there exists $g \in \mathcal{H}$ such that $\|f - g\|_\infty \leq \epsilon$. For $p_g \in \mathcal{P}$, since $\int p \log \frac{p_\theta}{p_g}\, dx \leq \left\| \log \frac{p_\theta}{p_g} \right\|_\infty \leq 2\|f - g\|_\infty \leq 2\epsilon$, we have

$$KL(p\|p_g) = \int_\Omega p \log \frac{p}{p_g}\, dx = \int_\Omega p \log \frac{p}{p_\theta}\, dx + \int_\Omega p \log \frac{p_\theta}{p_g}\, dx \leq 3\epsilon$$

and the result follows. ■

8.3 Proof of Theorem 4

(i) By the reproducing property of $\mathcal{H}$, since $\partial_i f(x) = \langle f, \partial_i k(x, \cdot)\rangle_{\mathcal{H}}$ for all $i \in [d]$, it is easy to verify that

$$\begin{aligned} J(f) &= \frac{1}{2} \int_\Omega p_0(x) \sum_{i=1}^d \langle f - f_0, \partial_i k(x, \cdot)\rangle_{\mathcal{H}}^2\, dx \\ &= \frac{1}{2} \int_\Omega p_0(x) \sum_{i=1}^d \langle f - f_0, (\partial_i k(x, \cdot) \otimes \partial_i k(x, \cdot))\,(f - f_0)\rangle_{\mathcal{H}}\, dx \\ &= \frac{1}{2} \int_\Omega p_0(x) \langle f - f_0, C_x(f - f_0)\rangle_{\mathcal{H}}\, dx, \end{aligned} \tag{11}$$

where in the second line, we used $\langle a, b\rangle_H^2 = \langle a, b\rangle_H \langle a, b\rangle_H = \langle a, (b \otimes b)a\rangle_H$ for $a, b \in H$ with H being a Hilbert space and

$$C_x := \sum_{i=1}^d \partial_i k(x, \cdot) \otimes \partial_i k(x, \cdot). \tag{12}$$

Observe that for all $x \in \Omega$, C_x is a Hilbert-Schmidt operator as $\|C_x\|_{HS} \leq \sum_{i=1}^d \|\partial_i k(x,\cdot)\|_{\mathcal{H}}^2 = \sum_{i=1}^d \partial_i \partial_{i+d} k(x,x) < \infty$ and $(f-f_0)\otimes(f-f_0)$ is also Hilbert-Schmidt as $\|(f-f_0)\otimes(f-f_0)\|_{HS} = \|f-f_0\|_{\mathcal{H}}^2 < \infty$. Therefore, (11) is equivalent to

$$J(f) = \frac{1}{2}\int_\Omega p_0(x)\,\langle (f-f_0)\otimes(f-f_0), C_x\rangle_{HS}\,dx.$$

Since the first condition in **(D)** implies $\int_\Omega \|C_x\|_{HS} p_0(x)\,dx < \infty$, C_x is p_0-integrable in the Bochner sense (see Diestel and Uhl, 1977, Definition 1 and Theorem 2), and therefore it follows from Diestel and Uhl (1977, Theorem 6) that

$$J(f) = \frac{1}{2}\left\langle (f-f_0)\otimes(f-f_0), \int_\Omega C_x\, p_0(x)\,dx\right\rangle_{HS},$$

where $C := \int_\Omega C_x\, p_0(x)\,dx$ is the Bochner integral of C_x, thereby yielding (6).

We now show that C is trace-class. Let $(e_l)_{l\in\mathbb{N}}$ be an orthonormal basis in $\mathcal{H}$ (a countable ONB exists as $\mathcal{H}$ is separable—see Remark 3(i)). Define $B := \sum_l \langle Ce_l, e_l\rangle_{\mathcal{H}}$ so that

$$B = \sum_l \int_\Omega \langle e_l, C_x e_l\rangle_{\mathcal{H}} p_0(x)\,dx = \sum_l \int_\Omega \sum_{i=1}^d \langle e_l, \partial_i k(x,\cdot)\rangle_{\mathcal{H}}^2 p_0(x)\,dx$$
$$\overset{(*)}{=} \int_\Omega \sum_{i\in[d],l} \langle e_l, \partial_i k(x,\cdot)\rangle_{\mathcal{H}}^2 p_0(x)\,dx \overset{(**)}{=} \int_\Omega \sum_{i=1}^d \|\partial_i k(x,\cdot)\|_{\mathcal{H}}^2 p_0(x)\,dx < \infty,$$

which means C is trace-class and therefore compact. Here, we used monotone convergence theorem in $(*)$ and Parseval's identity in $(**)$. Note that C is positive since $\langle f, Cf\rangle_{\mathcal{H}} = \int_\Omega p_0(x)\,\|\nabla f\|_2^2\,dx \geq 0,\ \forall f \in \mathcal{H}$.

(ii) From (6), we have $J(f) = \frac{1}{2}\langle f, Cf\rangle_{\mathcal{H}} - \langle f, Cf_0\rangle_{\mathcal{H}} + \frac{1}{2}\langle f_0, Cf_0\rangle_{\mathcal{H}}$. Using $\partial_i f_0(x) = \partial_i \log p_0(x) - \partial_i \log q_0(x)$ for all $i \in [d]$, we obtain that for any $f \in \mathcal{H}$,

$$\begin{aligned}
\langle f, Cf_0\rangle_{\mathcal{H}} &= \int_\Omega p_0(x)\sum_{i=1}^d \partial_i f(x)\partial_i f_0(x)\,dx \\
&= \int_\Omega \sum_{i=1}^d \partial_i f(x)\partial_i p_0(x)\,dx - \int_\Omega p_0(x)\sum_{i=1}^d \partial_i f(x)\partial_i \log q_0(x)\,dx \\
&\overset{(b)}{=} -\int_\Omega p_0(x)\sum_{i=1}^d \partial_i^2 f(x)\,dx - \int_\Omega p_0(x)\sum_{i=1}^d \partial_i f(x)\partial_i \log q_0(x)\,dx \\
&= -\int_\Omega p_0(x)\left\langle f, \overbrace{\sum_{i=1}^d \partial_i^2 k(x,\cdot) + \partial_i k(x,\cdot)\partial_i \log q_0(x)}^{\xi_x}\right\rangle_{\mathcal{H}} dx \overset{(c)}{=} \langle f, -\xi\rangle_{\mathcal{H}}, \quad (13)
\end{aligned}$$

where (b) follows from integration by parts under **(C)** and the equality in (c) is valid as ξ_x is Bochner p_0-integrable under **(D)** with $\varepsilon = 1$. Therefore $Cf_0 = -\xi$. For the third term, $\langle f_0, Cf_0\rangle_{\mathcal{H}} = \int_\Omega p_0(x)\sum_{i=1}^d (\partial_i f_0(x))^2\,dx$ and the result follows.

(iii) Define $c_0 := J(p_0\|q_0)$. For any $\lambda > 0$, it is easy to verify that

$$J_\lambda(f) = \frac{1}{2}\|(C+\lambda I)^{1/2} f + (C+\lambda I)^{-1/2}\xi\|_{\mathcal{H}}^2 - \frac{1}{2}\langle \xi, (C+\lambda I)^{-1}\xi\rangle_{\mathcal{H}} + c_0.$$

Clearly, $J_\lambda(f)$ is minimized if and only if $(C+\lambda I)^{1/2} f = -(C+\lambda I)^{-1/2}\xi$ and therefore $f_\lambda = -(C+\lambda I)^{-1}\xi$ is the unique minimizer of $J_\lambda(f)$.

(iv) Since (iv) is similar to (iii) with C replaced by $\hat{C}$ and ξ replaced by $\hat{\xi}$, we obtain $f_{\lambda,n} = (\hat{C}+\lambda I)^{-1}\hat{\xi}$. ■

8.4 Proof of Theorem 5

We prove the result based on the general representer theorem (Theorem A.2). From Theorem 4(iv), we have

$$\begin{aligned}
f_{\lambda,n} &= \arg\inf_{f\in\mathcal{H}} \frac{1}{2}\langle f, \hat{C}f\rangle_{\mathcal{H}} + \langle f, \hat{\xi}\rangle_{\mathcal{H}} + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \\
&= \arg\inf_{f\in\mathcal{H}} \frac{1}{2n}\sum_{a=1}^n\sum_{i=1}^d \langle f, \partial_i k(X_a,\cdot)\rangle_{\mathcal{H}}^2 + \langle f, \hat{\xi}\rangle_{\mathcal{H}} + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \\
&= \arg\inf_{f\in\mathcal{H}} V(\langle f,\phi_1\rangle_{\mathcal{H}}, \ldots, \langle f,\phi_{nd}\rangle_{\mathcal{H}}, \langle f,\phi_{nd+1}\rangle_{\mathcal{H}}) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2,
\end{aligned}$$

where $V(\theta_1,\ldots,\theta_{nd},\theta_{nd+1}) := \frac{1}{2n}\sum_{a=1}^n\sum_{i=1}^d \theta_{(a-1)d+i}^2 + \theta_{nd+1}$, $\phi_{(a-1)d+i} := \partial_i k(X_a,\cdot)$, $a \in [n]$, $i \in [d]$ and $\phi_{nd+1} := \hat{\xi}$. Therefore, it follows from Theorem A.2 that

$$f_{\lambda,n} = \delta\hat{\xi} + \sum_{a=1}^n\sum_{i=1}^d \beta_{(a-1)d+i}\phi_{(a-1)d+i} \quad (14)$$

where δ and $\boldsymbol{\beta}$ satisfy

$$\lambda\begin{pmatrix}\boldsymbol{\beta}\\ \delta\end{pmatrix} + \nabla V\left(\boldsymbol{K}\begin{pmatrix}\boldsymbol{\beta}\\ \delta\end{pmatrix}\right) = 0 \quad (15)$$

with $\boldsymbol{K} = \begin{pmatrix}\boldsymbol{G} & \boldsymbol{h}\\ \boldsymbol{h}^T & \|\hat{\xi}\|_{\mathcal{H}}^2\end{pmatrix}$. Since $\nabla V\begin{pmatrix}\boldsymbol{z}\\ t\end{pmatrix} = \begin{pmatrix}\frac{1}{n}\boldsymbol{z}\\ 1\end{pmatrix}$, (15) reduces to $\lambda\delta + 1 = 0$ and $\lambda\boldsymbol{\beta} + \frac{1}{n}\boldsymbol{G}\boldsymbol{\beta} + \frac{\delta}{n}\boldsymbol{h} = 0$ yielding $\delta = -\frac{1}{\lambda}$ and $(\frac{1}{n}\boldsymbol{G} + \lambda I)\boldsymbol{\beta} = \frac{1}{n\lambda}\boldsymbol{h}$. ■

Remark Instead of using the general representer theorem (Theorem A.2), it is possible to see that the standard representer theorem (Kimeldorf and Wahba, 1971; Schölkopf et al., 2001) gives a similar, but slightly different linear system, and the solutions are the same if $\boldsymbol{K}$ is non-singular. The general representer theorem yields that $\boldsymbol{\beta}$ and δ are solution to $\boldsymbol{F}\begin{pmatrix}\boldsymbol{\beta}\\ \delta\end{pmatrix} = \begin{pmatrix}\boldsymbol{0}\\ 1\end{pmatrix}$, where $\boldsymbol{F} = \begin{pmatrix}\frac{1}{n}\boldsymbol{G} + \lambda I & \frac{1}{n}\boldsymbol{h}\\ \boldsymbol{0}^T & \lambda\end{pmatrix}$. On the other hand, by using the standard representer theorem, it is easy to show that $f_{\lambda,n}$ has the form in (14) with δ and $\boldsymbol{\beta}$ being solution to $\boldsymbol{K}\boldsymbol{F}\begin{pmatrix}\boldsymbol{\beta}\\ \delta\end{pmatrix} = \boldsymbol{K}\begin{pmatrix}\boldsymbol{0}\\ 1\end{pmatrix}$. Clearly, both the solutions match if $\boldsymbol{K}$ is invertible while the latter has many solutions if $\boldsymbol{K}$ is not invertible.

8.5 Proof of Theorem 6

Consider

$$\begin{aligned} f_{\lambda,n} - f_\lambda &= -(\hat{C}+\lambda I)^{-1}\Big(\hat{\xi} + (\hat{C}+\lambda I) f_\lambda\Big) \overset{(*)}{=} -(\hat{C}+\lambda I)^{-1}\left(\hat{\xi} + \hat{C} f_\lambda + C(f_0 - f_\lambda)\right) \\ &= (\hat{C}+\lambda I)^{-1}(C-\hat{C})(f_\lambda - f_0) - (\hat{C}+\lambda I)^{-1}(\hat{\xi} + \hat{C} f_0) \\ &= (\hat{C}+\lambda I)^{-1}(C-\hat{C})(f_\lambda - f_0) - (\hat{C}+\lambda I)^{-1}(\hat{\xi} - \xi) + (\hat{C}+\lambda I)^{-1}(C-\hat{C}) f_0, \end{aligned}$$

where we used $\lambda f_\lambda = C(f_0 - f_\lambda)$ in $(*)$. Define $S_1 := \|(\hat{C}+\lambda I)^{-1}(C-\hat{C})(f_\lambda - f_0)\|_{\mathcal{H}}$, $S_2 := \|(\hat{C}+\lambda I)^{-1}(\hat{\xi}-\xi)\|_{\mathcal{H}}$ and $S_3 := \|(\hat{C}+\lambda I)^{-1}(C-\hat{C}) f_0\|_{\mathcal{H}}$ so that

$$\|f_{\lambda,n} - f_0\|_{\mathcal{H}} \le \|f_{\lambda,n} - f_\lambda\|_{\mathcal{H}} + \|f_\lambda - f_0\|_{\mathcal{H}} \le S_1 + S_2 + S_3 + \mathcal{A}_0(\lambda), \tag{16}$$

where $\mathcal{A}_0(\lambda) := \|f_\lambda - f_0\|_{\mathcal{H}}$. We now bound S_1, S_2 and S_3 using Proposition A.4. Note that $C = \int_\Omega C_x\, p_0(x)\, dx$ where C_x is defined in (12) is a positive, self-adjoint, trace-class operator and **(D)** (with $\varepsilon = 2$) implies that

$$\int_\Omega \|C_x\|_{HS}^2 p_0(x)\, dx \le \int_\Omega \left(\sum_{i=1}^d \|\partial_i k(x,\cdot)\|_{\mathcal{H}}^2\right)^2 p_0(x)\, dx \le d \sum_{i=1}^d \int_\Omega \|\partial_i k(x,\cdot)\|_{\mathcal{H}}^4\, p_0(x)\, dx < \infty.$$

Therefore, by Proposition A.4(i,iii),

$$S_1 \le \|(\hat{C}+\lambda I)^{-1}\| \|(C-\hat{C})(f_\lambda - f_0)\|_{\mathcal{H}} = O_{p_0}\left(\frac{\mathcal{A}_0(\lambda)}{\lambda\sqrt{n}}\right) \tag{17}$$

and

$$S_2 \le \|(\hat{C}+\lambda I)^{-1}\| \|\hat{\xi} - \xi\|_{\mathcal{H}} = O_{p_0}\left(\frac{1}{\lambda\sqrt{n}}\right), \tag{18}$$

where by using the technique in the proof of Proposition A.4(i), we show below that $\|\hat{\xi} - \xi\|_{\mathcal{H}} = O_{p_0}(n^{-1/2})$. Note that $\mathbb{E}_{p_0}\|\hat{\xi}-\xi\|_{\mathcal{H}}^2 = \frac{\int_\Omega \|\xi_x\|_{\mathcal{H}}^2 p_0(x)\, dx - \|\xi\|_{\mathcal{H}}^2}{n} \le \frac{\int_\Omega \|\xi_x\|_{\mathcal{H}}^2 p_0(x)\, dx}{n}$, where $\xi_x \in \mathcal{H}$ is defined in (13) and **(D)** (with $\varepsilon = 2$) implies that $\int_\Omega \|\xi_x\|_{\mathcal{H}}^2 p_0(x)\, dx < \infty$. Therefore $\|\hat{\xi} - \xi\|_{\mathcal{H}} = O_{p_0}(n^{-1/2})$ follows from an application of Chebyshev's inequality. Again using Proposition A.4(i,iii), we obtain that

$$S_3 \le \|(\hat{C}+\lambda I)^{-1}\| \|(C-\hat{C}) f_0\|_{\mathcal{H}} = O_{p_0}\left(\frac{1}{\lambda\sqrt{n}}\right). \tag{19}$$

Using the bounds in S_1, S_2 and S_3 in (16), we obtain

$$\|f_{\lambda,n} - f_0\|_{\mathcal{H}} = O_{p_0}\left(\frac{1}{\lambda\sqrt{n}} + \frac{\mathcal{A}_0(\lambda)}{\lambda\sqrt{n}}\right) + \mathcal{A}_0(\lambda). \tag{20}$$

(i) By Proposition A.3(i), we have that $\mathcal{A}_0(\lambda) \to 0$ as $\lambda \to 0$ if $f_0 \in \overline{\mathcal{R}(C)}$. Therefore, it follows from (20) that $\|f_{\lambda,n} - f_0\|_{\mathcal{H}} \to 0$ as $\lambda \to 0$, $\lambda\sqrt{n} \to \infty$ and $n \to \infty$.

(ii) If $f_0 \in \mathcal{R}(C^\beta)$ for $\beta > 0$, it follows from Proposition A.3(ii) that

$$\mathcal{A}_0(\lambda) \le \max\{1, \|C\|^{\beta-1}\} \|C^{-\beta} f_0\|_{\mathcal{H}} \lambda^{\min\{1,\beta\}}$$

and therefore the result follows by choosing $\lambda = n^{-\max\left\{\frac{1}{4}, \frac{1}{2(\beta+1)}\right\}}$.

(iii) Note that

$$S_1 = \|(\hat{C}+\lambda I)^{-1}(C-\hat{C})(f_\lambda - f_0)\|_{\mathcal{H}} \le \|C(\hat{C}+\lambda I)^{-1}\| \|C^{-1}\| \|(C-\hat{C})(f_\lambda - f_0)\|_{\mathcal{H}},$$

$$S_2 = \|(\hat{C}+\lambda I)^{-1}(\hat{\xi}-\xi)\|_{\mathcal{H}} \le \|C(\hat{C}+\lambda I)^{-1}\| \|C^{-1}\| \|\hat{\xi} - \xi\|_{\mathcal{H}},$$

$$S_3 = \|(\hat{C}+\lambda I)^{-1}(C-\hat{C}) f_0\|_{\mathcal{H}} \le \|C(\hat{C}+\lambda I)^{-1}\| \|C^{-1}\| \|(C-\hat{C}) f_0\|_{\mathcal{H}}$$

and

$$\mathcal{A}_0(\lambda) = \|f_\lambda - f_0\|_{\mathcal{H}} \le \|C^{-1}\| \|C(f_\lambda - f_0)\|_{\mathcal{H}}.$$

It follows from Proposition A.4(v) that $\|C(\hat{C}+\lambda I)^{-1}\| \lesssim 1$ for $n \ge \frac{c}{\lambda^2}$ where c is a sufficiently large constant that depends on $\sum_{i=1}^d \int_\Omega (\partial_i \partial_{i+d} k(x,x))^2 p_0(x)\, dx$ but not on n and λ. Using the bounds on $\|(C-\hat{C})(f_\lambda - f_0)\|_{\mathcal{H}}$, $\|\hat{\xi} - \xi\|_{\mathcal{H}}$ and $\|(C-\hat{C}) f_0\|_{\mathcal{H}}$ from part (i) and the bound on $\|C(f_\lambda - f_0)\|_{\mathcal{H}}$ from Proposition A.3(ii), we therefore obtain

$$\|f_{\lambda,n} - f_0\|_{\mathcal{H}} \lesssim O_{p_0}\left(\frac{1}{\sqrt{n}}\right) + \lambda \tag{21}$$

as $n \to \infty$ and the result follows. ■

Remark Under slightly strong assumptions on the kernel, the bound on S_1 in (17) can be improved to obtain $S_1 = O_{p_0}(n^{-1/2})$ while the one on S_3 in (19) can be refined to obtain $S_3 = O_{p_0}\left(\sqrt{\frac{\mathcal{N}(\lambda)}{\lambda n}}\right)$ where $\mathcal{N}(\lambda) := Tr((C+\lambda I)^{-1} C)$ is the intrinsic dimension of $\mathcal{H}$. Using the fact that $\mathcal{N}(\lambda) \le \frac{1}{\lambda}$, it is easy to verify that the latter is an improved bound than the one in (19). In addition S_3 dominates S_1. However, if S_2 in (18) is not improved, then S_2 dominates S_3, thereby resulting in a bound that does not capture the smoothness of k (or the corresponding $\mathcal{H}$). Unfortunately, even with a refined analysis (not reported here), we are not able to improve the bound on S_2 wherein the difficulty lies with handling ξ.

8.6 Proof of Theorem 7

Before we prove the result, we present a lemma.

Lemma 14 *Suppose* $\sup_{x\in\Omega} k(x,x) < \infty$ *and* $\operatorname{supp}(q_0) = \Omega$. *Then* $\mathcal{F} = \mathcal{H}$ *and for any* $f_0 \in \mathcal{H}$ *there exists* $\tilde{f}_0 \in \overline{\mathcal{R}(C)}$ *such that* $p_{\tilde{f}_0} = p_0$.

Proof Since $\sup_{x\in\Omega} k(x,x) < \infty$, it implies that, for every $f \in \mathcal{H}$, $\int_\Omega e^{f(x)} q_0(x)\, dx < \infty$ and hence $\mathcal{F} = \mathcal{H}$. Also, under the assumptions on k and q_0, it is easy to verify that $\operatorname{supp}(p_0) = \Omega$, which implies

$$\mathcal{N}(C) = \left\{ f \in \mathcal{H} \,:\, \int_\Omega \|\nabla f\|_2^2\, p_0(x)\, dx = 0 \right\}$$

is either $\mathbb{R}$ or $\{0\}$, where $\mathcal{N}(C)$ denotes the null space of C. Let $\tilde{f}_0$ be the orthogonal projection of f_0 onto $\overline{\mathcal{R}(C)} = \mathcal{N}(C)^\perp$. Then $\tilde{f}_0 - f_0 \in \mathbb{R}$ and therefore $p_{\tilde{f}_0} = p_{f_0}$. ■

Proof of Theorem 7. From Theorem 4(iii), $f_\lambda = (C+\lambda I)^{-1}Cf_0 = (C+\lambda I)^{-1}C\tilde{f}_0$ where the second equality follows from the proof of Lemma 14. Now, carrying out the decomposition as in the proof of Theorem 6(i), we obtain $f_{\lambda,n} - f_\lambda = (\hat{C}+\lambda I)^{-1}(C-\hat{C})(f_\lambda - \tilde{f}_0) - (\hat{C}+\lambda I)^{-1}(\hat{\xi}-\xi) + (\hat{C}+\lambda I)^{-1}(C-\hat{C})\tilde{f}_0$ and therefore,

$$\|f_{\lambda,n} - \tilde{f}_0\|_{\mathcal{H}} \leq \|(\hat{C}+\lambda I)^{-1}\| \left(\|(C-\hat{C})(f_\lambda - \tilde{f}_0)\|_{\mathcal{H}} + \|\xi - \hat{\xi}\|_{\mathcal{H}} + \|(C-\hat{C})\tilde{f}_0\|_{\mathcal{H}} \right) + \tilde{\mathcal{A}}_0(\lambda),$$

where $\tilde{\mathcal{A}}_0(\lambda) = \|f_\lambda - \tilde{f}_0\|_{\mathcal{H}}$. The bounds on these quantities follow those in the proof of Theorem 6(i) verbatim and so the consistency result in Theorem 6(i) holds for $\|f_{\lambda,n} - \tilde{f}_0\|_{\mathcal{H}}$. By Lemma 14, since $p_{f_0} = p_{\tilde{f}_0}$, it is sufficient to consider the convergence of $p_{f_{\lambda,n}}$ to $p_{\tilde{f}_0}$. Therefore, the convergence (along with rates) in L^r (for any $1 \leq r \leq \infty$), Hellinger and KL distances follow from using the bound $\|f_{\lambda,n} - \tilde{f}_0\|_\infty \leq \sqrt{\|k\|_\infty}\|f_{\lambda,n} - \tilde{f}_0\|_{\mathcal{H}}$ (obtained through the reproducing property of k) in Lemma A.1 and invoking Theorem 6.

In the following, we obtain a bound on $J(p_0\|p_{f_{\lambda,n}}) = \frac{1}{2}\|\sqrt{C}(f_{\lambda,n} - f_0)\|^2_{\mathcal{H}}$. While one can trivially use the bound $\|\sqrt{C}(f_{\lambda,n} - f_0)\|^2_{\mathcal{H}} \leq \|\sqrt{C}\|^2\|f_{\lambda,n} - f_0\|^2_{\mathcal{H}}$ to obtain a rate on $J(p_0\|p_{f_{\lambda,n}})$ through the result in Theorem 6(ii), a better rate can be obtained by carefully bounding $\|\sqrt{C}(f_{\lambda,n} - f_0)\|^2_{\mathcal{H}}$ as shown below. Consider

$$\|\sqrt{C}(f_{\lambda,n} - f_0)\|_{\mathcal{H}} \leq \|\sqrt{C}(f_{\lambda,n} - f_\lambda)\|_{\mathcal{H}} + \tilde{\mathcal{A}}_{\frac{1}{2}}(\lambda) + \mathcal{A}^\star_{\frac{1}{2}}(\lambda)$$

$$\leq \|\sqrt{C}(\hat{C}+\lambda I)^{-1}\| \left(\|(C-\hat{C})(f_\lambda - \tilde{f}_0)\|_{\mathcal{H}} + \|\xi - \hat{\xi}\|_{\mathcal{H}} + \|(C-\hat{C})\tilde{f}_0\|_{\mathcal{H}} \right) + \tilde{\mathcal{A}}_{\frac{1}{2}}(\lambda) + \mathcal{A}^\star_{\frac{1}{2}}(\lambda),$$

where $\tilde{\mathcal{A}}_{\frac{1}{2}}(\lambda) := \|\sqrt{C}(f_\lambda - \tilde{f}_0)\|_{\mathcal{H}}$ and $\mathcal{A}^\star_{\frac{1}{2}}(\lambda) := \|\sqrt{C}(\tilde{f}_0 - f_0)\|_{\mathcal{H}}$. It follows from Theorem 4(i) and Lemma 14 that $\mathcal{A}^\star_{\frac{1}{2}}(\lambda) = J(p_0\|p_{\tilde{f}_0}) = 0$. Also it follows from Proposition A.4(v) that $\|\sqrt{C}(\hat{C}+\lambda I)^{-1}\| \lesssim \frac{1}{\sqrt{\lambda}}$ for $n \geq \frac{c}{\lambda^2}$ where c is a large enough constant that does not depend on n and λ and depends only on $\sum_{i=1}^d \int \|\partial_i k(x,\cdot)\|^4_{\mathcal{H}} p_0(x)\,dx$. Using the bounds from the proof of Theorem 6(i) for the rest of the terms within parenthesis, we obtain

$$\|\sqrt{C}(f_{\lambda,n} - f_0)\|_{\mathcal{H}} \leq O_{p_0}\left(\frac{1}{\sqrt{\lambda n}}\right) + \tilde{\mathcal{A}}_{\frac{1}{2}}(\lambda). \tag{22}$$

The consistency result therefore follows from Proposition A.3(i) by noting that $\tilde{\mathcal{A}}_{\frac{1}{2}}(\lambda) \to 0$ as $\lambda \to 0$. If $f_0 \in \mathcal{R}(C^\beta)$ for some $\beta \geq 0$, then Proposition A.3(ii) yields $\tilde{\mathcal{A}}_{\frac{1}{2}}(\lambda) \leq \max\{1, \|C\|^{\beta-\frac{1}{2}}\}\lambda^{\min\{1,\beta+\frac{1}{2}\}}\|C^{-\beta}f_0\|_{\mathcal{H}}$ which when used in (22) provides the desired rate with $\lambda = n^{-\max\{\frac{1}{3},\frac{1}{2(\beta+1)}\}}$. If $\|C^{-1}\| < \infty$, then the result follows by noting $\|\sqrt{C}(f_{\lambda,n} - f_0)\|_{\mathcal{H}} \leq \|\sqrt{C}\|\|f_{\lambda,n} - f_0\|_{\mathcal{H}}$ and invoking the bound in (21). ■

8.7 Proof of Proposition 8

Observation 1: By (Wendland, 2005, Theorem 10.12), we have

$$\mathcal{H} = \left\{ f \in L^2(\mathbb{R}^d) \cap C_b(\mathbb{R}^d) : \frac{f^\wedge}{\sqrt{\psi^\wedge}} \in L^2(\mathbb{R}^d) \right\},$$

where $f^\wedge$ is defined in L^2 sense. Since

$$\int_{\mathbb{R}^d} |f^\wedge(\omega)|\,d\omega \leq \left(\int_{\mathbb{R}^d} \frac{|f^\wedge(\omega)|^2}{\psi^\wedge(\omega)}\,d\omega \right)^{\frac{1}{2}} \left(\int_{\mathbb{R}^d} \psi^\wedge(\omega)\,d\omega \right)^{\frac{1}{2}} < \infty$$

where we used $\psi^\wedge \in L^1(\mathbb{R}^d)$ (see Wendland, 2005, Corollary 6.12), we have $f^\wedge \in L^1(\mathbb{R}^d)$. Hence Plancherel's theorem and continuity of f along with the inverse Fourier transform of $f^\wedge$ allow to recover any $f \in \mathcal{H}$ pointwise from its Fourier transform as

$$f(x) = \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} e^{ix^T\omega} f^\wedge(\omega)\,d\omega,\ x \in \mathbb{R}^d. \tag{23}$$

Observation 2: Since $\psi^\wedge \in L^1(\mathbb{R}^d)$ and $\psi^\wedge \geq 0$, we have for all $j = 1, \ldots, d$,

$$\left(\int_{\mathbb{R}^d} |\omega_j| \psi^\wedge(\omega)\,d\omega \right)^2 = \left(\int_{\mathbb{R}^d} \psi^\wedge(\omega)\,d\omega \right)^2 \left(\int_{\mathbb{R}^d} |\omega_j| \frac{\psi^\wedge(\omega)}{\int_{\mathbb{R}^d} \psi^\wedge(\omega)\,d\omega}\,d\omega \right)^2$$
$$\overset{(*)}{\leq} \left(\int_{\mathbb{R}^d} \psi^\wedge(\omega)\,d\omega \right) \left(\int_{\mathbb{R}^d} |\omega_j|^2 \psi^\wedge(\omega)\,d\omega \right)$$
$$\leq \left(\int_{\mathbb{R}^d} \psi^\wedge(\omega)\,d\omega \right) \left(\int_{\mathbb{R}^d} \|\omega\|^2 \psi^\wedge(\omega)\,d\omega \right) \overset{(i)}{<} \infty,$$

where we used Jensen's inequality in $(*)$. This means $\omega_j \psi^\wedge(\omega) \in L^1(\mathbb{R}^d)$, $\forall\, j \in [d]$ which ensures the existence of its Fourier transform and so

$$\partial_j \psi(x) = \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} (i\omega_j)\psi^\wedge(\omega) e^{ix^T\omega}\,d\omega,\ x \in \mathbb{R}^d,\ \forall\, j \in [d]. \tag{24}$$

Observation 3: For $g \in \mathcal{H}$, we have for all $j \in [d]$,

$$\int_{\mathbb{R}^d} |\omega_j||g^\wedge(\omega)|\,d\omega \leq \left(\int_{\mathbb{R}^d} \frac{|g^\wedge(\omega)|^2}{\psi^\wedge(\omega)}\,d\omega \right)^{\frac{1}{2}} \left(\int_{\mathbb{R}^d} |\omega_j|^2 \psi^\wedge(\omega)\,d\omega \right)^{\frac{1}{2}}$$
$$\leq \left(\int_{\mathbb{R}^d} \frac{|g^\wedge(\omega)|^2}{\psi^\wedge(\omega)}\,d\omega \right)^{\frac{1}{2}} \left(\int_{\mathbb{R}^d} \|\omega\|^2 \psi^\wedge(\omega)\,d\omega \right)^{\frac{1}{2}} \overset{(i)}{<} \infty,$$

which implies $\omega_j g^\wedge(\omega) \in L^1(\mathbb{R}^d)$, $\forall\, j = 1, \ldots, d$. Therefore,

$$\partial_j g(x) = \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} (i\omega_j) g^\wedge(\omega) e^{ix^T\omega}\,d\omega,\ x \in \mathbb{R}^d,\ \forall\, j \in [d]. \tag{25}$$

Observation 4: For any $g \in \mathcal{G}$, we have

$$\int_{\mathbb{R}^d} \frac{|g^\wedge(\omega)|^2}{\psi^\wedge(\omega)}\,d\omega = \int_{\mathbb{R}^d} \frac{|g^\wedge(\omega)|^2}{\phi^\wedge(\omega)} \frac{\phi^\wedge(\omega)}{\psi^\wedge(\omega)}\,d\omega \leq \|g\|^2_{\mathcal{G}} \left\| \frac{\phi^\wedge}{\psi^\wedge} \right\|_\infty \overset{(ii)}{<} \infty,$$

which implies $g \in \mathcal{H}$, i.e., $\mathcal{G} \subset \mathcal{H}$.

We now use these observations to prove the result. Since $f_0 \in \mathcal{R}(C)$, there exists $g \in \mathcal{H}$ such that $f_0 = Cg$, which means

$$f_0(y) \;=\; \int_{\mathbb{R}^d} \sum_{j=1}^d \partial_j k(x,y)\,\partial_j\, p_0(x)\,dx$$

$$\overset{(24)}{=} \int_{\mathbb{R}^d} \sum_{j=1}^{d} \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} e^{i(x-y)^T\omega} (i\omega_j)\psi^\wedge(\omega)\, d\omega\, \partial_j g(x)\, p_0(x)\, dx$$

$$\overset{(\dagger)}{=} \int_{\mathbb{R}^d} \sum_{j=1}^{d} \left(\frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} e^{ix^T\omega} \partial_j g(x)\, p_0(x)\, dx \right) (i\omega_j)\psi^\wedge(\omega) e^{-iy^T\omega}\, d\omega$$

$$\overset{(25)}{=} \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} \sum_{j=1}^{d} \overline{(i(\cdot)_j g^\wedge * p_0^\wedge)(\omega)} (i\omega_j)\psi^\wedge(\omega) e^{-iy^T\omega}\, d\omega$$

which from (23) means $f_0^\wedge(\omega) = \sum_{j=1}^d \overline{(i(\cdot)_j g^\wedge * p_0^\wedge)(-\omega)}(-i\omega_j)\psi^\wedge(\omega)$ where we have invoked Fubini's theorem in (†) and $*$ represents the convolution. Define $\|\cdot\|_{L^r(\mathbb{R}^d)} := \|\cdot\|_r$ and $\theta := \frac{r}{r-1}$. Consider

$$\begin{aligned}
\|f_0\|_{\mathcal{G}}^2 &= \int_{\mathbb{R}^d} \frac{|f_0^\wedge(\omega)|^2}{\phi^\wedge(\omega)}\, d\omega = \int_{\mathbb{R}^d} \left| \sum_{j=1}^{d} \overline{(i(\cdot)_j g^\wedge * p_0^\wedge)(-\omega)}(i\omega_j) \right|^2 (\psi^\wedge)^2(\omega)(\phi^\wedge(\omega))^{-1}\, d\omega \\
&\le \int_{\mathbb{R}^d} \left(\sum_{j=1}^{d} \left| i(\cdot)_j g^\wedge * p_0^\wedge \right| (-\omega)|\omega_j| \right)^2 (\psi^\wedge)^2(\omega)(\phi^\wedge(\omega))^{-1}\, d\omega \\
&\le \int_{\mathbb{R}^d} \sum_{j=1}^{d} \left| i(\cdot)_j g^\wedge * p_0^\wedge \right|^2 (-\omega)\|\omega\|^2 (\psi^\wedge)^2(\omega)(\phi^\wedge(\omega))^{-1}\, d\omega \\
&\le \left\| \sum_{j=1}^{d} \left| i\omega_j g^\wedge(\omega) * p_0^\wedge(\omega) \right|^2 (\cdot) \right\|_{\frac{\theta}{2}} \left\| \|\cdot\|^2 (\psi^\wedge)^2(\cdot)(\phi^\wedge(\cdot))^{-1} \right\|_{\frac{r}{2-r}} \overset{(iii)}{<} \infty,
\end{aligned}$$

where in the following we show that $\sum_{j=1}^d |i\omega_j g^\wedge(\omega) * p_0^\wedge(\omega)|^2(\cdot) \in L^{\frac{\theta}{2}}(\mathbb{R}^d)$, i.e.,

$$\begin{aligned}
\left\| \sum_{j=1}^{d} \left| i\omega_j g^\wedge(\omega) * p_0^\wedge(\omega) \right|^2 (\cdot) \right\|_{\frac{\theta}{2}} &\le \sum_{j=1}^{d} \left\| \left| i\omega_j g^\wedge(\omega) * p_0^\wedge(\omega) \right|^2 (\cdot) \right\|_{\frac{\theta}{2}} = \sum_{j=1}^{d} \left\| i\omega_j g^\wedge(\omega) * p_0^\wedge(\omega) \right\|_\theta^2 \\
&\overset{(*)}{\le} \textstyle\sum_{j=1}^d \|i\omega_j g^\wedge(\omega)\|_1^2 \, \|p_0^\wedge\|_\theta^2 \overset{(**)}{\le} \|p_0\|_r^2 \sum_{j=1}^d \|i\omega_j g^\wedge(\omega)\|_1^2 \overset{(\ddagger)}{<} \infty,
\end{aligned}$$

where we have invoked generalized Young's inequality (Folland, 1999, Proposition 8.9) in $(*)$, Hausdorff-Young inequality (Folland, 1999, p. 253) in $(**)$, and observation 3 combined with (iv) in $(\ddagger)$. This shows that $f_0 \in \mathcal{R}(C) \Rightarrow f_0 \in \mathcal{G}$, i.e., $\mathcal{R}(C) \subset \mathcal{G}$. ■

8.8 Proof of Theorem 9

To prove Theorem 9, we need the following lemma (De Vito et al., 2012, Lemma 5), which is due to Andreas Maurer.

Lemma 15 *Suppose A and B are self-adjoint Hilbert-Schmidt operators on a separable Hilbert space H with spectrum contained in the interval $[a,b]$, and let $(\sigma_i)_{i\in I}$ and $(\tau_j)_{j\in J}$ be*

the eigenvalues of A and B, respectively. Given a function $r : [a,b] \to \mathbb{R}$, if there exists a finite constant L such that $|r(\sigma_i) - r(\tau_j)| \le L|\sigma_i - \tau_j|$, $\forall\, i \in I$, $j \in J$, then $\|r(A) - r(B)\|_{HS} \le L\|A - B\|_{HS}$.

Proof of Theorem 9. (i) The proof follows the ideas in the proof of Theorem 10 in Bauer et al. (2007), which is a more general result dealing with the smoothness condition, $f_0 \in \mathcal{R}(\Theta(C))$ where Θ is operator monotone. Recall that Θ is operator monotone on $[0,b]$ if for any pair of self-adjoint operators U, V with spectra in $[0,b]$ such that $U \le V$, we have $\Theta(U) \le \Theta(V)$, where "$\le$" is the partial ordering for self-adjoint operators on some Hilbert space H, which means for any $f \in H$, $\langle f, Uf\rangle_H \le \langle f, Vf\rangle_H$. In our case, we adapt the proof for $\Theta(C) = C^\beta$. Define $r_\lambda(\alpha) := g_\lambda(\alpha)\alpha - 1$. Since $f_0 \in \mathcal{R}(C^\beta)$, there exists $h \in \mathcal{H}$ such that $f_0 = C^\beta h$, which yields

$$\begin{aligned}
f_{g,\lambda,n} - f_0 &= -g_\lambda(\hat{C})\hat{\xi} - f_0 = -g_\lambda(\hat{C})(\hat{\xi} + \hat{C}f_0) + r_\lambda(\hat{C})C^\beta h \\
&= -g_\lambda(\hat{C})(\hat{\xi} - \xi) + g_\lambda(\hat{C})(C - \hat{C})f_0 + r_\lambda(\hat{C})\hat{C}^\beta h + r_\lambda(\hat{C})(C^\beta - \hat{C}^\beta)h. \quad (26)
\end{aligned}$$

so that

$$\begin{aligned}
\|f_{g,\lambda,n} - f_0\|_{\mathcal{H}} \le \underbrace{\|g_\lambda(\hat{C})(\hat{\xi} - \xi)\|_{\mathcal{H}}}_{(A)} + \underbrace{\|g_\lambda(\hat{C})(\hat{C} - C)f_0\|_{\mathcal{H}}}_{(B)} + \underbrace{\|r_\lambda(\hat{C})C^\beta h\|_{\mathcal{H}}}_{(C)} \\
+ \underbrace{\|r_\lambda(\hat{C})(C^\beta - \hat{C}^\beta)h\|_{\mathcal{H}}}_{(D)}.
\end{aligned}$$

We now bound (A)–(D). Since $(A) \le \|g_\lambda(\hat{C})\|\|\hat{\xi} - \xi\|_{\mathcal{H}}$, we have $(A) = O_{p_0}\left(\frac{1}{\lambda\sqrt{n}}\right)$ where we used (b) in **(E)** and the bound on $\|\hat{\xi} - \xi\|_{\mathcal{H}}$ from the proof of Theorem 6(i). Similarly, $(B) \le \|g_\lambda(\hat{C})\|\|(\hat{C} - C)f_0\|_{\mathcal{H}}$ implies $(B) = O_{p_0}\left(\frac{1}{\lambda\sqrt{n}}\right)$ where (b) in **(E)** and Proposition A.4(i) are invoked. Also, (d) in **(E)** implies that

$$(C) \le \|r_\lambda(\hat{C})\hat{C}^\beta\|\|h\|_{\mathcal{H}} \le \max\{\gamma_\beta, \gamma_{\eta_0}\}\lambda^{\min\{\beta,\eta_0\}}\|C^{-\beta}f_0\|_{\mathcal{H}}.$$

(D) can be bounded as

$$(D) \le \|r_\lambda(\hat{C})\|\|C^\beta - \hat{C}^\beta\|\|C^{-\beta}f_0\|_{\mathcal{H}}.$$

We now consider two cases:

$\underline{\beta \le 1}$: Since $\alpha \mapsto \alpha^\theta$ is operator monotone on $[0,\chi]$ for $0 \le \theta \le 1$, by Theorem 1 in Bauer et al. (2007), there exists a constant c_θ such that $\|\hat{C}^\theta - C^\theta\| \le c_\theta\|\hat{C} - C\|^\theta \le c_\theta\|\hat{C} - C\|_{HS}^\theta$. We now obtain a bound on $\|\hat{C} - C\|_{HS}$. To this end, consider

$$\begin{aligned}
\mathbb{E}\|\hat{C} - C\|_{HS}^2 &= \mathbb{E}\|\hat{C}\|_{HS}^2 - \|C\|_{HS}^2 \\
&\le \frac{1}{n}\int \left\| \sum_{i=1}^{d} \partial_i k(x,\cdot) \otimes \partial_i k(x,\cdot) \right\|_{HS}^2 p_0(x)\, dx \le \frac{d}{n} \sum_{i=1}^{d} \int \|\partial_i k(x,\cdot)\|^4 p_0(x)\, dx,
\end{aligned}$$

which by Chebyshev's inequality implies that

$$\|\hat{C} - C\|_{HS} = O_{p_0}(n^{-1/2})$$

and therefore $(D) = O_{p_0}(n^{-\beta/2})$. Since $\lambda \geq n^{-1/2}$, we have $(D) = O_{p_0}(\lambda^{\beta})$.

$\underline{\beta > 1}$: Since $\alpha \mapsto \alpha^{\theta}$ is Lipschitz on $[0, \chi]$ for $\theta \geq 1$, by Lemma 15, $\|C^{\beta} - \hat{C}^{\beta}\| \leq \|C^{\beta} - \hat{C}^{\beta}\|_{HS} \leq \beta\chi^{\beta-1}\|C - \hat{C}\|_{HS}$ and therefore $(C) = O_{p_0}(n^{-1/2})$.

Collecting all the above bounds, we obtain

$$\|f_{g,\lambda,n} - f_0\|_{\mathcal{H}} \leq O_{p_0}\left(\frac{1}{\lambda\sqrt{n}}\right) + O_{p_0}\left(\lambda^{\min\{\beta,\eta_0\}}\right)$$

and the result follows. The proofs of the claims involving L^r, h and KL follow exactly the same ideas as in the proof of Theorem 7 by using the above bound on $\|f_{g,\lambda,n} - f_0\|_{\mathcal{H}}$ in Lemma A.1.

(ii) We now bound $J(p_0\|p_{f_{g,\lambda,n}}) = \|\sqrt{C}(f_{g,\lambda,n} - f_0)\|_{\mathcal{H}}^2$ as follows. Note that

$$\sqrt{C}(f_{g,\lambda,n} - f_0) = \underbrace{(\sqrt{C} - \sqrt{\hat{C}})(f_{g,\lambda,n} - f_0)}_{(I')} + \underbrace{\sqrt{\hat{C}}(f_{g,\lambda,n} - f_0)}_{(II')}.$$

We bound $\|(I')\|_{\mathcal{H}}$ as

$$\begin{aligned}\|(I')\|_{\mathcal{H}} \leq \|\sqrt{C} - \sqrt{\hat{C}}\|\|f_{g,\lambda,n} - f_0\|_{\mathcal{H}} &\leq c_{\frac{1}{2}}\sqrt{\|C - \hat{C}\|}_{HS}\|f_{g,\lambda,n} - f_0\|_{\mathcal{H}} \\ &= O_{p_0}\left(\frac{1}{\sqrt{\lambda n}}\right) + O_{p_0}\left(\lambda^{\min\{\beta,\eta_0\}+\frac{1}{2}}\right),\end{aligned}$$

where we used the fact that $\alpha \mapsto \sqrt{\alpha}$ is operator monotone along with $\lambda \geq n^{-1/2}$. Using (26), $\|(II')\|_{\mathcal{H}}$ can be bounded as

$$\begin{aligned}\|(II')\|_{\mathcal{H}} \leq \|\sqrt{\hat{C}}g_\lambda(\hat{C})\|\|\hat{\xi} + \hat{C}f_0\|_{\mathcal{H}} + \|\sqrt{\hat{C}}r_\lambda(\hat{C})\hat{C}^{\beta}\|\|C^{-\beta}f_0\|_{\mathcal{H}} \\ + \|\sqrt{\hat{C}}r_\lambda(\hat{C})\|\|C^{\beta} - \hat{C}^{\beta}\|\|C^{-\beta}f_0\|_{\mathcal{H}}\end{aligned}$$

where

$$\|\sqrt{\hat{C}}g_\lambda(\hat{C})\| \leq \sqrt{\frac{A_g B_g}{\lambda}}, \;\; \|\sqrt{\hat{C}}r_\lambda(\hat{C})\hat{C}^{\beta}\| \leq (\gamma_{\beta+\frac{1}{2}} \vee \gamma_{\eta_0})\lambda^{\min\{\beta+\frac{1}{2},\eta_0\}}$$

and

$$\|\sqrt{\hat{C}}r_\lambda(\hat{C})\| \leq (\gamma_{\frac{1}{2}} \vee \gamma_{\eta_0})\lambda^{\min\{\frac{1}{2},\eta_0\}}$$

with $\|\hat{C}f_0 + \hat{\xi}\|$ and $\|C^{\beta} - \hat{C}^{\beta}\|$ bounded as in part (i) above. Here $(a \vee b) := \max\{a, b\}$. Combining $\|(I')\|_{\mathcal{H}}$ and $\|(II')\|_{\mathcal{H}}$, we obtain the required result.

(iii) The proof follows the ideas in the proof of Theorems 6 and 7. Consider $f_{g,\lambda,n} - f_0 = -g_\lambda(\hat{C})(\hat{C}f_0 + \hat{\xi}) + r_\lambda(\hat{C})f_0$ so that

$$\begin{aligned}\|f_{g,\lambda,n} - f_0\|_{\mathcal{H}} &\leq \|C^{-1}\|\|Cg_\lambda(\hat{C})(\hat{C}f_0 + \hat{\xi})\|_{\mathcal{H}} + \|C^{-1}\|\|Cr_\lambda(\hat{C})f_0\|_{\mathcal{H}} \\ &\leq \|C^{-1}\|\|\hat{C}f_0 + \hat{\xi}\|_{\mathcal{H}}\left(\|\hat{C}g_\lambda(\hat{C})\| + \|\hat{C} - C\|\|g_\lambda(\hat{C})\|\right) \\ &\quad + \|C^{-1}\|\|f_0\|_{\mathcal{H}}\left(\|\hat{C}r_\lambda(\hat{C})\| + \|\hat{C} - C\|\|r_\lambda(\hat{C})\|\right).\end{aligned}$$

Therefore $\|f_{g,\lambda,n} - f_0\|_{\mathcal{H}} = O_{p_0}(n^{-1/2}) + O\left(\lambda^{\min\{1,\eta_0\}}\right)$ where we used the fact that $\lambda \geq n^{-1/2}$ and the result follows. ■

8.9 Proof of Theorem 10

Before we analyze $J(p_0\|p_{f_{\lambda,n}})$, we need a small calculation for notational convenience. For any probability densities $p, q \in C^1$, it is clear that $\sqrt{2J(p\|q)} = \|\|\nabla \log p - \nabla \log q\|_2\|_{L^2(p)}$. We generalize this by defining

$$\sqrt{2J(p\|q\|\mu)} := \|\|\nabla \log p - \nabla \log q\|_2\|_{L^2(\mu)}.$$

Clearly, if $\mu = p$, then $J(p\|q\|\mu)$ matches with $J(p\|q)$. Therefore, for probability densities $p, q, r \in C^1$,

$$\sqrt{J(p\|r\|p)} \leq \sqrt{J(p\|q\|p)} + \sqrt{J(q\|r\|p)}. \tag{27}$$

Based on (27), we have

$$\begin{aligned}\sqrt{\inf_{p\in\mathcal{P}} J(p_0\|p)} &\leq \sqrt{J(p_0\|p_{f_{\lambda,n}}\|p_0)} \leq \sqrt{J(p_0\|p_{f^*}\|p_0)} + \sqrt{J(p_{f^*}\|p_{f_{\lambda,n}}\|p_0)} \\ &= \sqrt{\inf_{p\in\mathcal{P}} J(p_0\|p\|p_0)} + \sqrt{J(p_{f^*}\|p_{f_{\lambda,n}}\|p_0)} \\ &= \sqrt{\inf_{p\in\mathcal{P}} J(p_0\|p)} + \frac{1}{\sqrt{2}}\sqrt{\langle f_{\lambda,n} - f^*, C(f_{\lambda,n} - f^*)\rangle_{\mathcal{H}}} \\ &= \sqrt{\inf_{p\in\mathcal{P}} J(p_0\|p)} + \frac{1}{\sqrt{2}}\|\sqrt{C}(f_{\lambda,n} - f^*)\|_{\mathcal{H}} \\ &= \sqrt{\inf_{p\in\mathcal{P}} J(p_0\|p)} + \frac{1}{\sqrt{2}}\|\sqrt{C}(f_{\lambda,n} - f_\lambda)\|_{\mathcal{H}} + \frac{1}{\sqrt{2}}\mathcal{A}^*(\lambda),\end{aligned} \tag{28}$$

where $\mathcal{A}^*(\lambda) = \|\sqrt{C}(f_\lambda - f^*)\|_{\mathcal{H}}$. The result simply follows from the proof of Theorem 7, where we showed that $\|\sqrt{C}(f_{\lambda,n} - f_\lambda)\|_{\mathcal{H}} = O_{p_0}\left(\frac{1}{\sqrt{\lambda n}}\right)$ and $\mathcal{A}^*(\lambda) = O(\lambda^{\min\{1,\beta+\frac{1}{2}\}})$ if $f^* \in \mathcal{R}(C^{\beta})$ for $\beta \geq 0$ as $\lambda \to 0$, $n \to \infty$. When $\|C^{-1}\| < \infty$, we bound $\|\sqrt{C}(f_{\lambda,n} - f^*)\|_{\mathcal{H}}$ in (28) as $\|\sqrt{C}\|\|f_{\lambda,n} - f^*\|_{\mathcal{H}}$ where $\|f_{\lambda,n} - f^*\|_{\mathcal{H}}$ is in turn bounded as in (21). ■

8.10 Proof of Proposition 11

For $f \in \mathcal{H}$, we have

$$\|f\|_{\mathcal{W}_2}^2 = \int_\Omega \sum_{i=1}^d (\partial_i f)^2 p_0(x)\,dx \leq \|f\|_{\mathcal{H}}^2 \int_\Omega \sum_{i=1}^d \|\partial_i k(x,\cdot)\|_{\mathcal{H}}^2 p_0(x)\,dx < \infty,$$

which means $f \in \mathcal{W}_2(\Omega, p_0)$ and therefore $[f]_\sim \in W_2(\Omega, p_0)$. Since $\|I_k f\|_{W_2} = \|[f]_\sim\|_{\mathcal{W}_2^\sim} = \|f\|_{\mathcal{W}_2} \leq c\|f\|_{\mathcal{H}} < \infty$ where c is some constant, it is clear that I_k is a continuous map from $\mathcal{H}$ to $W_2(\Omega, p_0)$. The adjoint $S_k : W_2(\Omega, p_0) \to \mathcal{H}$ of $I_k : \mathcal{H} \to W_2(\Omega, p_0)$ is defined by the relation $\langle S_k f, g\rangle_{\mathcal{H}} = \langle f, I_k g\rangle_{W_2}$, $f \in W_2(\Omega, p_0)$, $g \in \mathcal{H}$. If $f := [h]_\sim \in \mathcal{W}_2^\sim(\Omega, p_0)$, then

$$\langle [h]_\sim, I_k g\rangle_{W_2} = \langle [h]_\sim, [g]_\sim\rangle_{\mathcal{W}_2^\sim} = \sum_{|\alpha|=1} \int_\Omega (\partial^\alpha h)(x)(\partial^\alpha g)(x)\, p_0(x)\,dx.$$

For $y \in \Omega$ and $g = k(\cdot, y)$, this yields

$$S_k[h]_\sim(y) = \langle S_k[h]_\sim, k(\cdot,y)\rangle_{\mathcal{H}} = \langle [h]_\sim, I_k k(\cdot,y)\rangle_{W_2} = \int_\Omega \sum_{i=1}^d \partial_i k(x,y)\partial_i h(x) p_0(x)\,dx.$$

We now show that I_k is Hilbert-Schmidt. Since $\mathcal{H}$ is separable, let $(e_l)_{l\geq 1}$ be an ONB of $\mathcal{H}$. Then we have

$$\begin{aligned}\sum_l \|I_k e_l\|_{W_2}^2 = \sum_l \int_\Omega \sum_{i=1}^d (\partial_i e_l(x))^2 p_0(x)\,dx &= \int_\Omega \sum_{i=1}^d \sum_l \langle e_l, \partial_i k(x,\cdot)\rangle_{\mathcal{H}}^2 p_0(x)\,dx \\ &= \int_\Omega \sum_{i=1}^d \|\partial_i k(x,\cdot)\|_{\mathcal{H}}^2 p_0(x)\,dx < \infty,\end{aligned}$$

which proves that I_k is Hilbert-Schmidt (hence compact) and therefore S_k is also Hilbert-Schmidt and compact. The other assertions about $S_k I_k$ and $I_k S_k$ are straightforward. ■

8.11 Proof of Theorem 12

By slight abuse of notation, $f_\star$ is used to denote $[f_\star]_\sim$ in the proof for simplicity. For $f \in \mathcal{F}$, we have

$$J(p_0\|p_f) = \frac{1}{2}\|I_k f - f_\star\|_{W_2}^2 = \frac{1}{2}\langle E_k f, f\rangle_{\mathcal{H}} - \langle S_k f_\star, f\rangle_{\mathcal{H}} + \frac{1}{2}\|f_\star\|_{W_2}^2.$$

Since k satisfies **(C)** it is easy to verify that $\langle S_k f_\star, f\rangle_{\mathcal{H}} = \langle f, -\xi\rangle_{\mathcal{H}}$, $\forall\, f \in \mathcal{H}$ (see proof of Theorem 4(ii)). This implies $S_k f_\star = -\xi$ and

$$J(p_0\|p_f) = \frac{1}{2}\langle E_k f, f\rangle_{\mathcal{H}} + \langle f, \xi\rangle_{\mathcal{H}} + \frac{1}{2}\|f_\star\|_{W_2}^2, \tag{29}$$

where ξ is defined in Theorem 4(ii), and E_k is precisely the operator C defined in Theorem 4(ii). Following the proof of Theorem 4(ii), for $\lambda > 0$, it is easy to show that the unique minimizer of the regularized objective, $J(p_0\|p_f) + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2$ exists and is given by

$$f_\lambda = -(E_k + \lambda I)^{-1}\xi = (E_k + \lambda I)^{-1} S_k f_\star. \tag{30}$$

We would like to reiterate that (29) and (30) also match with their counterparts in Theorem 4 and therefore as in Theorem 4(iv), an estimator of $f_\star$ is given by $f_{\lambda,n} = -(\hat{E}_k + \lambda I)^{-1}\hat{\xi}$. In other words, this is the same as in Theorem 4(iv) since $\hat{E}_k = \hat{C}$, and can be solved by a simple linear system provided in Theorem 5. Here $\hat{E}_k$ is the empirical estimator of E_k. Now consider

$$\begin{aligned}\sqrt{2\,J(p_0\|p_{f_{\lambda,n}})} = \|I_k f_{\lambda,n} - f_\star\|_{W_2} &\leq \|I_k(f_{\lambda,n} - f_\lambda)\|_{W_2} + \|I_k f_\lambda - f_\star\|_{W_2} \\ &= \|\sqrt{E_k}(f_{\lambda,n} - f_\lambda)\|_{\mathcal{H}} + \mathcal{B}(\lambda),\end{aligned} \tag{31}$$

where $\mathcal{B}(\lambda) := \|I_k f_\lambda - f_\star\|_{W_2}$. The proof now proceeds using the following decomposition, equivalent to the one used in the proof of Theorem 6(i), i.e.,

$$f_{\lambda,n} - f_\lambda = -(\hat{E}_k + \lambda I)^{-1}\hat{\xi} - f_\lambda$$

$$\begin{aligned}&= -(\hat{E}_k + \lambda I)^{-1}(\hat{\xi} + \hat{E}_k f_\lambda + \lambda f_\lambda) \\ &\overset{(\dagger)}{=} -(\hat{E}_k + \lambda I)^{-1}(\hat{\xi} + \hat{E}_k f_\lambda + S_k f_\star - E_k f_\lambda - \hat{S}_k f_\star + \hat{S}_k f_\star),\end{aligned}$$

where we used (30) in (†). $\hat{S}_k f_\star$ is well-defined as it is the empirical version of the restriction of S_k to $\mathcal{W}_2^\sim(p_0)$. Since $S_k f_\star - E_k f_\lambda = S_k(f_\star - I_k f_\lambda)$ and $\hat{S}_k f_\star - \hat{E}_k f_\lambda = \hat{S}_k(f_\star - I_k f_\lambda)$, we have

$$f_{\lambda,n} - f_\lambda = -(\hat{E}_k + \lambda I)^{-1}(\hat{\xi} + \hat{S}_k f_\star) + (\hat{E}_k + \lambda I)^{-1}(\hat{S}_k - S_k)(f_\star - I_k f_\lambda)$$

and so

$$\|\sqrt{E_k}(f_{\lambda,n} - f_\lambda)\|_{\mathcal{H}} \leq \|\sqrt{E_k}(\hat{E}_k + \lambda I)^{-1}\| \left(\|\hat{\xi} + \hat{S}_k f_\star\|_{\mathcal{H}} + \|(\hat{S}_k - S_k)(f_\star - I_k f_\lambda)\|_{\mathcal{H}}\right). \tag{32}$$

It follows from Proposition A.4(v) that

$$\|\sqrt{E_k}(\hat{E}_k + \lambda I)^{-1}\| \lesssim \frac{1}{\sqrt{\lambda}} \tag{33}$$

for $n \geq \frac{c}{\lambda^2}$ where c is a sufficiently large constant that does not depend on n and λ. Following the proof of Proposition A.4(i), we have

$$\mathbb{E}\|\hat{\xi} + \hat{S}_k f_\star\|_{\mathcal{H}}^2 = \frac{n-1}{n}\|\xi + S_k f_\star\|_{\mathcal{H}}^2 + \frac{1}{n}\int_\Omega \left\|\sum_{i=1}^d \partial_i k(x,\cdot)\partial_i f_\star + \xi_x\right\|_{\mathcal{H}}^2 p_0(x)\,dx$$

wherein the first term is zero as $S_k f_\star + \xi = 0$ and since

$$\left\|\sum_{i=1}^d \partial_i k(x,\cdot)\partial_i f_\star + \xi_x\right\|_{\mathcal{H}}^2 \leq 2\|\xi_x\|_{\mathcal{H}}^2 + 2\chi\|\nabla f_\star\|_2^2,$$

the integral in the second term is finite because of **(D)** and $f^* \in W_2(\Omega, p_0)$. Therefore, an application of Chebyshev's inequality yields

$$\|\hat{\xi} + \hat{S}_k f_\star\|_{\mathcal{H}} = O_{p_0}(n^{-1/2}). \tag{34}$$

We now show that $\|(\hat{S}_k - S_k)(f_\star - I_k f_\lambda)\|_{\mathcal{H}} = O_{p_0}(\mathcal{B}(\lambda)n^{-1/2})$. To this end, define $g := f_\star - I_k f_\lambda$ and consider

$$\mathbb{E}_{p_0}\|\hat{S}_k g - S_k g\|_{\mathcal{H}}^2 = \frac{\int_\Omega \|\sum_{i=1}^d \partial_i k(x,\cdot)\partial_i g(x)\|_{\mathcal{H}}^2 p_0(x)\,dx - \|S_k g\|_{\mathcal{H}}^2}{n} \leq \frac{\chi}{n}\|g\|_{W_2}^2,$$

which therefore yields the claim through an application of Chebyshev's inequality. Using this along with (33) and (34) in (32), and using the resulting bound in (31) yields

$$\sqrt{2\,J(p_0\|p_{f_{\lambda,n}})} \leq O_{p_0}\left(\frac{1}{\sqrt{\lambda n}} + \frac{\mathcal{B}(\lambda)}{\sqrt{\lambda n}}\right) + \mathcal{B}(\lambda). \tag{35}$$

(i) We bound $\mathcal{B}(\lambda)$ as follows. First note that

$$\mathcal{B}(\lambda) = \|I_k(S_k I_k + \lambda I)^{-1} S_k f_\star - f_\star\|_{W_2} = \|(T_k + \lambda I)^{-1} T_k f_\star - f_\star\|_{W_2}$$

and so for any $h \in \mathcal{H}$, we have

$$\mathcal{B}(\lambda) = \|(T_k + \lambda I)^{-1} T_k f_\star - f_\star\|_{W_2} \leq \underbrace{\|((T_k + \lambda I)^{-1} T_k - I)(f_\star - I_k h)\|_{W_2}}_{(I)} + \underbrace{\|(T_k + \lambda I)^{-1} T_k I_k h - I_k h\|_{W_2}}_{(II)}. \quad (36)$$

Since T_k is a self-adjoint compact operator, there exists $(\alpha_l)_{l\in\mathbb{N}}$ and ONB $(\phi_l)_{l\in\mathbb{N}}$ of $\overline{\mathcal{R}(T_k)}$ so that $T_k = \sum_l \alpha_l \langle \phi_l, \cdot \rangle_{W_2} \phi_l$. Let $(\psi_j)_{j\in\mathbb{N}}$ be the orthonormal basis of $\mathcal{N}(T_k)$. Then we have

$$(I)^2 = \sum_l \left(\frac{\alpha_l}{\alpha_l + \lambda} - 1 \right)^2 \langle f_\star - I_k h, \phi_l \rangle_{W_2}^2 + \sum_j \langle f_\star - I_k h, \psi_j \rangle_{W_2}^2 \leq \sum_l \langle f_\star - I_k h, \phi_l \rangle_{W_2}^2 + \sum_j \langle f_\star - I_k h, \psi_j \rangle_{W_2}^2 = \|f_\star - I_k h\|_{W_2}^2. \quad (37)$$

From $(T_k + \lambda I)^{-1} T_k = I_k (E_k + \lambda I)^{-1} S_k$ and $S_k I_k h = E_k h$, we have

$$(II) = \|I_k (E_k + \lambda I)^{-1} E_k h - I_k h\|_{W_2} = \|\sqrt{E_k}(E_k + \lambda I)^{-1} E_k h - \sqrt{E_k} h\|_{\mathcal{H}} \leq \|h\|_{\mathcal{H}} \sqrt{\lambda}, \quad (38)$$

where the inequality follows from Proposition A.3(ii). Using (37) and (38) in (36), we obtain $\mathcal{B}(\lambda) \leq \|f_\star - I_k h\|_{W_2} + \|h\|_{\mathcal{H}} \sqrt{\lambda}$, using which in (35) yields

$$\sqrt{2\, J(p_0 \| p_{f_{\lambda,n}})} \leq \|f_\star - I_k h\|_{W_2} + O_{p_0}\left(\frac{1}{\sqrt{\lambda n}} \right) + \|h\|_{\mathcal{H}} \sqrt{\lambda}.$$

Since the above inequality holds for any $h \in \mathcal{H}$, we therefore have

$$\sqrt{2\, J(p_0 \| p_{f_{\lambda,n}})} \leq \inf_{h\in\mathcal{H}} \left(\|f_\star - I_k h\|_{W_2} + \sqrt{\lambda} \|h\|_{\mathcal{H}} \right) + O_{p_0}\left(\frac{1}{\sqrt{\lambda n}} \right) = K(f_\star, \sqrt{\lambda}, W_2(p_0), I_k(\mathcal{H})) + O_{p_0}\left(\frac{1}{\sqrt{\lambda n}} \right) \quad (39)$$

where the K-functional is defined in (A.6). Note that $I_k(\mathcal{H}) \cong \mathcal{H}/\mathcal{H} \cap \mathbb{R}$ is continuously embedded in $W(p_0)$. From (A.6), it is clear that the K-functional as a function of t is an infimum over a family of affine linear and increasing functions and therefore is concave, continuous and increasing w.r.t. t. This means, in (39), as $\lambda \to 0$,

$$K(f_\star, \sqrt{\lambda}, W_2(p_0), I_k(\mathcal{H})) \to \inf_{h\in\mathcal{H}} \|f_\star - I_k h\|_{W_2} = \sqrt{2 \inf_{p\in\mathcal{P}} J(p_0 \| p)}.$$

Since $J(p_0 \| p_{f_{\lambda,n}}) \geq \inf_{p\in\mathcal{P}} J(p_0 \| p)$, we have that $J(p_0 \| p_{f_{\lambda,n}}) \to \inf_{p\in\mathcal{P}} J(p_0 \| p)$ as $\lambda \to 0$, $\lambda n \to \infty$ and $n \to \infty$.

(ii) Recall $\mathcal{B}(\lambda)$ from (i). From Proposition A.3(i) it follows that $\mathcal{B}(\lambda) \to 0$ as $\lambda \to 0$ if $f_\star \in \overline{\mathcal{R}(T_k)}$. Therefore, (35) reduces to $\sqrt{2\, J(p_0 \| p_{f_{\lambda,n}})} \leq O_{p_0}\left(\frac{1}{\sqrt{\lambda n}} \right) + \mathcal{B}(\lambda)$ and the consistency result follows. If $f_\star \in \mathcal{R}(T_k^\beta)$ for some $\beta > 0$, then the rates follow from Proposition A.3 by noting that $\mathcal{B}(\lambda) \leq \max\{1, \|T_k\|^{\beta-1}\} \lambda^{\min\{1,\beta\}} \|T_k^{-\beta} f_\star\|_{W_2}$ and choosing $\lambda = n^{-\max\left\{\frac{1}{3}, \frac{1}{2\beta+1}\right\}}$.

(iii) This simply follows from an analysis similar to the one used in the proof of Theorem 6(iii). ■

8.12 Proof of Proposition 13

For any $p \in \mathcal{P}_{\text{FD}}$, define $f := \log \frac{p}{q_0}$, which implies that $[f]_\sim \in W_2(p)$. Since $I_k(\mathcal{H})$ is dense in $W_2(p)$, we have for any $\epsilon > 0$, there exists $g \in \mathcal{H}$ such that $\|[f]_\sim - I_k g\|_{W_2} \leq \sqrt{2\epsilon}$. For a given $g \in \mathcal{H}$, pick $p_g \in \mathcal{P}$. Therefore,

$$J(p \| p_g) = \frac{1}{2} \int_\Omega p(x) \|\nabla \log p - \nabla \log p_g\|_2^2 \, dx = \frac{1}{2} \|[f]_\sim - I_k g\|_{W_2}^2 \leq \epsilon$$

and the result follows. ■

Acknowledgments

We thank the action editor and two anonymous reviewers for their detailed comments that significantly improved the paper. Theorem A.2 and the proof are due to an anonymous reviewer. We also thank Dougal Sutherland for a careful reading of the paper which helped to fix minor errors. A part of the work was carried out while BKS was a Research Fellow in the Statistical Laboratory, Department of Pure Mathematics and Mathematical Statistics, University of Cambridge. BKS thanks Richard Nickl for many valuable comments and insightful discussions. KF is supported in part by MEXT Grant-in-Aid for Scientific Research on Innovative Areas 25120012.

A. Appendix: Technical Results

In this appendix, we present some technical results that are used in the proofs.

A.1 Bounds on Various Distances Between p_f and p_g

In the following result, claims (iii) and (iv) are quoted from Lemma 3.1 of van der Vaart and van Zanten (2008).

Lemma A.1 *Define $\mathcal{P}_\infty := \left\{ p_f = e^{f - A(f)} q_0 : f \in \ell^\infty(\Omega) \right\}$, where q_0 is a probability density on $\Omega \subseteq \mathbb{R}^d$ and $\ell^\infty(\Omega)$ is the space of bounded measurable functions on Ω. Then for any $p_f, p_g \in \mathcal{P}_\infty$, we have*

(i) $\|p_f - p_g\|_{L^r(\Omega)} \leq 2 e^{2\|f-g\|_\infty} e^{2\min\{\|f\|_\infty, \|g\|_\infty\}} \|f - g\|_\infty \|q_0\|_{L^r(\Omega)}$ *for any* $1 \leq r \leq \infty$;

(ii) $\|p_f - p_g\|_{L^1(\Omega)} \leq 2 e^{\|f-g\|_\infty} \|f - g\|_\infty$;

(iii) $KL(p_f\|p_g) \le c\,\|f-g\|_\infty^2 e^{\|f-g\|_\infty}\,(1+\|f-g\|_\infty)$ *where* c *is a universal constant;*

(iv) $h(p_f,p_g) \le e^{\|f-g\|_\infty/2}\|f-g\|_\infty$.

Proof *(i)* Define $B(f) := \int e^f q_0\,dx$. Consider

$$\begin{aligned}\|p_f-p_g\|_{L^r(\Omega)} &= \left\|\frac{e^f q_0}{B(f)}-\frac{e^g q_0}{B(g)}\right\|_{L^r(\Omega)} = \frac{\left\|e^f q_0 B(g)-e^g q_0 B(f)\right\|_{L^r(\Omega)}}{B(f)B(g)}\\ &= \frac{\left\|e^f q_0\,(B(g)-B(f))+\left(e^f-e^g\right)q_0 B(f)\right\|_{L^r(\Omega)}}{B(f)B(g)}\\ &\le \frac{\left\|e^f q_0\,(B(g)-B(f))\right\|_{L^r(\Omega)}}{B(f)B(g)}+\frac{\left\|\left(e^f-e^g\right)q_0 B(f)\right\|_{L^r(\Omega)}}{B(f)B(g)}\\ &\le \frac{|B(g)-B(f)|\,\|e^f q_0\|_{L^r(\Omega)}}{B(g)B(f)}+\frac{\left\|(e^f-e^g)q_0\right\|_{L^r(\Omega)}}{B(g)}. \end{aligned} \tag{A.1}$$

Observe that

$$|B(f)-B(g)| \le \int_\Omega |e^f-e^g|q_0\,dx = \int_\Omega e^g|e^{f-g}-1|q_0\,dx \le e^{\|f-g\|_\infty}\|f-g\|_\infty B(g)$$

since $|e^{u-v}-1| \le |u-v|e^{|u-v|}$ for any $u,v\in\mathbb{R}$. Similarly,

$$\left\|(e^f-e^g)q_0\right\|_{L^r(\Omega)} \le e^{\|f-g\|_\infty}\|f-g\|_\infty\|e^g q_0\|_{L^r(\Omega)}.$$

Using these above, we obtain

$$\|p_f-p_g\|_{L^r(\Omega)} \le e^{\|f-g\|_\infty}\|f-g\|_\infty\left(\frac{\|e^f q_0\|_{L^r(\Omega)}}{B(f)}+\frac{\|e^g q_0\|_{L^r(\Omega)}}{B(g)}\right). \tag{A.2}$$

Since $\|e^f q_0\|_{L^r(\Omega)} \le e^{\|f\|_\infty}\|q_0\|_{L^r(\Omega)}$ and $B(f)\ge e^{-\|f\|_\infty}$, from (A.2) we obtain

$$\begin{aligned}\|p_f-p_g\|_{L^r(\Omega)} &\le e^{\|f-g\|_\infty}\|f-g\|_\infty\|q_0\|_{L^r(\Omega)}\left(e^{2\|f\|_\infty}+e^{2\|g\|_\infty}\right).\\ &\le 2e^{\|f-g\|_\infty}\|f-g\|_\infty\|q_0\|_{L^r(\Omega)}e^{2\max\{\|f\|_\infty,\|g\|_\infty\}}\\ &\le 2e^{2\|f-g\|_\infty}\|f-g\|_\infty\|q_0\|_{L^r(\Omega)}e^{2\min\{\|f\|_\infty,\|g\|_\infty\}}\end{aligned}$$

where we used $\max\{a,b\}\le\min\{a,b\}+|a-b|$ for $a,b\ge 0$ in the last line above.
(ii) This simply follows from (A.2) by using $r=1$. ■

A.2 General Representer Theorem

The following is the general representer theorem for abstract Hilbert spaces.

Theorem A.2 (General representer theorem) *Let* H *be a real Hilbert space and let* $(\phi_i)_{i=1}^m \in H^m$. *Suppose* $J: H\to\mathbb{R}$ *be such that* $J(f) = V\left(\langle f,\phi_1\rangle_H,\ldots,\langle f,\phi_m\rangle_H\right)$, $f\in H$ *where* $V:\mathbb{R}^n\to\mathbb{R}$ *is a convex differentiable function. Define*

$$f_\lambda = \arg\inf_{f\in H} J(f)+\frac{\lambda}{2}\|f\|_H^2,$$

where $\lambda>0$. *Then there exists* $(\alpha_i)_{i=1}^m\in\mathbb{R}^m$ *such that* $f_\lambda=\sum_{i=1}^m\alpha_i\phi_i$ *where* $\boldsymbol{\alpha} := (\alpha_1,\ldots,\alpha_m)$ *satisfies the following (possibly nonlinear) equation*

$$\lambda\boldsymbol{\alpha}+\nabla V\left(\boldsymbol{K}\boldsymbol{\alpha}\right)=0,$$

with $\boldsymbol{K}$ *being a linear map on* $\mathbb{R}^m$ *and* $(\boldsymbol{K})_{i,j}=\langle\phi_i,\phi_j\rangle_H$, $i\in[m]$, $j\in[m]$.

Proof Define $A: H\to\mathbb{R}^m$, $f\mapsto(\langle f,\phi_i\rangle_H)_{i=1}^m$. Then $f_\lambda = \arg\inf_{f\in H} V(Af)+\frac{\lambda}{2}\|f\|_H^2$. Therefore, Fermat's rule yields

$$\begin{aligned}0=A^*\nabla V(Af_\lambda)+\lambda f_\lambda &\Leftrightarrow f_\lambda = A^*\left(-\frac{1}{\lambda}\nabla V(Af_\lambda)\right)\\ &\Leftrightarrow (\exists\,\boldsymbol{\alpha}\in\mathbb{R}^m)\quad f_\lambda=A^*\boldsymbol{\alpha},\ \boldsymbol{\alpha}=-\frac{1}{\lambda}\nabla V(Af_\lambda)\\ &\Leftrightarrow (\exists\,\boldsymbol{\alpha}\in\mathbb{R}^m)\quad f_\lambda=A^*\boldsymbol{\alpha},\ \boldsymbol{\alpha}=-\frac{1}{\lambda}\nabla V(AA^*\boldsymbol{\alpha}),\end{aligned}$$

where $A^*:\mathbb{R}^m\to H$ is the adjoint of A which can be obtained as follows. Note that

$$\langle Af,\boldsymbol{\alpha}\rangle = \sum_{i=1}^m\alpha_i\langle f,\phi_i\rangle_H = \left\langle f,\sum_{i=1}^m\alpha_i\phi_i\right\rangle_H \qquad (\forall\, f\in H)\ (\forall\,\boldsymbol{\alpha}\in\mathbb{R}^m)$$

and thus $A^*\boldsymbol{\alpha}=\sum_{i=1}^m\alpha_i\phi_i$. Therefore $AA^*\boldsymbol{\alpha}=\sum_{j=1}^m\alpha_j A\phi_j=\sum_{j=1}^m\alpha_j(\langle\phi_j,\phi_i\rangle_H)_{i=1}^m$, and so for every $i\in[m]$, $(AA^*\boldsymbol{\alpha})_i=\sum_{j=1}^m\langle\phi_j,\phi_i\rangle_H\alpha_j$ and hence $AA^*=\boldsymbol{K}$. ■

A.3 Bounds on Approximation Errors, $\mathcal{A}_0(\lambda)$ and $\mathcal{A}_{\frac{1}{2}}(\lambda)$

The following result is quite well-known in the linear inverse problem theory (Engl et al., 1996).

Proposition A.3 *Let* C *be a bounded, self-adjoint compact operator on a separable Hilbert space* H. *For* $\lambda>0$ *and* $f\in H$, *define* $f_\lambda := (C+\lambda I)^{-1}Cf$ *and* $\mathcal{A}_\theta(\lambda) := \|C^\theta(f_\lambda-f)\|_H$ *for* $\theta\ge 0$. *Then the following hold.*

(i) For any $\theta>0$, $\mathcal{A}_\theta(\lambda)\to 0$ *as* $\lambda\to 0$ *and if* $f\in\overline{\mathcal{R}(C)}$, *then* $\mathcal{A}_0(\lambda)\to 0$ *as* $\lambda\to 0$.

(ii) If $f\in\mathcal{R}(C^\beta)$ *for* $\beta\ge 0$ *and* $\beta+\theta>0$, *then*

$$\mathcal{A}_\theta(\lambda)\le\max\{1,\|C\|^{\beta+\theta-1}\}\lambda^{\min\{1,\beta+\theta\}}\|C^{-\beta}f\|_H.$$

Proof (i) Since C is bounded, compact, and self-adjoint, the Hilbert-Schmidt theorem (Reed and Simon, 1972, Theorems VI.16, VI.17) ensures that $C = \sum_l \alpha_l \phi_l \langle \phi_l, \cdot \rangle_H$, where $(\alpha_l)_{l\in\mathbb{N}}$ are the positive eigenvalues and $(\phi_l)_{l\in\mathbb{N}}$ are the corresponding unit eigenvectors that form an ONB for $\mathcal{R}(C)$. Let $\theta = 0$. Since $f \in \overline{\mathcal{R}(C)}$,

$$\mathcal{A}_0^2(\lambda) = \left\|(C+\lambda I)^{-1}Cf - f\right\|_H^2 = \left\|\sum_i \frac{\alpha_i}{\alpha_i+\lambda}\langle f,\phi_i\rangle_H \phi_i - \sum_i \langle f,\phi_i\rangle_H \phi_i\right\|_H^2$$
$$= \left\|\sum_i \frac{\lambda}{\alpha_i+\lambda}\langle f,\phi_i\rangle_H \phi_i\right\|_H^2 = \sum_i \left(\frac{\lambda}{\alpha_i+\lambda}\right)^2 \langle f,\phi_i\rangle_H^2 \to 0 \text{ as } \lambda \to 0$$

by the dominated convergence theorem. For any $\theta > 0$, we have

$$\mathcal{A}_\theta^2(\lambda) = \left\|C^\theta(C+\lambda I)^{-1}Cf - C^\theta f\right\|_H^2.$$

Let $f = f_R + f_N$ where $f_R \in \overline{\mathcal{R}(C^\theta)}$, $f_N \in \overline{\mathcal{R}(C^\theta)}^\perp$ if $0 < \theta \le 1$ and $f_R \in \overline{\mathcal{R}(C)}$, $f_N \in \overline{\mathcal{R}(C)}^\perp$ if $\theta \ge 1$. Then

$$\mathcal{A}_\theta^2(\lambda) = \left\|C^\theta(C+\lambda I)^{-1}Cf - C^\theta f\right\|_H^2 = \left\|C^\theta(C+\lambda I)^{-1}Cf_R - C^\theta f_R\right\|_H^2$$
$$= \left\|\sum_i \frac{\alpha_i^{1+\theta}}{\alpha_i+\lambda}\langle f_R,\phi_i\rangle_H \phi_i - \sum_i \alpha_i^\theta \langle f_R,\phi_i\rangle_H \phi_i\right\|_H^2$$
$$= \left\|\sum_i \frac{\lambda\alpha_i^\theta}{\alpha_i+\lambda}\langle f_R,\phi_i\rangle_H \phi_i\right\|_H^2 = \sum_i \left(\frac{\lambda\alpha_i^\theta}{\alpha_i+\lambda}\right)^2 \langle f_R,\phi_i\rangle_H^2 \to 0$$

as $\lambda \to 0$.

(ii) If $f \in \mathcal{R}(C^\beta)$, then there exists $g \in H$ such that $f = C^\beta g$. This yields

$$\mathcal{A}_\theta^2(\lambda) = \left\|C^\theta(C+\lambda I)^{-1}Cf - C^\theta f\right\|_H^2 = \left\|C^\theta(C+\lambda I)^{-1}C^{\beta+1}g - C^{\theta+\beta}g\right\|_H^2$$
$$= \left\|\sum_i \frac{\lambda\alpha_i^{\theta+\beta}}{\alpha_i+\lambda}\langle g,\phi_i\rangle_H \phi_i\right\|_H^2 = \sum_i \left(\frac{\lambda\alpha_i^{\theta+\beta}}{\alpha_i+\lambda}\right)^2 \langle g,\phi_i\rangle_H^2. \tag{A.3}$$

Suppose $0 < \beta + \theta < 1$. Then

$$\frac{\alpha_i^{\beta+\theta}\lambda}{\alpha_i+\lambda} = \left(\frac{\alpha_i}{\alpha_i+\lambda}\right)^{\beta+\theta}\left(\frac{\lambda}{\alpha_i+\lambda}\right)^{1-\theta-\beta}\lambda^{\beta+\theta} \le \lambda^{\beta+\theta}.$$

On the other hand, for $\beta + \theta \ge 1$, we have

$$\frac{\alpha_i^{\beta+\theta}\lambda}{\alpha_i+\lambda} = \left(\frac{\alpha_i}{\alpha_i+\lambda}\right)\alpha_i^{\beta+\theta-1}\lambda \le \|C\|^{\beta+\theta-1}\lambda.$$

Using the above in (A.3) yields the result. ■

A.4 Bound on the Norm of Certain Operators and Functions

The following result is used in many places throughout the paper. We would like to highlight that special cases of this result are known, e.g., see the proof of Theorem 4 in Caponnetto and Vito (2007) where concentration inequalities are obtained for the quantities in Proposition A.4 using Bernstein's inequality. Here, we provide asymptotic statements using Chebyshev's inequality.

Proposition A.4 *Let $\mathcal{X}$ be a topological space, H be a separable Hilbert space and $\mathcal{L}_2^+(H)$ be the space of positive, self-adjoint Hilbert-Schmidt operators on H. Define $R := \int_{\mathcal{X}} r(x)\, d\mathbb{P}(x)$ and $\hat{R} := \frac{1}{n}\sum_{a=1}^m r(X_a)$ where $\mathbb{P} \in M_+^1(\mathcal{X})$, $(X_a)_{a=1}^m \overset{i.i.d.}{\sim} \mathbb{P}$ and r is a $\mathcal{L}_2^+(H)$-valued measurable function on $\mathcal{X}$ satisfying $\int_{\mathcal{X}} \|r(x)\|_{HS}^2\, d\mathbb{P}(x) < \infty$. Define $g_\lambda := (R+\lambda I)^{-1}Rg$ for $g \in H$, $\lambda > 0$, and $\mathcal{A}_0(\lambda) := \|g_\lambda - g\|_H$. Let $\alpha \ge 0$ and $\theta > 0$. Then the following hold:*

(i) $\|(\hat{R}-R)(g_\lambda - g)\|_H = O_{\mathbb{P}}\left(\frac{\mathcal{A}_0(\lambda)}{\sqrt{m}}\right)$.

(ii) $\|R^\alpha(R+\lambda I)^{-\theta}\| \le \lambda^{\alpha-\theta}$.

(iii) $\|\hat{R}^\alpha(\hat{R}+\lambda I)^{-\theta}\| \le \lambda^{\alpha-\theta}$.

(iv) $\|(R+\lambda I)^{-\theta}(\hat{R}-R)\| = O_{\mathbb{P}}\left(\sqrt{\frac{1}{m\lambda^{2\theta}}}\right)$.

(v) $\|R^\alpha(\hat{R}+\lambda I)^{-1}\| \lesssim \lambda^{\alpha-1}$ *for* $m \ge \frac{c}{\lambda^2}$ *where is c is a sufficiently large constant that depends on $\int \|r(x)\|_{HS}^2\, d\mathbb{P}(x)$ but not on m and λ.*

Proof (i) Note that for any $f \in H$,

$$\mathbb{E}_{\mathbb{P}}\|(\hat{R}-R)f\|_H^2 = \mathbb{E}_{\mathbb{P}}\|\hat{R}f\|_H^2 + \|Rf\|_H^2 - 2\mathbb{E}_{\mathbb{P}}\langle \hat{R}f, Rf\rangle_H,$$

where $\mathbb{E}_{\mathbb{P}}\langle \hat{R}f, Rf\rangle_H = \frac{1}{n}\sum_{a=1}^n \mathbb{E}_{\mathbb{P}}\langle r(X_a)f, Rf\rangle_H = \frac{1}{n}\sum_{a=1}^n \mathbb{E}_{\mathbb{P}}\langle r(X_a), f\otimes Rf\rangle_{HS}$. Since $\int_{\mathcal{X}} \|r(x)\|_{HS}^2\, d\mathbb{P}(x) < \infty$, $r(x)$ is $\mathbb{P}$-integrable in the Bochner sense (see Diestel and Uhl, 1977, Definition 1 and Theorem 2), and therefore it follows from Diestel and Uhl (1977, Theorem 6) that $\mathbb{E}_{\mathbb{P}}\langle r(X_a), f\otimes Rf\rangle_{HS} = \langle \int_{\mathcal{X}} r(x)\, d\mathbb{P}(x), f\otimes Rf\rangle_{HS} = \|Rf\|_H^2$. Therefore,

$$\mathbb{E}_{\mathbb{P}}\|(\hat{R}-R)f\|_H^2 = \mathbb{E}_{\mathbb{P}}\|\hat{R}f\|_H^2 - \|Rf\|_H^2,$$

where

$$\mathbb{E}_{\mathbb{P}}\|\hat{R}f\|_H^2 = \mathbb{E}_{\mathbb{P}}\left\|\frac{1}{m}\sum_{a=1}^m r(X_a)f\right\|_H^2 = \frac{1}{m^2}\sum_{a,b=1}^m \mathbb{E}_{\mathbb{P}}\langle r(X_a)f, r(X_b)f\rangle_H.$$

Splitting the sum into two parts (one with $a = b$ and the other with $a \ne b$), it is easy to verify that $\mathbb{E}_{\mathbb{P}}\|\hat{R}f\|_H^2 = \frac{1}{m}\int_{\mathcal{X}} \|r(x)f\|_H^2\, d\mathbb{P}(x) + \frac{m-1}{m}\|Rf\|_H^2$, thereby yielding

$$\mathbb{E}_{\mathbb{P}}\|(\hat{R}-R)f\|_H^2 = \frac{1}{m}\left(\int_{\mathcal{X}} \|r(x)f\|_H^2\, d\mathbb{P}(x) - \|Rf\|_H^2\right) \le \frac{1}{m}\int_{\mathcal{X}} \|r(x)f\|_H^2\, d\mathbb{P}(x)$$
$$\le \frac{\|f\|_H^2}{m}\int_{\mathcal{X}} \|r(x)\|_H^2\, d\mathbb{P}(x).$$

Using $f = g_\lambda - g$, an application of Chebyshev's inequality yields the result.

(ii, iii) $\|R^\alpha(R+\lambda I)^{-\theta}\| = \sup_i \frac{\gamma_i^\alpha}{(\gamma_i+\lambda)^\theta} = \sup_i \left[\left(\frac{\gamma_i}{\gamma_i+\lambda}\right)^\alpha \frac{1}{(\gamma_i+\lambda)^{\theta-\alpha}}\right] \leq \sup_i \frac{1}{(\gamma_i+\lambda)^{\theta-\alpha}} \leq \lambda^{\alpha-\theta}$, where $(\gamma_i)_{i\in\mathbb{N}}$ are the eigenvalues of R. (iii) follows by replacing $(\gamma_i)_{i\in\mathbb{N}}$ with the eigenvalues of $\hat{R}$.

(iv) Since $\|(R+\lambda I)^{-\theta}(\hat{R}-R)\| \leq \|(R+\lambda I)^{-\theta}(\hat{R}-R)\|_{HS}$, consider $\mathbb{E}_{\mathbb{P}}\|(R+\lambda I)^{-\theta}(\hat{R}-R)\|_{HS}^2$, which using the technique in the proof of (i), can be shown to be bounded as

$$\mathbb{E}_{\mathbb{P}}\|(R+\lambda I)^{-\theta}(\hat{R}-R)\|_{HS}^2 \leq \frac{1}{m}\int_{\mathcal{X}} \|(R+\lambda I)^{-\theta} r(x)\|_{HS}^2 \, d\mathbb{P}(x). \tag{A.4}$$

Note that

$$\begin{aligned}\|(R+\lambda I)^{-\theta} r(x)\|_{HS}^2 &= \langle (R+\lambda I)^{-\theta} r(x), (R+\lambda I)^{-\theta} r(x)\rangle_{HS} \\ &= \|(R+\lambda I)^{-2\theta}\|\mathrm{Tr}\,(r(x)r(x)) = \|(R+\lambda I)^{-2\theta}\|\|r(x)\|_{HS}^2 \\ &\leq \lambda^{-2\theta}\|r(x)\|_{HS}^2, \end{aligned} \tag{A.5}$$

where the last inequality follows from (iii). Using (A.5) in (A.4), we obtain

$$\mathbb{E}_{\mathbb{P}}\|(R+\lambda I)^{-\theta}(\hat{R}-R)\|_{HS}^2 \leq \frac{1}{m\lambda^{2\theta}}\int_{\mathcal{X}} \|r(x)\|_{HS}^2 \, d\mathbb{P}(x).$$

The result therefore follows by an application of Chebyshev's inequality.

(v) We use the idea in Step 2.1 of the proof of Theorem 4 in Caponnetto and Vito (2007), where $R^\alpha(\hat{R}+\lambda I)^{-1}$ is written equivalently as follows: Note that $\hat{R}+\lambda I = (\hat{R}-R)+(R+\lambda I)$, which implies

$$(\hat{R}+\lambda I)^{-1} = \left((\hat{R}-R)+(R+\lambda I)\right)^{-1} = (R+\lambda I)^{-1}\left(I-(R-\hat{R})(R+\lambda I)^{-1}\right)^{-1}$$

and so $R^\alpha(\hat{R}+\lambda I)^{-1} = R^\alpha(R+\lambda I)^{-1}\left(I-(R-\hat{R})(R+\lambda I)^{-1}\right)^{-1}$. Using the Von Neumann series representation, we have

$$R^\alpha(\hat{R}+\lambda I)^{-1} = R^\alpha(R+\lambda I)^{-1}\sum_{j=0}^{\infty}\left((R-\hat{R})(R+\lambda I)^{-1}\right)^j$$

so that

$$\begin{aligned}\|R^\alpha(\hat{R}+\lambda I)^{-1}\| &\leq \|R^\alpha(R+\lambda I)^{-1}\|\sum_{j=0}^{\infty}\|(R-\hat{R})(R+\lambda I)^{-1}\|_{HS}^j \\ &\leq \lambda^{\alpha-1}\sum_{j=0}^{\infty}\|(R-\hat{R})(R+\lambda I)^{-1}\|_{HS}^j.\end{aligned}$$

From the proof of (iv), we have that for any $\delta > 0$, with probability at least $1-\delta$, $\|(R-\hat{R})(R+\lambda I)^{-1}\|_{HS} \leq \sqrt{\frac{\int_{\mathcal{X}}\|r(x)\|_{HS}^2 d\mathbb{P}(x)}{m\lambda^2\delta}}$. Suppose $m \geq \frac{\int_{\mathcal{X}}\|r(x)\|_{HS}^2 d\mathbb{P}(x)}{s^2\lambda^2\delta}$ where $s<1$. Then $\sum_{j=0}^{\infty}\|(R-\hat{R})(R+\lambda I)^{-1}\|_{HS}^j \leq \sum_{j=0}^{\infty} s^j = \frac{1}{1-s}$. This means for $m \geq \frac{c}{\lambda^2}$ where c is sufficiently large, we obtain $\|R^\alpha(\hat{R}+\lambda I)^{-1}\| \lesssim \lambda^{\alpha-1}$. ■

A.5 Interpolation Space

In this section, we briefly recall the definition of interpolation spaces of the real method. To this end, let E_0 and E_1 be two arbitrary Banach spaces that are continuously embedded in some topological (Hausdorff) vector space $\mathcal{E}$. Then, for $x \in E_0+E_1 := \{x_0+x_1 : x_0 \in E_0,\, x_1 \in E_1\}$ and $t>0$, the *K-functional* of the real interpolation method (see Bennett and Sharpley, 1988, Definition 1.1, p. 293) is defined by

$$K(x,t,E_0,E_1) := \inf\{\|x_0\|_{E_0} + t\|x_1\|_{E_1} \,:\, x_0 \in E_0,\, x_1 \in E_1,\, x = x_0+x_1\}.$$

Suppose E and F are two Banach spaces that satisfy $F \hookrightarrow E$ (i.e., $F \subset E$ and the inclusion operator $\mathrm{id} : F \to E$ is continuous), then the K-functional reduces to

$$K(x,t,E,F) = \inf_{y\in F}\|x-y\|_E + t\|y\|_F. \tag{A.6}$$

The K-functional can be used to define interpolation norms, for $0<\theta<1$, $1 \leq s \leq \infty$ and $x \in E_0+E_1$, as

$$\|x\|_{\theta,s} := \begin{cases} \left(\int \left(t^{-\theta}K(x,t)\right)^s t^{-1}\,dt\right)^{1/s}, & 1 \leq s < \infty \\ \sup_{t>0} t^{-\theta}K(x,t), & s=\infty. \end{cases}$$

Moreover, the corresponding interpolation spaces (Bennett and Sharpley, 1988, Definition 1.7, p. 299) are defined as

$$[E_0,E_1]_{\theta,s} := \{x \in E_0+E_1 \,:\, \|x\|_{\theta,s} < \infty\}.$$

B. Appendix: Miscellaneous Results

In this appendix, we present the proofs of some claims that we made in Sections 1, 4 and 5.

B.1 Relation between Fisher and Kullback-Leibler Divergences

The following result provides a relationship between Fisher and Kullback-Leibler divergences.

Proposition B.1 *Let p and q be probability densities defined on $\mathbb{R}^d$. Define $p_t := p * N(0,tI_d)$ and $q_t := q * N(0,tI_d)$ where $N(0,tI_d)$ denotes a normal distribution on $\mathbb{R}^d$ with mean zero and diagonal covariance with $t>0$. Suppose p_t and q_t satisfy*

$$\partial_i p_t(x)\log p_t(x) = o\left(\|x\|_2^\alpha\right),\ \partial_i p_t(x)\log q_t(x) = o\left(\|x\|_2^\alpha\right) \text{ and } \partial_i \log q_t(x) p_t(x) = o\left(\|x\|_2^\alpha\right)$$

as $\|x\|_2 \to \infty$ for all $i \in [d]$ where $\alpha = 1-d$. Then

$$KL(p\|q) = \int_0^\infty J(p_t\|q_t)\,dt, \tag{B.1}$$

where J is defined in (3).

Proof Under the conditions mentioned on p_t and q_t, it can be shown that

$$\frac{d}{dt}KL(p_t\|q_t) = -J(p_t\|q_t). \tag{B.2}$$

See Theorem 1 in Lyu (2009) for a proof. The above identity is a simple generalization of de Bruijn's identity that relates the Fisher information to the derivative of the Shannon entropy (see Cover and Thomas, 1991, Theorem 16.6.2). Integrating w.r.t. t on both sides of (B.2), we obtain $KL(p_t\|q_t)\Big|_{t=0}^{\infty} = -\int_0^\infty J(p_t\|q_t)\,dt$ which yields the equality in (B.1) as $KL(p_t\|q_t) \to 0$ as $t \to \infty$ and $KL(p_t\|q_t) \to KL(p\|q)$ as $t \to 0$. ■

B.2 Estimation of p_0: Unbounded k

To handle the case of unbounded k, in the following, we assume that there exists a positive constant M such that $\|f_0\|_{\mathcal{H}} \le M$, so that an estimator of f_0 can be constructed as

$$\breve{f}_{\lambda,n} = \arg\inf_{f\in\mathcal{H}} \hat{J}_\lambda(f) \;\text{ subject to }\; \|f\|_{\mathcal{H}} \le M, \tag{B.3}$$

where $\hat{J}_\lambda$ is defined in Theorem 4(iv). This modification yields a valid estimator $p_{\breve{f}_{\lambda,n}}$ as long as k satisfies $\int_\Omega e^{M\sqrt{k(x,x)}}q_0(x)\,dx < \infty$, since this implies $\breve{f}_{\lambda,n} \in \mathcal{F}$. The construction of $\breve{f}_{\lambda,n}$ requires the knowledge of M, however, which we assume is known *a priori*. Using the representer theorem in RKHS, it can be shown (see Section B.2.1) that

$$\breve{f}_{\lambda,n} = \breve{\delta}\hat{\xi} + \sum_{b=1}^{n}\sum_{j=1}^{d}\breve{\beta}_{(b-1)d+j}\partial_j k(X_b,\cdot)$$

where $\breve{\delta}$ and $\breve{\boldsymbol{\beta}}$ are obtained by solving the following quadratically constrained quadratic program (QCQP),

$$(\breve{\boldsymbol{\beta}},\breve{\delta}) =: \breve{\Theta} = \arg\min_{\Theta\in\mathbb{R}^{nd+1}} \frac{1}{2}\Theta^T\boldsymbol{H}\Theta + \Theta^T\Delta \;\text{ subject to }\; \Theta^T\boldsymbol{K}\Theta \le M^2,$$

with $\Delta := (\boldsymbol{h}, \|\hat{\xi}\|^2_{\mathcal{H}})$, $\Theta := (\boldsymbol{\beta},\delta)$ and $\boldsymbol{K}$, $\boldsymbol{H}$ being defined in the proof of Theorem 5 and the remark following it. The following result investigates the consistency and convergence rates for $p_{\breve{f}_{\lambda,n}}$.

Theorem B.2 (Consistency and rates for $p_{\breve{f}_{\lambda,n}}$) *Let $M \ge \|f_0\|_{\mathcal{H}}$ be a fixed constant, and $\breve{f}_{n,\lambda}$ be a clipped estimator given by (B.3). Suppose* **(A)**–**(D)** *with $\varepsilon = 2$ hold. Let $\operatorname{supp}(q_0) = \Omega$ and $\int_\Omega e^{M\sqrt{k(x,x)}}q_0(x)\,dx < \infty$. Define $\eta(x) = \sqrt{k(x,x)}e^{M\sqrt{k(x,x)}}$. Then, as $\lambda\sqrt{n} \to \infty$, $\lambda \to 0$ and $n \to \infty$,*

(i) $\|p_{\breve{f}_{\lambda,n}} - p_0\|_{L^1(\Omega)} \to 0$, $KL(p_0\|p_{\breve{f}_{\lambda,n}}) \to 0$ *if* $\eta \in L^1(\Omega, q_0)$;

(ii) for $1 < r \le \infty$, $\|p_{\breve{f}_{\lambda,n}} - p_0\|_{L^r(\Omega)} \to 0$ *if* $\eta q_0 \in L^1(\Omega)\cap L^r(\Omega)$ *and* $e^{M\sqrt{k(\cdot,\cdot)}}q_0 \in L^r(\Omega)$;

(iii) $h(p_{\breve{f}_{\lambda,n}}, p_0) \to 0$ *if* $\sqrt{k(\cdot,\cdot)}\eta \in L^1(\Omega, q_0)$;

(iv) $J(p_0\|p_{\breve{f}_{\lambda,n}}) \to 0$.

In addition, if $f_0 \in \mathcal{R}(C^\beta)$ for some $\beta > 0$, then $\|p_{\breve{f}_{\lambda,n}} - p_0\|_{L^r(\Omega)} = O_{p_0}(\theta_n)$, $h(p_0, p_{\breve{f}_{\lambda,n}}) = O_{p_0}(\theta_n)$, $KL(p_0\|p_{\breve{f}_{\lambda,n}}) = O_{p_0}(\theta_n)$ and $J(p_0\|p_{\breve{f}_{\lambda,n}}) = O_{p_0}(\theta_n^2)$ where $\theta_n := n^{-\min\left\{\frac{1}{4},\frac{\beta}{2(\beta+1)}\right\}}$ with $\lambda = n^{-\max\left\{\frac{1}{4},\frac{1}{2(\beta+1)}\right\}}$ assuming the respective conditions in (i)-(iii) above hold.

Proof For any $x \in \Omega$, since $|f_0(x)| \le \|f_0\|_{\mathcal{H}}\sqrt{k(x,x)} \le M\sqrt{k(x,x)}$ and $|\breve{f}_{\lambda,n}(x)| \le M\sqrt{k(x,x)}$, we have

$$\left|e^{\breve{f}_{\lambda,n}(x)} - e^{f_0(x)}\right| \le e^{M\sqrt{k(x,x)}}\left|\breve{f}_{\lambda,n}(x) - f_0(x)\right| \le \eta(x)\left\|\breve{f}_{\lambda,n} - f_0\right\|_{\mathcal{H}}, \tag{B.4}$$

where we used the fact that $|e^x - e^y| \le e^a|x-y|$ for $x, y \in [-a,a]$ and $\eta(x) := \sqrt{k(x,x)}e^{M\sqrt{k(x,x)}}$. In the following, we obtain bounds for $\left\|p_{\breve{f}_{\lambda,n}} - p_0\right\|_{L^r(\Omega)}$ for any $1 \le r \le \infty$, $h(p_{\breve{f}_{\lambda,n}}, p_0)$ and $KL(p_0\|p_{\breve{f}_{\lambda,n}})$ in terms of $\left\|\breve{f}_{\lambda,n} - f_0\right\|_{\mathcal{H}}$. Define $B(f) := \int_\Omega e^f q_0\,dx$. Since k satisfies $\int_\Omega e^{M\sqrt{k(x,x)}}q_0(x)\,dx < \infty$, then it is clear that $\breve{f}_{\lambda,n} \in \mathcal{F}$ as $B(\breve{f}_{\lambda,n}) < \infty$ since

$$\int_\Omega e^{\breve{f}_{\lambda,n}(x)}q_0(x)\,dx \le \int_\Omega e^{\|\breve{f}_{\lambda,n}\|_{\mathcal{H}}\sqrt{k(x,x)}}q_0(x)\,dx \le \int_\Omega e^{M\sqrt{k(x,x)}}q_0(x)\,dx < \infty.$$

Similarly, it is easy to verify that $B(f_0) < \infty$.

(i) Recalling (A.1), we have

$$\left\|p_{\breve{f}_{\lambda,n}} - p_0\right\|_{L^r(\Omega)} \le \frac{|B(\breve{f}_{\lambda,n}) - B(f_0)|\|e^{f_0}q_0\|_{L^r(\Omega)}}{B(\breve{f}_{\lambda,n})B(f_0)} + \frac{\|(e^{f_0} - e^{\breve{f}_{\lambda,n}})q_0\|_{L^r(\Omega)}}{B(\breve{f}_{\lambda,n})}.$$

If $r = 1$, we obtain

$$\left\|p_{\breve{f}_{\lambda,n}} - p_0\right\|_{L^1(\Omega)} \le \frac{|B(\breve{f}_{\lambda,n}) - B(f_0)|}{B(\breve{f}_{\lambda,n})} + \frac{\|(e^{f_0} - e^{\breve{f}_{\lambda,n}})q_0\|_{L^1(\Omega)}}{B(\breve{f}_{\lambda,n})}.$$

Using (B.4), we bound $|B(\breve{f}_{\lambda,n}) - B(f_0)|$ as

$$|B(\breve{f}_{\lambda,n}) - B(f_0)| \le \int_\Omega \left|e^{\breve{f}_{\lambda,n}(x)} - e^{f_0(x)}\right|q_0(x)\,dx \le \|\eta\|_{L^1(\Omega,q_0)}\left\|\breve{f}_{\lambda,n} - f_0\right\|_{\mathcal{H}}.$$

Also for any $f \in \mathcal{H}$ with $\|f\|_{\mathcal{H}} \le M$, we have $B(f) \ge \int_\Omega e^{-M\sqrt{k(x,x)}}q_0(x)\,dx =: \theta$, where $\theta > 0$. Again using (B.4), we have

$$\|(e^{f_0} - e^{\breve{f}_{\lambda,n}})q_0\|_{L^r(\Omega)} \le \|\eta q_0\|_{L^r(\Omega)}\|\breve{f}_{\lambda,n} - f_0\|_{\mathcal{H}}$$

and $\|e^{f_0}q_0\|_{L^r(\Omega)} \le \|e^{M\sqrt{k(x,x)}}q_0\|_{L^r(\Omega)}$. Therefore,

$$\left\|p_{\breve{f}_{\lambda,n}} - p_0\right\|_{L^r(\Omega)} \le \frac{\|\eta\|_{L^1(\Omega,q_0)}\|e^{M\sqrt{k(x,x)}}q_0\|_{L^r(\Omega)}\left\|\breve{f}_{\lambda,n} - f_0\right\|_{\mathcal{H}}}{\theta^2}$$

$$+\frac{\|\eta q_0\|_{L^r(\Omega)}\|\breve{f}_{\lambda,n}-f_0\|_{\mathcal{H}}}{\theta}$$

and for $r=1$,

$$\left\|p_{\breve{f}_{\lambda,n}}-p_0\right\|_{L^1(\Omega)}\leq\frac{2\,\|\eta\|_{L^1(\Omega,q_0)}\left\|\breve{f}_{\lambda,n}-f_0\right\|_{\mathcal{H}}}{\theta}.$$

(ii) Also

$$\begin{aligned}KL(p_0\|p_{\breve{f}_{\lambda,n}})&=\int_\Omega p_0\log\frac{p_0}{p_{\breve{f}_{\lambda,n}}}\,dx=\int_\Omega\log\left(e^{f_0-\breve{f}_{\lambda,n}}\frac{B(\breve{f}_{\lambda,n})}{B(f_0)}\right)p_0(x)\,dx\\&=\int_\Omega\left(f_0-\breve{f}_{\lambda,n}+\log\frac{B(\breve{f}_{\lambda,n})}{B(f_0)}\right)p_0(x)\,dx\\&\leq\frac{|B(\breve{f}_{\lambda,n})-B(f_0)|}{B(f_0)}+\|\breve{f}_{\lambda,n}-f_0\|_{L^1(\Omega,p_0)}\leq\frac{2\,\|\eta q_0\|_{L^1(\Omega)}}{\theta}\|\breve{f}_{\lambda,n}-f_0\|_{\mathcal{H}}.\end{aligned}$$

(iii) It is easy to verify that

$$\begin{aligned}h(p_{\breve{f}_{\lambda,n}},p_0)&=\left\|\frac{e^{\breve{f}_{\lambda,n}/2}}{\|e^{\breve{f}_{\lambda,n}/2}\|_{L^2(\Omega,q_0)}}-\frac{e^{f_0/2}}{\left\|e^{f_0/2}\right\|_{L^2(\Omega,q_0)}}\right\|_{L^2(\Omega,q_0)}\\&\leq\frac{2\|e^{\breve{f}_{\lambda,n}/2}-e^{f_0/2}\|_{L^2(\Omega,q_0)}}{\left\|e^{f_0/2}\right\|_{L^2(\Omega,q_0)}}\end{aligned}$$

where the above inequality is obtained by carrying out and simplifying the decomposition as in (A.1). Using (B.4), we therefore have

$$h(p_{\breve{f}_{\lambda,n}},p_0)\leq\sqrt{\frac{\int_\Omega k(x,x)e^{M\sqrt{k(x,x)}}q_0\,dx}{\theta}}\|\breve{f}_{\lambda,n}-f_0\|_{\mathcal{H}}.$$

(iv) As $f_0,\,\breve{f}_{\lambda,n}\in\mathcal{F}$, by Theorem 4, we obtain $J(p_0\|p_{\breve{f}_{\lambda,n}})=\frac{1}{2}\|\sqrt{C}(\breve{f}_{\lambda,n}-f_0)\|_{\mathcal{H}}^2\leq\frac{1}{2}\|\sqrt{C}\|^2\|\breve{f}_{\lambda,n}-f_0\|_{\mathcal{H}}^2$.

Note that we have bounded the various distances between $p_{\breve{f}_{\lambda,n}}$ and p_0 in terms of $\|\breve{f}_{\lambda,n}-f_0\|_{\mathcal{H}}$. Since $\breve{f}_{\lambda,n}=f_{\lambda,n}$ with probability converging to 1, the assertions on consistency are proved by Theorem 6(i) in combination with Lemma 14—as we did not explicitly assume $f_0\in\overline{\mathcal{R}(C)}$—and the rates follow from Theorem 6(iii). ■

Remark The following observations can be made while comparing the scenarios of using bounded vs. unbounded kernels in the problem of estimating p_0 through Theorems 7 and B.2. First, the consistency results in L^r, Hellinger and KL distances are the same but for additional integrability conditions on k and q_0. The additional integrability conditions are not too difficult to hold in practice as they involve k and q_0 which can be chosen appropriately. However, the unbounded situation in Theorem B.2 requires the knowledge of M which is usually not known. On the other hand, the consistency result in J in Theorem B.2 is slightly weaker than in Theorem 7. This may be an artifact of our analysis as

we are not able to adapt the bounding technique used in the proof of Theorem 7 to bound $J(p_0\|p_{\breve{f}_{\lambda,n}})=\frac{1}{2}\|\sqrt{C}(\breve{f}_{\lambda,n}-f_0)\|_{\mathcal{H}}^2$ as it critically depends on the boundedness of k. Therefore, we used a trivial bound of $J(p_0\|p_{\breve{f}_{\lambda,n}})=\frac{1}{2}\|\sqrt{C}(\breve{f}_{\lambda,n}-f_0)\|_{\mathcal{H}}^2\leq\frac{1}{2}\|\sqrt{C}\|^2\|\breve{f}_{\lambda,n}-f_0\|_{\mathcal{H}}^2$, which yields the result through Theorem 6(i). Due to the same reason, we also obtain a slower rate of convergence in J. Second, the rate of convergence in KL is slower than in Theorem B.2, which again may be an artifact of our analysis. The convergence rate for KL in Theorem 7 is based on the application of Theorem 6(ii) in Lemma A.1, where the bound on KL in Lemma A.1 critically uses the boundedness to upper bound KL in terms of squared Hellinger distance.

B.2.1 Derivation of $\breve{f}_{\lambda,n}$

Any $f\in\mathcal{H}$ can be decomposed as $f=f_\parallel+f_\perp$ where

$$f_\parallel\in\operatorname{span}\left\{\hat{\xi},(\partial_j k(X_b,\cdot))_{b,j}\right\}=:\mathcal{H}_\parallel,$$

which is a closed subset of $\mathcal{H}$ and $f_\perp\in\mathcal{H}_\parallel^\perp:=\left\{g\in\mathcal{H}:\langle g,h\rangle_{\mathcal{H}}=0,\,\forall\,h\in\mathcal{H}_\parallel\right\}$ so that $\mathcal{H}=\mathcal{H}_\parallel\oplus\mathcal{H}_\parallel^\perp$. Since the objective function in (B.3) matches with the one in Theorem 5, using the above decomposition in (B.3), it is easy to verify that $\hat{J}$ depends only on $f_\parallel\in\mathcal{H}_\parallel$ so that (B.3) reduces to

$$(\breve{f}_{\lambda,n}^\parallel,\breve{f}_{\lambda,n}^\perp)=\arg\inf_{\substack{f_\parallel\in\mathcal{H}_\parallel,f_\perp\in\mathcal{H}^\perp\\\|f_\parallel\|_{\mathcal{H}}^2+\|f_\perp\|_{\mathcal{H}}^2\leq M^2}}\hat{J}_\lambda(f_\parallel)+\frac{\lambda}{2}\|f_\parallel\|_{\mathcal{H}}^2+\frac{\lambda}{2}\|f_\perp\|_{\mathcal{H}}^2\tag{B.5}$$

and $\breve{f}_{\lambda,n}=\breve{f}_{\lambda,n}^\parallel+\breve{f}_{\lambda,n}^\perp$. Since $f_\parallel$ is of the form in (14), using it in (B.5), it is easy to show that $\hat{J}_\lambda(f_\parallel)+\frac{\lambda}{2}\|f_\parallel\|_{\mathcal{H}}^2=\frac{1}{2}\Theta^T\boldsymbol{H}\Theta+\Theta^T\Delta$. Similarly, it can be shown that $\|f_\parallel\|_{\mathcal{H}}^2=\Theta^T\boldsymbol{K}\Theta$. Since $f_\perp$ appears in (B.5) only through $\|f_\perp\|_{\mathcal{H}}^2$, (B.5) reduces to

$$(\Theta_\parallel,c_\perp)=\arg\inf_{\substack{\Theta\in\mathbb{R}^{nd+1},c\geq0\\\Theta^T\boldsymbol{K}\Theta+c\leq M^2}}\frac{1}{2}\Theta^T\boldsymbol{H}\Theta+\Theta^T\Delta+\frac{\lambda}{2}c,\tag{B.6}$$

where $\breve{f}_{\lambda,n}^\parallel$ is constructed as in (14) using $\Theta_\parallel$ and $\breve{f}_{\lambda,n}^\perp$ is such that $\|\breve{f}_{\lambda,n}^\perp\|_{\mathcal{H}}^2=c_\perp$. The necessary and sufficient conditions for the optimality of $(\Theta_\parallel,c_\perp)$ is given by the following Karush-Kuhn-Tucker conditions,

$$\begin{aligned}(\boldsymbol{H}+2\tau\boldsymbol{K})\Theta_\parallel+\Delta=0,\ \frac{\lambda}{2}+\eta-\tau=0&\qquad\text{(Stationarity)}\\\Theta_\parallel^T\boldsymbol{K}\Theta_\parallel+c_\perp\leq M^2,\ c_\perp\geq0&\qquad\text{(Primal feasibility)}\\\eta\geq0,\ \tau\geq0&\qquad\text{(Dual feasibility)}\\\tau c_\perp=0,\ \eta(\Theta_\parallel^T\boldsymbol{K}\Theta_\parallel+c_\perp-M^2)=0&\qquad\text{(Complementary slackness)}\end{aligned}$$

Combining the dual feasibility and stationary conditions, we have $\eta=\tau-\frac{\lambda}{2}\geq0$, i.e., $\tau\geq\frac{\lambda}{2}$. Using this in the complementary slackness involving τ and $c_\perp$, it follows that $c_\perp=0$. Since $\|\breve{f}_{\lambda,n}^\perp\|^2=c_\perp$, we have $\breve{f}_{\lambda,n}^\perp=0$, i.e., $\breve{f}_{\lambda,n}$ is completely determined by $\breve{f}_{\lambda,n}^\parallel$. Therefore $\breve{f}_{\lambda,n}^\parallel$ is of the form in (14) and (B.6) reduces to a quadratically constrained quadratic program.

B.3 $\mathcal{R}(C^\beta)$ and Interpolation Spaces

Proposition B.3 presents an interpretation for $\mathcal{R}(C^\beta)$ ($\beta > 0$ and $\beta \notin \mathbb{N}$) as interpolation spaces between $\mathcal{R}(C^{\lceil\beta\rceil})$ and $\mathcal{R}(C^{\lfloor\beta\rfloor})$ where $\mathcal{R}(C^0) := \mathcal{H}$. An inspection of its proof shows that Proposition B.3 holds for any self-adjoint, bounded, compact operator defined on a separable Hilbert space.

Proposition B.3 *Suppose* **(B)** *and* **(D)** *hold with* $\varepsilon = 1$. *Then for all* $\beta > 0$ *and* $\beta \notin \mathbb{N}$

$$\mathcal{R}(C^\beta) = \left[\mathcal{R}(C^{\lfloor\beta\rfloor}), \mathcal{R}(C^{\lceil\beta\rceil})\right]_{\beta-\lfloor\beta\rfloor,2}$$

where $\mathcal{R}(C^0) := \mathcal{H}$, *and the spaces* $\mathcal{R}(C^\beta)$ *and* $\left[\mathcal{R}(C^{\lfloor\beta\rfloor}), \mathcal{R}(C^{\lceil\beta\rceil})\right]_{\beta-\lfloor\beta\rfloor,2}$ *have equivalent norms.*

To prove Proposition B.3, we need the following result which we quote from Steinwart and Scovel (2012, Lemma 6.3) (also see Tartar, 2007, Lemma 23.1) that interpolates L^2-spaces whose underlying measures are absolutely continuous with respect to a measure ν.

Lemma B.4 *Let* ν *be a measure on a measurable space* Θ *and* $w_0 : \Theta \to [0,\infty)$ *and* $w_1 : \Theta \to [0,\infty)$ *be measurable functions. For* $0 < \beta < 1$, *define* $w_\beta := w_0^{1-\beta} w_1^\beta$. *Then we have*

$$[L^2(w_0\, d\nu), L^2(w_1\, d\nu)]_{\beta,2} = L^2(w_\beta\, d\nu)$$

and the norms on these two spaces are equivalent. Moreover, this result still holds for weights $w_0 : \Theta \to (0,\infty)$ *and* $w_1 : \Theta \to [0,\infty]$, *if one uses the convention* $0 \cdot \infty := 0$ *in the definition of the weighted spaces.*

Proof of Proposition B.3. The proof is based on the ideas used in the proof of Theorem 4.6 in Steinwart and Scovel (2012). Recall that by the Hilbert-Schmidt theorem, C has the following representation,

$$C = \sum_{i\in I} \alpha_i \phi_i \langle \phi_i, \cdot\rangle_{\mathcal{H}}$$

where $(\alpha_i)_{i\in I}$ are the positive eigenvalues of C, $(\phi_i)_{i\in I}$ are the corresponding unit eigenvectors that form an ONB for $\mathcal{R}(C)$ and I is an index set which is either finite (if $\mathcal{H}$ is finite-dimensional) or $I = \mathbb{N}$ with $\lim_{i\to\infty} \alpha_i = 0$ (if $\mathcal{H}$ is infinite dimensional). Let $(\psi_i)_{i\in J}$ be an ONB for $\mathcal{N}(C)$ where J is some index set so that any $f \in \mathcal{H}$ can be written as

$$f = \sum_{i\in I} \langle f, \phi_i\rangle_{\mathcal{H}} \phi_i + \sum_{i\in J} \langle f, \psi_i\rangle_{\mathcal{H}} \psi_i =: \sum_{i\in I\cup J} a_i \theta_i$$

where $\theta_i := \phi_i$ if $i \in I$ and $\theta_i := \psi_i$ if $i \in J$ with $a_i := \langle f, \theta_i\rangle_{\mathcal{H}}$. Let $\beta > 0$. By definition, $g \in \mathcal{R}(C^\beta)$ is equivalent to $\exists h \in \mathcal{H}$ such that $g = C^\beta h$, i.e.,

$$g = \sum_{i\in I} \alpha_i^\beta \langle h, \phi_i\rangle_{\mathcal{H}} \phi_i =: \sum_{i\in I} b_i \alpha_i^\beta \phi_i$$

where $b_i := \langle h, \phi_i\rangle_{\mathcal{H}}$. Clearly $\sum_{i\in I} b_i^2 = \sum_{i\in I} \langle h, \phi_i\rangle_{\mathcal{H}}^2 \le \|h\|_{\mathcal{H}}^2 < \infty$, i.e., $(b_i) \in \ell_2(I)$. Therefore

$$\mathcal{R}(C^\beta) = \left\{ \sum_{i\in I} b_i \alpha_i^\beta \phi_i \,:\, (b_i) \in \ell_2(I) \right\} = \left\{ \sum_{i\in I} c_i \phi_i \,:\, (c_i) \in \ell_2(I, \alpha^{-2\beta}) \right\}$$

where $\alpha := (\alpha_i)_{i\in I}$. Let us equip this space with the bilinear form

$$\left\langle \sum_{i\in I} c_i\phi_i, \sum_{i\in I} d_i \phi_i \right\rangle_{\mathcal{R}(C^\beta)} := \langle (c_i), (d_i)\rangle_{\ell_2(I,\alpha^{-2\beta})}$$

so that it induces the norm

$$\left\| \sum_{i\in I} c_i \phi_i \right\|_{\mathcal{R}(C^\beta)} := \|(c_i)\|_{\ell_2(I,\alpha^{-2\beta})}\,.$$

It is easy to verify that $(\alpha_i^\beta \phi_i)_{i\in I}$ is an ONB of $\mathcal{R}(C^\beta)$. Also since $\mathcal{R}(C^{\beta_1}) \subset \mathcal{R}(C^{\beta_2})$ for $0 < \beta_2 < \beta_1 < \infty$ and $\mathrm{id} : \mathcal{R}(C^{\beta_1}) \to \mathcal{R}(C^{\beta_2})$ is continuous, i.e., for any $g \in \mathcal{R}(C^{\beta_1})$,

$$\begin{aligned}\|g\|_{\mathcal{R}(C^{\beta_2})} = \|(c_i)\|_{\ell_2(I,\alpha^{-2\beta_2})} = \sqrt{\sum_{i\in I} \frac{c_i^2}{\alpha_i^{2\beta_2}}} &\le \sup_{i\in I} |\alpha_i|^{\beta_1-\beta_2} \|(c_i)\|_{\ell_2(I,\alpha^{-2\beta_1})} \\ &= \|C\|^{\beta_1-\beta_2} \|g\|_{\mathcal{R}(C^{\beta_1})} < \infty\end{aligned}$$

and so $\mathcal{R}(C^{\beta_1}) \hookrightarrow \mathcal{R}(C^{\beta_2})$. Similarly, we can show that $\mathcal{R}(C) \hookrightarrow \mathcal{H}$. In the following, we first prove the result for $0 < \beta < 1$ and then for $\beta > 1$.

(a) $0 < \beta < 1$: For any $f \in \mathcal{H}$ and $g \in \mathcal{R}(C)$, we have

$$\|f-g\|_{\mathcal{H}}^2 = \left\| \sum_{i\in I\cup J} a_i\theta_i - \sum_{i\in I} c_i\phi_i \right\|_{\mathcal{H}}^2 = \left\| \sum_{i\in I\cup J} (a_i - c_i)\theta_i \right\|_{\mathcal{H}}^2 = \|(a_i-c_i)\|_{\ell_2(I\cup J)}^2$$

where we define $c_i := 0$ for $i \in J$. For $t > 0$, we find

$$\begin{aligned}K(f,t,\mathcal{H},\mathcal{R}(C)) &= \inf_{g\in\mathcal{R}(C)} \|f-g\|_{\mathcal{H}} + t\|g\|_{\mathcal{R}(C)} \\ &= \inf_{(c_i)\in\ell_2(I,\alpha^{-2})} \|(a_i-c_i)\|_{\ell_2(I\cup J)} + t\|(c_i)\|_{\ell_2(I,\alpha^{-2})} \\ &= K(a,t,\ell_2(I\cup J), \ell_2(I,\alpha^{-2}))\,.\end{aligned}$$

From this we immediately obtain the equivalence

$$f \in [\mathcal{H}, \mathcal{R}(C)]_{\beta,2} \iff (a_i) \in [\ell_2(I\cup J), \ell_2(I,\alpha^{-2})]_{\beta,2}$$

where $0 < \beta < 1$. Applying the second part of Lemma B.4 to the counting measure on $I \cup J$ yields

$$[\ell_2(I\cup J), \ell_2(I,\alpha^{-2})]_{\beta,2} = \ell_2(I, \alpha^{-2\beta}).$$

Since $\mathcal{R}(C^\beta)$ and $\ell_2(I, \alpha^{-2\beta})$ are isometrically isomorphic, we obtain $\mathcal{R}(C^\beta) = [\mathcal{H}, \mathcal{R}(C)]_{\beta,2}$.

(b) $\beta > 1$ and $\beta \notin \mathbb{N}$: Define $\gamma := \lfloor \beta \rfloor$. Let $f \in \mathcal{R}(C^\gamma)$ and $g \in \mathcal{R}(C^{\gamma+1})$, i.e., $\exists\, (c_i) \in \ell_2(I, \alpha^{-2\gamma})$ and $(d_i) \in \ell_2(I, \alpha^{-2\gamma-2})$ such that $f = \sum_{i\in I} c_i \phi_i$ and $g = \sum_{i\in I} d_i \phi_i$. Since

$$\|f - g\|^2_{\mathcal{R}(C^\gamma)} = \|(c_i - d_i)\|^2_{\ell_2(I,\alpha^{-2\gamma})},$$

for $t > 0$, we have

$$\begin{aligned} K(f, t, \mathcal{R}(C^\gamma), \mathcal{R}(C^{\gamma+1})) &= \inf_{g\in\mathcal{R}(C^{\gamma+1})} \|f - g\|_{\mathcal{R}(C^\gamma)} + t\|g\|_{\mathcal{R}(C^{\gamma+1})} \\ &= \inf_{(d_i)\in\ell_2(I,\alpha^{-2\gamma-2})} \|(c_i - d_i)\|_{\ell_2(I,\alpha^{-2\gamma})} + t\|(d_i)\|_{\ell_2(I,\alpha^{-2\gamma-2})} \\ &= K(c, t, \ell_2(I, \alpha^{-2\gamma}), \ell_2(I, \alpha^{-2\gamma-2})), \end{aligned}$$

from which we obtain the following equivalence

$$f \in [\mathcal{R}(C^\gamma), \mathcal{R}(C^{\gamma+1})]_{\beta-\gamma,2} \iff (c_i) \in [\ell_2(I, \alpha^{-2\gamma}), \ell_2(I, \alpha^{-2\gamma-2})]_{\beta-\gamma,2} \overset{(*)}{=} \ell_2(I, \alpha^{-2\beta}),$$

where $(*)$ follows from Lemma B.4 and the result is obtained by noting that $\ell_2(I, \alpha^{-2\beta})$ and $\mathcal{R}(C^\beta)$ are isometrically isomorphic. ■

B.4 Denseness of $I_k\mathcal{H}$ in $W_2(\mathbb{R}^d, p)$

In this section, we discuss the denseness of $I_k\mathcal{H}$ in $W_2(\mathbb{R}^d, p)$ for a given $p \in \mathcal{P}_{\text{FD}}$, where $\mathcal{P}_{\text{FD}}$ is defined in Theorem 13, which is equivalent to the injectivity of S_k (see Rudin, 1991, Theorem 4.12). To this end, in the following result we show that under certain conditions on a bounded continuous translation invariant kernel on $\mathbb{R}^d$, the restriction of S_k to $\mathcal{W}_2^\sim(\mathbb{R}^d, p)$ is injective when $d = 1$, while the result for any general $d > 1$ is open. However, even for $d = 1$, this does not guarantee the injectivity of S_k (which is defined on $W_2(\mathbb{R}^d, p)$). Therefore, the question of characterizing the injectivity of S_k (or equivalently the denseness of $I_k\mathcal{H}$ in $W_2(\mathbb{R}^d, p)$) is open.

Proposition B.5 *Suppose $k(x, y) = \psi(x - y)$, $x, y \in \mathbb{R}^d$ where $\psi \in C_b(\mathbb{R}^d) \cap L^1(\mathbb{R}^d)$, $\int \|\omega\|_2 \psi^\wedge(\omega)\, d\omega < \infty$ and $\operatorname{supp}(\psi^\wedge) = \mathbb{R}^d$. If $d = 1$, then the restriction of S_k to $\mathcal{W}_2^\sim(\mathbb{R}^d, p)$ is injective for any $p \in \mathcal{P}_{FD}$.*

Proof Fix any $p \in \mathcal{P}_{\text{FD}}$. We need to show that for $[f]_\sim \in \mathcal{W}_2^\sim(\mathbb{R}^d, p)$, $S_k[f]_\sim = 0$ implies $[f]_\sim = 0$. From Proposition 11, we have

$$\begin{aligned} S_k[f]_\sim &= \int_{\mathbb{R}^d} \sum_{j=1}^d \partial_j k(x, \cdot) \partial_j f(x)\, p(x)\, dx = \int_{\mathbb{R}^d} \sum_{j=1}^d \partial_j \psi(x - \cdot) \partial_j f(x)\, p(x)\, dx \\ &= \int_{\mathbb{R}^d} \sum_{j=1}^d \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} i\omega_j \psi^\wedge(\omega) e^{i\langle \omega, \cdot - x\rangle}\, d\omega\, \partial_j f(x)\, p(x)\, dx \\ &= \int_{\mathbb{R}^d} \sum_{j=1}^d \phi_j(\omega) \psi^\wedge(\omega) e^{i\langle \omega, \cdot\rangle}\, d\omega \end{aligned}$$

where

$$\phi_j(\omega) := \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} (i\omega_j) e^{-i\langle \omega, x\rangle} \partial_j f(x)\, p(x)\, dx.$$

$S_k f = 0$ implies $\sum_{j=1}^d \phi_j(\omega)\psi^\wedge(\omega) = 0$ for all $\omega \in \mathbb{R}^d$. Since $\operatorname{supp}(\psi^\wedge) = \mathbb{R}^d$, we have $\sum_{j=1}^d \phi_j(\omega) = 0$ a.e., i.e., for ω-a.e.,

$$0 = \sum_{j=1}^d \int_{\mathbb{R}^d} (i\omega_j) \partial_j f(x)\, p(x)\, e^{-i\langle \omega, x\rangle}\, dx = \sum_{j=1}^d (i\omega_j)(p\partial_j f)^\wedge(\omega).$$

For $d = 1$, this implies $(\partial_j f)p = 0$ a.e. and so $\|f\|_{W_2} = 0$. ■

Examples of kernels that satisfy the conditions in Proposition B.5 include the Gaussian, Matérn (with $\beta > 1$) and inverse multiquadrics on $\mathbb{R}$.

References

N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.

A. Barron and C-H. Sheu. Approximation of density functions by sequences of exponential families. *Annals of Statistics*, 19(3):1347–1369, 1991.

F. Bauer, S. Pereverzev, and L. Rosasco. On regularization algorithms in learning theory. *Journal of Complexity*, 23(1):52–72, 2007.

C. Bennett and R. Sharpley. *Interpolation of Operators*. Academic Press, Boston, 1988.

L. D. Brown. *Fundamentals of Statistical Exponential Families with Applications in Statistical Decision Theory*. IMS, Hayward, CA, 1986.

S. Canu and A. J. Smola. Kernel methods and the exponential family. *Neurocomputing*, 69: 714–720, 2005.

A. Caponnetto and E. De Vito. Optimal rates for regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7:331–368, 2007.

D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., New York, 1991.

E. De Vito, L. Rosasco, and A. Toigo. Learning sets with separating kernels. *http://arxiv.org/abs/1204.3573*, April 2012.

R. DeVore, G. Kerkyacharian, D. Picard, and V. Temlyakov. Mathematical methods for supervised learning. IMI Preprints, University of South Carolina, 2004.

J. Diestel and J. J. Uhl. *Vector Measures*. American Mathematical Society, Providence, 1977.

A. Doucet, N. de Freitas, and N. J. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, New York, 2001.

J. J. Duistermaat and J. A. C. Kolk. *Multidimensional Real Analysis II: Integration*. Cambridge University Press, Cambridge, UK, 2004.

H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1996.

G. B. Folland. *Real Analysis: Modern Techniques and Their Applications*. Wiley-Interscience, New York, 1999.

K. Fukumizu. Exponential manifold by reproducing kernel Hilbert spaces. In P. Gibilisco, E. Riccomagno, M.-P. Rogantin, and H. Winn, editors, *Algebraic and Geometric mothods in Statistics*, pages 291–306. Cambridge University Press, 2009.

K. Fukumizu, F. Bach, and M. Jordan. Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

K. Fukumizu, A. Gretton, X. Sun, and B. Schölkopf. Kernel measures of conditional dependence. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 489–496, Cambridge, MA, 2008. MIT Press.

K. Fukumizu, F. Bach, and M. Jordan. Kernel dimension reduction in regression. *Annals of Statistics*, 37(4):1871–1905, 2009.

K. Fukumizu, L. Song, and A. Gretton. Kernel Bayes' rule: Bayesian inference with positive definite kernels. *Journal of Machine Learning Research*, 14:3753–3783, 2013.

A. Gretton, K. Borgwardt, M. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012.

C. Gu and C. Qiu. Smoothing spline density estimation: Theory. *Annals of Statistics*, 21 (1):217–234, 1993.

A. Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6:695–709, 2005.

A. Hyvärinen. Some extensions of score matching. *Computational Statistics & Data Analysis*, 51:2499–2512, 2007.

O. Johnson. *Information Theory and The Central Limit Theorem*. Imperial College Press, London, UK, 2004.

G. S. Kimeldorf and G. Wahba. Some results on Tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33:82–95, 1971.

C. Ley and Y. Swan. Stein's density approach and information inequalities. *Electronic Communications in Probability*, 18:1–14, 2013.

S. Lyu. Interpretation and generalization of score matching. In *Proc. 25th conference on Uncertainty in Artificial Intelligence*, Montreal, Canada, 2009.

G. Pistone and C. Sempi. An infinite-dimensional geometric structure on the space of all the probability measures equivalent to a given one. *Annals of Statistics*, 23(5):1543–1561, 1995.

M. Reed and B. Simon. *Functional Analysis*. Academic Press, New York, 1972.

L. Rosasco, E. De Vito, and A. Verri. Spectral methods for regularization in learning theory. Technical Report DISI-TR-05-18, DISI, Universitá degli Studi di Genova, 2005.

W. Rudin. *Functional Analysis*. McGraw-Hill, USA, 1991.

H. Sasaki, A. Hyvärinen, and M. Sugiyama. Clustering via mode seeking by direct estimation of the gradient of a log-density. In *Proc. of European Conference on Machine Learning*, pages 19–34, 2014.

B. Schölkopf, R. Herbrich, and A. Smola. A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computational Learning Theory and 5th European Conference on Computational Learning Theory*, pages 416–426, London, UK, 2001. Springer-Verlag.

S. Smale and D.-X. Zhou. Learning theory estimates via integral operators and their approximations. *Constructive Approximation*, 26:153–172, 2007.

B. K. Sriperumbudur, A. Gretton, K. Fukumizu, B. Schölkopf, and G. R. G. Lanckriet. Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11:1517–1561, 2010.

B. K. Sriperumbudur, K. Fukumizu, and G. R. G. Lanckriet. Universality, characteristic kernels and RKHS embedding of measures. *Journal of Machine Learning Research*, 12: 2389–2410, 2011.

I. Steinwart and A. Christmann. *Support Vector Machines*. Springer, New York, 2008.

I. Steinwart and C. Scovel. Mercer's theorem on general domains: On the interaction between measures, kernels and RKHSs. *Constructive Approximation*, 35:363–417, 2012.

H. Strathmann, D. Sejdinovic, S. Livingstone, Z. Szabó, and A. Gretton. Gradient-free Hamiltonian Monte Carlo with efficient kernel exponential families. In C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, R. Garnett, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 955–963. Curran Associates, Inc., 2015.

S. Sun, M. Kolar, and J. Xu. Learning structured densities via infinite dimensional exponential families. In C. Cortes, N.D. Lawrence, D.D. Lee, M. Sugiyama, R. Garnett, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 2278–2286. Curran Associates, Inc., 2015.

L. Tartar. *An Introduction to Sobolev Spaces and Interpolation Spaces*. Springer-Verlag, Berlin, 2007.

A. B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, New York, 2009.

S. van de Geer. *Empirical Processes in M-Estimation*. Cambridge University Press, Cambridge, UK, 2000.

A. W. van der Vaart and J. H. van Zanten. Rates of contraction of posterior distributions based on Gaussian process priors. *Annals of Statistics*, 36(3):1435–1463, 2008.

L. Wasserman. *All of Nonparametric Statistics*. Springer, 2006.

H. Wendland. *Scattered Data Approximation*. Cambridge University Press, Cambridge, UK, 2005.

D.-X. Zhou. Derivative reproducing properties for kernel methods in learning theory. *Journal of Computational and Applied Mathematics*, 220:456–463, 2008.

Journal of Machine Learning Research 18 (2017) 1-52 Submitted 2/16; Revised 6/17; Published 7/17

Lens Depth Function and k-Relative Neighborhood Graph: Versatile Tools for Ordinal Data Analysis

Matthäus Kleindessner kleindes@informatik.uni-tuebingen.de
Ulrike von Luxburg luxburg@informatik.uni-tuebingen.de
Department of Computer Science
University of Tübingen
Sand 14, 72076 Tübingen, Germany

Editor: Gal Elidan

Abstract

In recent years it has become popular to study machine learning problems in a setting of ordinal distance information rather than numerical distance measurements. By ordinal distance information we refer to binary answers to distance comparisons such as $d(A, B) < d(C, D)$. For many problems in machine learning and statistics it is unclear how to solve them in such a scenario. Up to now, the main approach is to explicitly construct an ordinal embedding of the data points in the Euclidean space, an approach that has a number of drawbacks. In this paper, we propose algorithms for the problems of medoid estimation, outlier identification, classification, and clustering when given only ordinal data. They are based on estimating the lens depth function and the k-relative neighborhood graph on a data set. Our algorithms are simple, are much faster than an ordinal embedding approach and avoid some of its drawbacks, and can easily be parallelized.

Keywords: ordinal data, ordinal distance information, comparison-based algorithms, lens depth function, k-relative neighborhood graph, ordinal embedding, non-metric multi-dimensional scaling

1. Introduction

In a typical machine learning setting we are given a data set $\mathcal{D}$ of objects together with a dissimilarity function d (or a similarity function s) quantifying how "close" objects are to each other. The machine learning rationale is that objects that are close to each other tend to have the same class label, belong to the same clusters, and so on. However, in recent years a whole new branch of the machine learning literature has emerged that relaxes this scenario (e.g., Agarwal et al., 2007, Jamieson and Nowak, 2011, van der Maaten and Weinberger, 2012, Heikinheimo and Ukkonen, 2013, Kleindessner and von Luxburg, 2014, Terada and von Luxburg, 2014, Jain et al., 2016; see Section 5.1 for a discussion of related work). Instead of being able to evaluate the dissimilarity function d itself, we only get to see binary answers to some comparisons of dissimilarity values such as

$$d(A, B) \overset{?}{<} d(C, D), \tag{1}$$

where $A, B, C, D \in \mathcal{D}$. We refer to any collection of answers to such comparisons, some of them possibly being incorrect, as ordinal distance information or ordinal data.

Besides theoretical interest, there are several real-life motivations for studying machine learning tasks in a setting of ordinal distance information:

- Human-based computation / crowdsourcing: In complex tasks, such as estimating the value of a car shown in an image or clustering biographies of celebrities, it can be hard to come up with a meaningful dissimilarity function that can be evaluated automatically, while humans often have a good sense of which objects should be considered (dis-)similar. It is then natural to incorporate the human expertise into the machine learning process. As it is a general phenomenon that humans are significantly better at comparing stimuli than at identifying a single one (Stewart et al., 2005), it is widely believed and accepted that humans are also better and more reliable in assessing dissimilarity on a relative scale ("Movie A is more similar to movie B than movie C is to movie D") than on an absolute one ("The dissimilarity between A and B is 0.3 and the dissimilarity between C and D is 0.8"). For this reason, ordinal questions are often used whenever humans are involved in gathering distance information. In addition to obtaining more robust results, this also has the advantage that one does not need to align people's different assessment scales.

- There are situations where ordinal distance information is readily available, but the underlying dissimilarity function is completely in the dark. Schultz and Joachims (2003) provide the example of search-engine query logs: if a user clicks on two search results, say A and B, but not on a third result C, then A and B can be assumed to be semantically more similar than A and C, or B and C, are.

- There are several applications where actual dissimilarity values between objects can be collected, but it is clear to the practitioner that these values only reflect a rough picture and should be considered informative only on an ordinal scale level. In this case, feeding the numerical scores to a machine learning algorithm can offer the problem that the algorithm interprets them stronger than they are meant to be. For example, discarding the actual values of signal strength measurements but only keeping their order can help to reduce the influence of measurement errors and thus bring some benefit in sensor localization (Liu et al., 2004; Xiao et al., 2006).

A big part of the literature on ordinal data deals with the problem of ordinal embedding. Given a data set $\mathcal{D}$ together with ordinal relationships, the goal is to map the objects in $\mathcal{D}$ to points in a Euclidean space $\mathbb{R}^m$ such that the ordinal relationships are preserved, with respect to the Euclidean interpoint distances, as well as possible. Clearly, ordinal embedding is a way of transforming ordinal data back to a standard setting: once $\mathcal{D}$ is represented by points in $\mathbb{R}^m$, we can apply any machine learning algorithm for vector-valued data. However, such a two-step approach comes with a number of problems, among them the high running time of ordinal embedding algorithms and the necessity to choose a dimension m for the space of the embedding (to name just two—see Section 5.1.2 for a complete discussion). Our aim is to solve machine learning problems in a setting of ordinal distance information directly, without constructing an ordinal embedding as an intermediate step.

There exist several different approaches in which ordinal relationships can be evaluated (see Section 5.1.1 for more discussion and references). While comparisons of the form

©2017 Matthäus Kleindessner and Ulrike von Luxburg.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-061.html.

$d(A,B) \overset{?}{<} d(C,D)$ as in (1) are the most general form, there are other forms that, depending on the application, are of higher relevance. In particular in scenarios of human-based computation and crowdsourcing it is popular to show three objects A, B, and C at a time and to ask for information on $d(A,B) \overset{?}{<} d(A,C)$, that is, compared to (1), object D equals object A ("Which of the bottom two images is more similar to the top one?"). Recently, Heikinheimo and Ukkonen (2013) proposed an algorithm for estimating a medoid of a data set $\mathcal{D}$ based on statements of the form

$$\textit{Object } A \textit{ is the outlier within the triple of objects } (A,B,C), \qquad (\boxplus)$$

where A, B, C are pairwise distinct objects in $\mathcal{D}$ and such a statement formally means that

$$\big(d(A,B) > d(B,C)\big) \;\wedge\; \big(d(A,C) > d(B,C)\big).$$

Statements of the kind $(\boxplus)$ can easily be collected via crowdsourcing too ("Which among the following three images is the odd one out?"). In this paper, we suggest and study a similar but subtly different kind of question. Given three objects, we ask which of the objects is "the most central" object in the sense that it is the best representative for the three objects. The answers then have the form

$$\textit{Object } A \textit{ is the most central object within the triple of objects } (A,B,C) \qquad (\star)$$

with the formal interpretation that

$$\big(d(A,B) < d(B,C)\big) \;\wedge\; \big(d(A,C) < d(B,C)\big).$$

An illustration of the meaning of a statement of the kind $(\star)$ is provided in Figure 1 (left) by an example of a triple of cars consisting of a sports car, a fire truck, and an off-road vehicle: the sports car and the fire truck are rather different, but the off-road vehicle is not so different from either of them and can most likely be taken for a representative of the three cars—the off-road vehicle is the most central object within the triple.

Considering machine learning problems when given only ordinal data, in many cases it is pretty unclear how to solve them other than by constructing an ordinal embedding. For example, how can we construct a classifier based solely on a collection of answers to distance comparisons of the form (1)? **The most important insight of this paper is that ordinal distance information in the form $(\star)$ (but not in other forms—in particular, not in the form $(\boxplus)$) can immediately be related to two very helpful tools: depth functions and relative neighborhood graphs.** In a nutshell, depth functions (see, e.g., Mosler, 2013) come from multivariate statistics and are a means to generalize the concept of a univariate median to multivariate distributions and to quantify "centrality" of points with respect to such a distribution. The relative neigborhood graph (RNG; Toussaint, 1980) and its generalization, the k-RNG, are examples of proximity graphs, which play a prominent role in computer vision. In a proximity graph two vertices are connected by an edge if and only if the two vertices are in some sense close to each other. It is known from the literature that both depth functions and relative neigborhood graphs can be used to solve various machine learning problems. Our contribution is

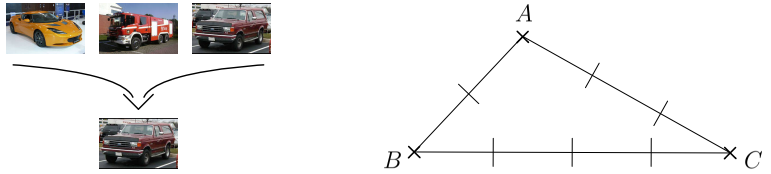


Figure 1: Illustration of the meaning of statement of the kind $(\star)$. Left: Within the three cars shown at the top, the off-road vehicle shown at the bottom a second time is the most central/best representative one. Right: A more formal approach: we have $d(A,B) < d(A,C) < d(B,C)$, and hence A is the most central data point within (A,B,C).

to establish that one particular depth function, the lens depth function (Liu and Modarres, 2011), as well as the k-RNG can be *computed* given the correct statements of the kind $(\star)$ for every triple of objects of $\mathcal{D}$, but nothing else. More importantly, the lens depth function and the k-RNG can be *estimated* when given not all but only some of possibly incorrect statements of the kind $(\star)$. This leads to algorithms solely based on ordinal data for four common machine learning problems, namely the problems of medoid estimation, outlier identification, classification, and clustering. Our algorithms are simple and can easily and highly efficiently be parallelized. We ran several experiments to compare our algorithms to competitors, in particular to the approach of first solving the ordinal embedding problem and then applying vector-based algorithms. We find that in situations with small sample size and small dimensions, the embedding approach tends to be superior to our algorithms in terms of error rates, while our algorithms are highly superior in terms of computing time (even without parallelization). The strength of our algorithms lies in the regime where the ordinal embedding algorithms break down due to computational complexity, but our algorithms still yield useful results. In any situation, our methods avoid some of the drawbacks inherent in an embedding approach.

The paper is organized as follows: We start with the setup including assumptions on the dissimilarity function d in Section 2. In Section 3 we formally define the lens depth function and the k-RNG and establish their relationships to ordinal data of the form $(\star)$. Furthermore, we motivate how we can make use of these relationships in order to solve the machine learning problems of medoid estimation, outlier identification, classification, and clustering when the only available information about a data set $\mathcal{D}$ is an arbitrary collection of statements of the kind $(\star)$. We formally state our proposed algorithms and discuss their running times, space requirements, and some implementation aspects in Section 4. Related work and further background are presented in Section 5. In Section 6 we present experiments on both artificial and real data. The paper concludes with a discussion and several directions to future work in Section 7.

2. Setup

Let $\mathcal{X}$ be an arbitrary set and $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a dissimilarity function on $\mathcal{X}$: a higher value of d means that two elements of $\mathcal{X}$ are more dissimilar to each other. The terms dissimilarity and distance are used synonymously. We assume d to satisfy the following properties for all $x, y \in \mathcal{X}$:

- $d(x, y) \geq 0$
- $d(x, y) = 0$ if and only if $x = y$
- $d(x, y) = d(y, x)$, that is d is symmetric.

With these properties, $(\mathcal{X}, d)$ is a semimetric space. Note that we do not require d to satisfy the triangle inequality, and hence $(\mathcal{X}, d)$ is not necessarily a metric space. In the following, we consider a finite subset $\mathcal{D} \subseteq \mathcal{X}$ and refer to $\mathcal{D}$ as a data set and to the elements of $\mathcal{D}$ as objects or data points.

We do not have access to d for evaluating dissimilarities between objects directly. Instead, we are only given an arbitrary collection $\mathcal{S}$ of statements

$$\textit{Object } A \textit{ is the most central object within the triple of objects } (A, B, C), \qquad (\star)$$

where (A, B, C) could be any triple of pairwise distinct objects in $\mathcal{D}$. At this point we do not make any assumptions on how $\mathcal{S}$ is related to the set of all statements, that is the set of statements of the kind $(\star)$ for all triples of objects (e.g., sampled uniformly at random). However, we need to make some assumptions if we want to provide a theoretical justification for our proposed algorithms (compare with Section 3.1 and Section 3.2.1). Statement $(\star)$ is equivalent to

$$\big(d(A, B) < d(B, C)\big) \;\wedge\; \big(d(A, C) < d(B, C)\big). \qquad (2)$$

Hence, the most central data point within a triple of data points is the data point opposite to the longest side in the triangle spanned by the three data points. An illustration of this can be seen in Figure 1 (right). Note that if we assume that there are no ties in the total order of all dissimilarities between objects, there is a unique most central object within every triple of objects. Also note that (2) is equivalent to

$$\big(d(A, B) + d(A, C)\big) < \big(d(B, A) + d(B, C)\big) \;\wedge\; \big(d(A, B) + d(A, C)\big) < \big(d(C, A) + d(C, B)\big),$$

and thus A is the medoid of $\{A, B, C\}$ (see Section 3.1 if you want to recall the definition of a medoid).

Statements might be repeatedly present in $\mathcal{S}$. More importantly, we allow $\mathcal{S}$ to be noisy due to errors in the measurement process and even inconsistent. Noisy means that $\mathcal{S}$ might comprise incorrect statements claiming that, for example, object A is the most central object within (A, B, C) although in fact object B is the most central one. Inconsistent means that we might have contradicting statements: one statement claims that object A is the most central object within (A, B, C), but another one claims that object B is. Noisy and inconsistent ordinal data is likely to be encountered in any real-world problem—think of a crowdsourcing setting, where different users will have different opinions from time to time.

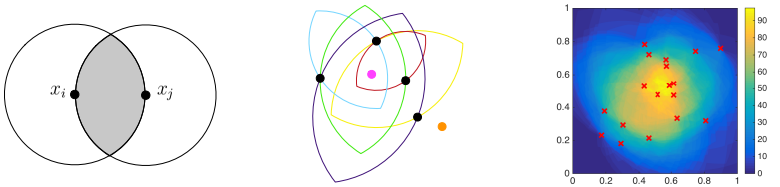


Figure 2: Left: Illustration of $Lens(x_i, x_j)$ in case of the Euclidean plane. The lens is shown in grey. Middle: The pink point at the center is contained in almost every lens spanned by any of two data points, while the orange one located at the bottom right edge of the point set is not contained in a single lens. Right: Heat map of the lens depth function for a data set of 18 points (in red) in the unit square of the Euclidean plane.

3. Lens Depth Function and k-Relative Neighborhood Graph and Motivation for our Algorithms

The most important geometric object in the following is the lens spanned by two points $x_i, x_j \in \mathcal{X}$. Consider a ball of radius $d(x_i, x_j)$ centered at x_i, and similarly a ball of the same radius centered at x_j. The lens spanned by x_i and x_j consists of all those points of $\mathcal{X}$ that are located in the intersection of these two balls. Formally,

$$\begin{aligned} Lens(x_i, x_j) &= \{x \in \mathcal{X} : d(x, x_i) < d(x_i, x_j)\} \cap \{x \in \mathcal{X} : d(x, x_j) < d(x_i, x_j)\} \\ &= \big\{x \in \mathcal{X} : \max\{d(x, x_i), d(x, x_j)\} < d(x_i, x_j)\big\}. \end{aligned}$$

An illustration of $Lens(x_i, x_j)$ in case of the Euclidean plane can be seen on the left side of Figure 2. The key insight for us are the following equivalences:

$$\begin{aligned} & x \in Lens(x_i, x_j) \Leftrightarrow \\ & d(x, x_i) < d(x_i, x_j) \text{ and } d(x, x_j) < d(x_i, x_j) \Leftrightarrow \\ & x \text{ is the most central point within } (x, x_i, x_j). \end{aligned} \qquad (3)$$

In particular, if we had knowledge of all ordinal relationships of type $(\star)$ for a data set $\mathcal{D} \subseteq \mathcal{X}$, we could check for any data point x_k and any two data points x_i, x_j whether x_k is contained in $Lens(x_i, x_j)$ or not.

3.1 Lens Depth Function

The lens depth function (Liu and Modarres, 2011) is an instance of a statistical depth function. These functions are a widely known tool in multivariate statistics. They have been designed to measure centrality with respect to point clouds or probability distributions. We will provide more information about statistical depth functions in general, including references, in Section 5.2. What makes the lens depth function special for us is that it does not rely on Euclidean structures or numeric distance values. This is in contrast to all other

depth functions from the literature. Given a data set $\mathcal{D} = \{x_1, \ldots, x_n\} \subseteq \mathcal{X}$, the lens depth function $LD(\,\cdot\,; \mathcal{D}) : \mathcal{X} \to \mathbb{N}_0$ is defined as

$$LD(x; \mathcal{D}) = \big|\, \{(x_i, x_j) : x_i, x_j \in \mathcal{D}, i < j, x \in Lens(x_i, x_j)\} \,\big|, \quad x \in \mathcal{X}.$$

To understand its meaning, consider a set of data points in the Euclidean plane. A point located at the "heart of the set" will lie in the lenses of many pairs of data points. Thus the lens depth function will attain a high value at this point, indicating its high centrality. In contrast, points at the boundary of the point cloud will lie in only a few lenses and will have a low lens depth value, indicating their low centrality. See the middle sketch of Figure 2 for an illustration. The right side of Figure 2 shows a heat map of the lens depth function for a data set consisting of 18 points in the Euclidean plane as an example.

Exploiting (3) we can see immediately how easily the lens depth function can be evaluated based on statements of the kind ($\star$). Given *all* statements of the kind ($\star$) for a data set $\mathcal{D} = \{x_1, \ldots, x_n\}$, that is one statement for every unordered triple (x_i, x_j, x_k) of pairwise distinct objects in $\mathcal{D}$, we can immediately *evaluate* $LD(x_t; \mathcal{D})$ for any $t \in \{1, \ldots, n\}$. It simply holds that

$$LD(x_t; \mathcal{D}) = \text{number of statements comprising } x_t \text{ as most central data point.} \tag{4}$$

We note that $LD(x_t; \mathcal{D})$ as given in (4) can be considered, up to a normalizing constant of $1/\binom{n-1}{2}$, as probability of the fixed data point x_t being the most central data point in a triple comprising x_t and two data points drawn uniformly at random without replacement from $\mathcal{D} \setminus \{x_t\}$. This insight gives us a handle for the realistic situation that we are not given all statements of the kind ($\star$), but only an arbitrary collection $\mathcal{S}$ of statements, some of them possibly being incorrect. Namely, we can still *estimate* $LD(x_t; \mathcal{D})$ by estimating the probability of the described event by its relative frequency:

$$LD(x_t; \mathcal{D}) \mathrel{\dot{\approx}} \frac{\text{number of statements in } \mathcal{S} \text{ that comprise } x_t \text{ as most central data point}}{\text{number of statements in } \mathcal{S} \text{ that comprise } x_t}. \tag{5}$$

This estimate will be reasonable whenever statements in $\mathcal{S}$ comprising x_t appear to be sampled approximately uniformly at random from the set of all statements that comprise x_t, the number of statements in $\mathcal{S}$ comprising x_t is large enough, and the proportion of incorrect statements is sufficiently small. Note that if we assume $\mathcal{S}$ to be sampled uniformly at random from the set of all statements, this will imply that for every $x_t \in \mathcal{D}$ statements in $\mathcal{S}$ comprising x_t are a uniform sample from the set of all statements that comprise x_t.

We now explain how we can use our insights to devise algorithms for the machine learning problems of medoid estimation, outlier identification, and classification when only given a collection of statements of the kind ($\star$) for a data set (the algorithms are formally stated in Section 4). The basic principle is that we replace the true lens depth function with its estimate according to (5) in the following existing approaches to these problems (see Section 5.2 for further information and references):

- **Medoid estimation (cf. Algorithm 1 in Section 4):** A medoid O_{MED} of a data set $\mathcal{D}$ is a most central object in the sense that it has minimal total distance to all other objects, that is it minimizes

$$D(O) = \sum_{O_i \in \mathcal{D}} d(O, O_i), \quad O \in \mathcal{D}. \tag{6}$$

  Since the lens depth function provides a measure of centrality too, even though in a different sense, a maximizer of the lens depth function (restricted to $\mathcal{D}$) is a natural candidate for an estimate of a medoid.

- **Outlier identification (cf. Algorithm 2 in Section 4):** An outlier in a data set $\mathcal{D}$ is "an observation . . . which appears to be inconsistent with the remainder of that set of data" (Barnett and Lewis, 1978, Chapter 1). Points with a low lens depth value are non-central points according to the lens depth function and thus are natural candidates for outliers. We will see in the experiments in Section 6.1.2 that this approach works well for data sets with a uni-modal structure, but can fail in multi-modal cases.

- **Classification (cf. Algorithm 3 in Section 4):** The simplest approach to classification based on the lens depth function is to assign a test point to that class in which it is a more central point: For each of the classes we could compute a separate lens depth function and evaluate a test point's corresponding depth value. The test point is then classified as belonging to the class that gives rise to the highest lens depth value. However, it has been found that such a *max-depth* approach has some severe limitations (compare with Section 5.2).

  To overcome these limitations, we use a feature-based approach. When dealing with a K-class classification problem, we consider the data-dependent feature map

$$x \mapsto (LD(x; Class_1), LD(x; Class_2), \ldots, LD(x; Class_K)) \in \mathbb{R}^K, \quad x \in \mathcal{X}, \tag{7}$$

  and then apply an out-of-the-box classification algorithm to the K-dimensional representation of the data set.

3.2 k-Relative Neighborhood Graph

We now use the lenses spanned by two data points in order to define the k-relative neighborhood graph (k-RNG). In our language, for a data set $\mathcal{D} = \{x_1, \ldots, x_n\} \subseteq \mathcal{X}$ and a parameter $k \in \mathbb{N}$ the k-RNG on $\mathcal{D}$ is the graph with vertex set $\mathcal{D}$ in which two distinct vertices x_i and x_j are connected by an undirected edge if and only if the lens spanned by these points contains fewer than k data points from $\mathcal{D}$:

$$x_i \sim x_j \;\Leftrightarrow\; |Lens(x_i, x_j) \cap \mathcal{D}| < k. \tag{8}$$

The rationale behind this definition is that two data points may be considered close to each other whenever the lens spanned by them contains only a few data points. The k-relative neighborhood graph is best known when $k = 1$. In this form it is simply called relative neighborhood graph (RNG) and has already been introduced in Toussaint (1980). The general k-RNG has been defined by Chang et al. (1992). Examples for a data set in

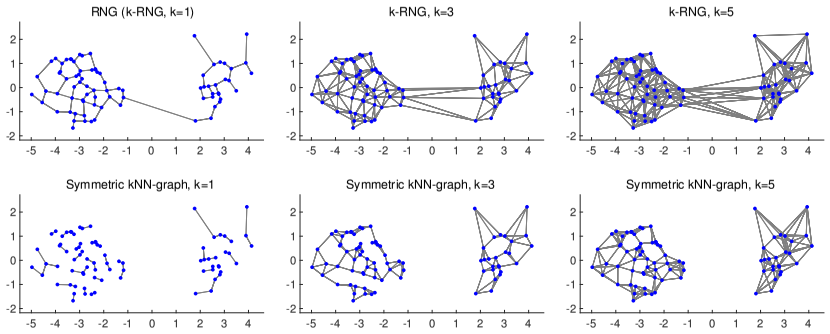


Figure 3: k-relative neighborhood graphs (1st row) and symmetric k-nearest neighbor graphs (2nd row) on 80 points from a mixture of two Gaussians. Note that as opposed to the k-NN graphs, the k-relative neighborhood graphs tend to have more connections between points from the different mixture components. In fact, a k-RNG is always connected (see Section 5.3). This might be desirable in some situations, but undesirable in others.

the Euclidean plane can be seen in Figure 3. For comparison, we also also show symmetric k-nearest neighbor graphs on the data set. The symmetric k-nearest neighbor graph or k-NN graph for short, also with parameter $k \in \mathbb{N}$, is more popular in machine learning. In that graph two vertices are connected by an undirected edge whenever one of them is among the k closest data points to the other one (with respect to the distance function d).

Given *all* statements of the kind $(\star)$ for a data set $\mathcal{D}$, it is straightforward to build the *true* k-RNG on $\mathcal{D}$ similarly to the exact evaluation of the lens depth function (4). Below, we will discuss how to build an *estimate* of the k-RNG on $\mathcal{D}$ when given only an arbitrary collection of statements, some of them possibly being incorrect, and a problem involved in Section 3.2.1. Before, let us explain how k-relative neighborhood graphs can be used for classification and clustering.

- **Classification (cf. Algorithm 4 in Section 4):** Given a set of labeled points and an additional test point that we would like to classify, we can construct the k-RNG on the union of the set of labeled points and the singleton of the test point and take a majority vote of the test point's neighbors in the graph. There is no need to construct the whole graph. We just have to find the test point's neighbors in the graph. Note that the basic principle is the same as for the well-known k-NN classifier (e.g., Shalev-Shwartz and Ben-David, 2014, Chapter 19), replacing the directed k-NN graph by the k-RNG.
- **Clustering (cf. Algorithm 5 in Section 4):** As we can do with the symmetric k-NN graph, it is straightforward to apply spectral clustering to the k-RNG on a data set $\mathcal{D}$ (see von Luxburg, 2007, for a comprehensive introduction to spectral clustering—that work suggests the symmetric k-NN graph as one of a few graphs that can be used). We propose

two versions: one is to simply work with an estimate of the ordinary unweighted k-RNG, the other one is to use an estimate of a k-RNG in which an edge between connected vertices x_i and x_j is weighted by

$$\exp\left(-\frac{1}{\sigma^2}\cdot\frac{|Lens(x_i,x_j)\cap\mathcal{D}|^2}{(|\mathcal{D}|-2)^2}\right) \tag{9}$$

for a scaling parameter $\sigma > 0$.

3.2.1 The Problem of Estimating the k-RNG from Noisy Ordinal Data

The key insight for estimating the k-RNG on a data set $\mathcal{D}$ from ordinal distance information of type $(\star)$ is similar to the one for estimating the lens depth function: the characterization (8) is equivalent to two distinct, fixed data points x_i and x_j being connected in the k-RNG if and only if the probability of a data point drawn uniformly at random from $\mathcal{D}\setminus\{x_i,x_j\}$ lying in $Lens(x_i,x_j)$ is smaller than $k/(|\mathcal{D}|-2)$. Given a collection $\mathcal{S}$ of statements of the kind $(\star)$, this probability can be estimated by

$$V(x_i,x_j)=\frac{N(x_i,x_j)}{D(x_i,x_j)}, \tag{10}$$

where

$$\begin{aligned} N(x_i,x_j) &= \text{number of statements in } \mathcal{S} \text{ comprising both } x_i \text{ and } x_j \text{ and} \\ &\qquad \text{another data point as most central data point,} \\ D(x_i,x_j) &= \text{number of statements in } \mathcal{S} \text{ comprising both } x_i \text{ and } x_j. \end{aligned} \tag{11}$$

Thus our strategy to estimate the k-RNG on $\mathcal{D}$ is the following: we connect two data points x_i and x_j with $i\neq j$ by an undirected edge if and only if

$$V(x_i,x_j) < \frac{k}{|\mathcal{D}|-2}. \tag{12}$$

If all statements in $\mathcal{S}$ are correct and, for every x_i and x_j with $i\neq j$, there are sufficiently many statements in $\mathcal{S}$ that comprise both x_i and x_j and these statements appear to be sampled approximately uniformly at random from the set of all statements that comprise x_i and x_j, we can expect our estimate of the k-RNG to be reasonable.

However, incorrect statements in $\mathcal{S}$ create a problem for our strategy. Usually, we are interested in a k-RNG for a small value of the parameter k, aiming at connecting only data points that are close to each other. Consequently, according to (12), in order that the data points x_i and x_j are connected in our estimate of the k-RNG, the estimated probability $V(x_i,x_j)$ has to be small. However, in case of erroneous ordinal data comprising sufficiently many incorrect statements, there will always be statements wrongly indicating that there are some data points in $Lens(x_i,x_j)$ that in fact are not, and thus $V(x_i,x_j)$ will always be somewhat large. Hence, many of the edges of the true k-RNG on $\mathcal{D}$ will not be present in our estimate.
To make this formal, consider the following simple noise model: Statements of the kind $(\star)$ are incorrect, independently of each other, with some fixed probability *errorprob*. In an

incorrect statement the two data points that are not most central appear to be most central with probability 1/2 each. In our experiments in Section 6.1, this noise model is referred to as Noise model I. Assume $\mathcal{S}$ to be sampled uniformly at random from all statements. Denote by $p = p(x_i, x_j)$ the probability that a data point drawn uniformly at random from $\mathcal{D} \setminus \{x_i, x_j\}$ lies in $Lens(x_i, x_j)$, that is $p = |Lens(x_i, x_j) \cap \mathcal{D}|/(|\mathcal{D}| - 2)$. Denote by $\tilde{p} = \tilde{p}(x_i, x_j)$ the probability that the following experiment yields a positive result: A data point is drawn uniformly at random from $\mathcal{D} \setminus \{x_i, x_j\}$. Independently, a Bernoulli trial with a probability of success equaling *errorprob* is performed. If the Bernoulli trial fails, the experiment yields a positive result if and only if the drawn data point falls into $Lens(x_i, x_j)$. If the Bernoulli trial succeeds, the experiment yields a positive result if and only if the data point does not fall into $Lens(x_i, x_j)$ and another Bernoulli trial, with a probability of success of one half and performed independently, succeeds. It is clear that under the considered model, $V(x_i, x_j)$ as given in (10) and (11) is an estimate of $\tilde{p}$ rather than of p. Assuming that *errorprob* is less than 2/3, we can relate $\tilde{p}$ and p via

$$\tilde{p} = p \cdot (1 - errorprob) + (1 - p) \cdot errorprob \cdot \frac{1}{2}, \tag{13}$$

or equivalently

$$p = \frac{\tilde{p} - \frac{1}{2} \cdot errorprob}{1 - \frac{3}{2} \cdot errorprob}. \tag{14}$$

The probability $\tilde{p}$ is obtained from p by applying an affine transformation and vice versa. It follows from (13) that our strategy yields an estimate of the k'-RNG with

$$k' = \frac{k - \frac{1}{2} \cdot errorprob \cdot (|\mathcal{D}| - 2)}{1 - \frac{3}{2} \cdot errorprob} \tag{15}$$

rather than of the intended k-RNG. In particular, we have $k' < k$ for $k < \frac{1}{3}(|\mathcal{D}| - 2)$ and $k' \leq 0$ for $k \leq \frac{1}{2} \cdot errorprob \cdot (|\mathcal{D}| - 2)$. This means that whenever $k < \frac{1}{3}(|\mathcal{D}| - 2)$, our strategy produces an estimate containing fewer edges than we would like to have, and whenever $k \leq \frac{1}{2} \cdot errorprob \cdot (|\mathcal{D}| - 2)$, it even produces an estimate of an empty graph, that is a graph without any edges at all.

These findings might seem worse than they actually are: using our estimated graph for classification or clustering, we do not care whether we work with the estimate of a k'-RNG instead of a k-RNG, but only whether our classification or clustering result is useful. However, we have to bear them in mind when choosing the parameter k in our algorithms: Using cross-validation for choosing k for Algorithm 4 (classification by means of a majority vote of neighbors in the graph), we may only use Leave-one-out cross-validation variants since we have to ensure roughly the same size of the training set during cross-validation and the training set in the ultimate classification task. Otherwise, a value of k that is optimal during cross-validation will not be optimal in the ultimate classification problem since k' depends on $|\mathcal{D}|$ as stated in (15). Applying Algorithm 5 (spectral clustering on the estimated k-RNG), we have to choose k so large that the constructed graph is connected. This is not only required by some versions of spectral clustering, but also indicates that the graph is indeed an estimate of a true k'-RNG with $k' \geq 1$ rather than of an empty graph.

If we know the value of *errorprob*, or have at least an estimate of it, we can correct for the bias of our strategy. In order to estimate the k-RNG on a data set $\mathcal{D}$ for the intended value of k, according to (14), two data points x_i and x_j with $i \neq j$ should be connected if and only if

$$\frac{V(x_i, x_j) - \frac{1}{2} \cdot errorprob}{1 - \frac{3}{2} \cdot errorprob} < \frac{k}{|\mathcal{D}| - 2}, \tag{16}$$

which equals (12) if $errorprob = 0$. Note that although the left-hand side of equation (16) is an unbiased estimator of $p(x_i, x_j)$ for every x_i and x_j with $i \neq j$ (assuming $\mathcal{S}$ to be sampled uniformly at random from all statements), due to the thresholding step in (16) our estimation strategy is still not an unbiased estimator of the intended k-RNG.

4. Algorithms for Medoid Estimation, Outlier Identification, Classification, and Clustering

In this section we formally state our algorithms for the problems of medoid estimation, outlier identification, classification, and clustering when the only available information about a data set $\mathcal{D}$ is a collection $\mathcal{S}$ of statements of the kind $(\star)$. Furthermore, we discuss running times, space requirements, and some implementation aspects.

4.1 Medoid Estimation

The following Algorithm 1 returns as output an estimate of a medoid of $\mathcal{D}$ as motivated in Section 3.1. The estimate is given by an object that maximizes the estimated lens depth function on $\mathcal{D}$. By setting the estimated lens depth value $LD(O)$ to zero for objects O that do not appear in any statement in $\mathcal{S}$, which means that we do not have any information about O, we ensure that such an object is never returned as output (unless there is no available information about $\mathcal{D}$ at all, i.e. $\mathcal{S} = \emptyset$).

Algorithm 1 Estimating a medoid

Input: a collection $\mathcal{S}$ of statements of the kind $(\star)$ for some data set $\mathcal{D}$

Output: an estimate of a medoid of $\mathcal{D}$

1: for every object O in $\mathcal{D}$ compute

$$LD(O) := \frac{\text{number of statements comprising } O \text{ as most central object}}{\text{number of statements comprising } O}$$

▷ if the denominator equals zero, set $LD(O) = 0$

2: **return** an object O for which $LD(O)$ is maximal

If we assume that every object in $\mathcal{D}$ can be identified by a unique index from $\{1, \ldots, |\mathcal{D}|\}$ and, given a statement in $\mathcal{S}$, the indices of the three objects involved can be accessed in constant time, then Algorithm 1 can be implemented with $\mathcal{O}(|\mathcal{D}| + |\mathcal{S}|)$ time and $\mathcal{O}(|\mathcal{D}|)$ space in addition to storing $\mathcal{S}$. This can be done by going through $\mathcal{S}$ only once and updating counters for the three objects found in a statement. If the objects in $\mathcal{D}$ are not indexed by

$1, \ldots, |\mathcal{D}|$, we can use minimal perfect hashing in order to first create such an indexing. This requires about $\mathcal{O}(|\mathcal{D}|)$ time and space (Hagerup and Tholey, 2001; Botelho et al., 2007), so the overall requirements remain unaffected by this additional step. An important feature of Algorithm 1 is that it can easily be parallelized by partitioning $\mathcal{S}$ into several subsets that may be processed independently. Since one usually may expect that $|\mathcal{S}| \gg |\mathcal{D}|$, such a parallelization has almost ideal speedup, that is doubling the number of processing elements leads to almost only half of the running time.

4.2 Outlier Identification

By means of the following Algorithm 2 we can identify outliers in $\mathcal{D}$ given as input only a collection $\mathcal{S}$ of statements of the kind $(\star)$. Outlier candidates are data points with low estimated lens depth values $LD(O)$. By setting $LD(O)$ to zero for objects O that do not appear in any statement we guarantee that such objects are identified as outliers.

Algorithm 2 Identifying outlier candididates

Input: a collection $\mathcal{S}$ of statements of the kind $(\star)$ for some data set $\mathcal{D}$

Output: a subset of $\mathcal{D}$ containing objects that are outlier candidates

1: for every object O in $\mathcal{D}$ compute

$$LD(O) := \frac{\text{number of statements comprising } O \text{ as most central object}}{\text{number of statements comprising } O}$$

▷ if the denominator equals zero, set $LD(O) = 0$

2: identify objects with exceptionally small values of $LD(O)$

3: **return** the set of identified objects

The only difference between Algorithm 2 and Algorithm 1 is that instead of returning the object with the highest value of $LD(O)$ as estimate of a medoid we return objects with exceptionally small values as outlier candidates. The running time of Algorithm 2 depends on the identification strategy in Step 2, but if one simply identifies c objects with smallest values ($1 \leq c \leq |\mathcal{D}|$), then Algorithm 2 can be implemented with $\mathcal{O}(|\mathcal{D}| + |\mathcal{S}|)$ time and $\mathcal{O}(|\mathcal{D}|)$ space in addition to storing $\mathcal{S}$ analogously to Algorithm 1. Here we make use of the fact that the selection of the c-th smallest value in an array of length $|\mathcal{D}|$ can be done in $\mathcal{O}(|\mathcal{D}|)$ time and space (Blum et al., 1973). Just as for Algorithm 1, the first step of Algorithm 2 can easily be parallelized.

4.3 Classification

We propose two different algorithms for dealing with K-class classification in a data set $\mathcal{D}$ consisting of a subset $\mathcal{L}$ of labeled objects and a subset $\mathcal{U}$ of unlabeled objects when given no more information than the class labels for the objects in $\mathcal{L}$ and a collection $\mathcal{S}$ of statements of the kind $(\star)$ for $\mathcal{D}$. Our goal is to predict a class label for every object in $\mathcal{U}$.

Our first proposed algorithm, Algorithm 3, is based on the lens depth function and has been motivated in Section 3.1. It consists of computing a feature embedding of $\mathcal{D}$ into

$[0,1]^K \subseteq \mathbb{R}^K$, in which each feature corresponds to the estimated lens depth value with respect to one class, and subsequently applying a classification algorithm that is suitable for K-class classification on $\mathbb{R}^K$ to this embedding.

Algorithm 3 K-class classification I

Input: a collection $\mathcal{S}$ of statements of the kind $(\star)$ for some data set $\mathcal{D}$ comprising a set $\mathcal{L}$ of labeled objects and a set $\mathcal{U}$ of unlabeled objects; a class label for every labeled object in $\mathcal{L}$ according to its membership in one of K classes (referred to as $Class_1, \ldots, Class_K$)
▷ note that we have $\mathcal{D} = \mathcal{L} \,\dot\cup\, \mathcal{U}$ and $\mathcal{L} = Class_1 \,\dot\cup\, Class_2 \,\dot\cup\, \ldots \,\dot\cup\, Class_K$

Output: an inferred class label for every unlabeled object in $\mathcal{U}$

1: for every object O in $\mathcal{D}$ and $i \in \{1, \ldots, K\}$ compute

$N_{C_i}(O) :=$ number of statements comprising O and two labeled objects from $Class_i$ with O as most central object

$D_{C_i}(O) :=$ number of statements comprising O and two labeled objects from $Class_i$

$$LD_{C_i}(O) := \frac{N_{C_i}(O)}{D_{C_i}(O)}$$

▷ if $D_{C_i}(O)$ equals zero, set $LD_{C_i}(O) = 0$

2: train an arbitrary classifier (suitable for K-class classification on $\mathbb{R}^K$) with training data

$$\{(LD_{C_1}(O_l), LD_{C_2}(O_l), \ldots, LD_{C_K}(O_l)) : O_l \in \mathcal{L}\} \subseteq \mathbb{R}^K$$

where the label of $(LD_{C_1}(O_l), LD_{C_2}(O_l), \ldots, LD_{C_K}(O_l))$ equals the label of O_l

3: **return** as inferred class label of every unlabeled object $O_u \in \mathcal{U}$ the label predicted by the classifier applied to $(LD_{C_1}(O_u), LD_{C_2}(O_u), \ldots, LD_{C_K}(O_u)) \in \mathbb{R}^K$

Assuming that the number of classes K is bounded by a constant, the first step of Algorithm 3 requires $\mathcal{O}(|\mathcal{D}| + |\mathcal{S}|)$ operations and $\mathcal{O}(|\mathcal{D}|)$ space in addition to storing $\mathcal{S}$. This is the same as for Algorithm 1 and Algorithm 2. As before, this step can easily and highly efficiently be parallelized (assuming that $|\mathcal{S}| \gg |\mathcal{D}|$). The time and space complexities of the remaining steps depend on the generic classifier that is used.

Our second proposed algorithm, Algorithm 4, is based on the k-RNG and has been motivated in Section 3.2. It is an instance-based learning method like the well-known k-NN classifier: There is no explicit training phase involved. An unlabeled object is readily classified by assigning the label that is most frequently encountered among the neighbors of the unlabeled object in the estimated k-RNG.
Assuming that the number of classes K is bounded by a constant, Algorithm 4 can be implemented with $\mathcal{O}(|\mathcal{D}|+|\mathcal{U}|\cdot|\mathcal{L}|+|\mathcal{S}|) = \mathcal{O}(|\mathcal{U}|\cdot|\mathcal{L}|+|\mathcal{S}|)$ time and $\mathcal{O}(|\mathcal{D}|+|\mathcal{U}|\cdot|\mathcal{L}|) = \mathcal{O}(|\mathcal{U}|\cdot|\mathcal{L}|)$ space in addition to storing $\mathcal{S}$. Here we have to assign to each labeled object a unique identifier in $\{1, \ldots, |\mathcal{L}|\}$ and to each unlabeled object a unique identifier in $\{1, \ldots, |\mathcal{U}|\}$ that can be looked up in constant time. This allows us to increment a value of $N(O_u, O_l)$ or $D(O_u, O_l)$ for $(O_u, O_l) \in \mathcal{U} \times \mathcal{L}$ stored in an array of size $|\mathcal{U}| \times |\mathcal{L}|$ within constant time. Once

Algorithm 4 K-class classification II

Input: a collection $\mathcal{S}$ of statements of the kind $(\star)$ for some data set $\mathcal{D}$ comprising a set $\mathcal{L}$ of labeled objects and a set $\mathcal{U}$ of unlabeled objects; a class label for every labeled object in $\mathcal{L}$ according to its membership in one of K classes (referred to as $Class_1, \ldots, Class_K$); an integer parameter k

▷ note that we have $\mathcal{D} = \mathcal{L} \,\dot\cup\, \mathcal{U}$ and $\mathcal{L} = Class_1 \,\dot\cup\, Class_2 \,\dot\cup\, \ldots \,\dot\cup\, Class_K$

Output: an inferred class label for every unlabeled object in $\mathcal{U}$

1: for every unlabeled object $O_u \in \mathcal{U}$ and every labeled object $O_l \in \mathcal{L}$ compute

$$\begin{aligned} N(O_u, O_l) &:= \text{number of statements comprising both } O_u \text{ and } O_l \text{ and another} \\ &\qquad \text{labeled object as most central object} \\ D(O_u, O_l) &:= \text{number of statements comprising both } O_u \text{ and } O_l \text{ and another} \\ &\qquad \text{labeled object} \\ V(O_u, O_l) &:= \frac{N(O_u, O_l)}{D(O_u, O_l)} \end{aligned}$$

▷ if $D(O_u, O_l)$ equals zero, set $V(O_u, O_l) = \infty$

2: **return** as inferred class label of every unlabeled object $O_u \in \mathcal{U}$ the majority vote (ties broken randomly) of the labels of those objects $O_l \in \mathcal{L}$ that satisfy

$$V(O_u, O_l) < \frac{k}{|\mathcal{L}| - 1}$$

the objects are indexed by $1, \ldots, |\mathcal{D}|$ (compare with Section 4.1), we can easily assign such identifiers in $\mathcal{O}(|\mathcal{D}|)$ time and space. Again, it is straightforward to parallelize Algorithm 4 by partitioning $\mathcal{S}$.

4.4 Clustering

Our proposed Algorithm 5 for clustering a data set $\mathcal{D}$ when only given a collection $\mathcal{S}$ of statements of the kind $(\star)$ as input consists of estimating the k-RNG on $\mathcal{D}$ and applying spectral clustering to the estimate. Note that some versions of spectral clustering require the underlying similarity graph not to contain isolated vertices. A true k-RNG never contains isolated vertices since a k-RNG is always connected (compare with Section 5.3), but if k is chosen too small, an estimated k-RNG might contain isolated vertices (compare with Section 3.2.1).

The first step of Algorithm 5 can be implemented with $\mathcal{O}(|\mathcal{D}|^2 + |\mathcal{S}|)$ time and $\mathcal{O}(|\mathcal{D}|^2)$ space in addition to storing $\mathcal{S}$. It can be parallelized in the same way as the corresponding parts of the previous algorithms. However, here we achieve almost ideal speedup only in case $|\mathcal{S}| \gg |\mathcal{D}|^2$. The second step can be implemented with $\mathcal{O}(|\mathcal{D}|^2)$ time and $\mathcal{O}(|\mathcal{D}|^2)$ space. The complexity of Step 3 is the one of spectral clustering after the construction of a similarity graph. Its costs are dominated by the complexity of eigenvector computations and are

Algorithm 5 Clustering

Input: a collection $\mathcal{S}$ of statements of the kind $(\star)$ for some data set $\mathcal{D} = \{O_1, \ldots, O_n\}$; an integer parameter k; number l of clusters to construct; a parameter $\sigma > 0$ in case of weighted version

Output: a hard clustering $C_1, \ldots, C_l \subseteq \mathcal{D}$ with $C_1 \,\dot\cup\, C_2 \,\dot\cup\, \ldots \,\dot\cup\, C_l = \mathcal{D}$

1: for every pair (O_i, O_j) of objects in $\mathcal{D}$ compute

$$\begin{aligned} N(O_i, O_j) &:= \text{number of statements comprising both } O_i \text{ and } O_j \\ &\qquad \text{and another object as most central object} \\ D(O_i, O_j) &:= \text{number of statements comprising both } O_i \text{ and } O_j \\ V(O_i, O_j) &:= \frac{N(O_i, O_j)}{D(O_i, O_j)} \end{aligned}$$

▷ if $D(O_i, O_j) = 0$, set $V(O_i, O_j) = \infty$ (in particular, $V(O_i, O_i) = \infty$ for $i = 1, \ldots, n$)

2: let $W = (w_{ij})_{i,j=1,\ldots,n}$ be a (n, n)-matrix and either set

$$W_{ij} = \begin{cases} 1 & \text{if } V(O_i, O_j) < k/(|\mathcal{D}| - 2) \\ 0 & \text{else} \end{cases} \qquad \triangleright \text{ unweighted version}$$

or

$$W_{ij} = \begin{cases} e^{-\frac{V(O_i, O_j)^2}{\sigma^2}} & \text{if } V(O_i, O_j) < k/(|\mathcal{D}| - 2) \\ 0 & \text{else} \end{cases} \qquad \triangleright \text{ weighted version}$$

3: apply spectral clustering to W with l as input parameter for the number of clusters

4: **return** clusters $C_1, \ldots, C_l$ according to the clusters produced in Step 3

commonly stated to be in general in $\mathcal{O}(n^3) = \mathcal{O}(|\mathcal{D}|^3)$ regarding time and $\mathcal{O}(n^2) = \mathcal{O}(|\mathcal{D}|^2)$ regarding space for an arbitrary number of clusters l, unless approximations are applied (Yan et al., 2009; Li et al., 2011). In many cases the estimate of the k-RNG constructed by Algorithm 5 might be sparse (compare with Section 5.3), and then the eigenvector computations can be done much more efficiently (Bai et al., 2000). However, in the worst case the overall running time of Algorithm 5 can be up to $\mathcal{O}(|\mathcal{D}|^3 + |\mathcal{S}|)$. The overall space requirements are $\mathcal{O}(|\mathcal{D}|^2)$ in addition to storing $\mathcal{S}$.

5. Related Work and Further Background

In this section we present related work and further background on ordinal data analysis, statistical depth functions, and the k-relative neighborhood graph. In a first reading, the reader may skip this part and go to the experimental Section 6 immediately.

5.1 Machine Learning in a Setting of Ordinal Distance Information

We have mentioned in Section 1 that ordinal data can be distinguished with respect to the kind of ordinal relationships that it consists of and that ordinal embedding is a general approach to machine learning in a setting of ordinal distance information. Here we discuss these two topics in more detail.

5.1.1 Different Types of Ordinal Data

The most general form of ordinal distance information consists of binary answers to some dissimilarity comparisons

$$d(A,B) \overset{?}{<} d(C,D), \tag{17}$$

where A, B, C, D could be any objects of some data set. In the machine learning literature, this very general type of ordinal data has been studied in Agarwal et al. (2007), Kleindessner and von Luxburg (2014), Terada and von Luxburg (2014), and Arias-Castro (2015).
The type most often studied in the literature is the one of similarity triplets (Jamieson and Nowak, 2011; Tamuz et al., 2011; van der Maaten and Weinberger, 2012; Wilber et al., 2014; Amid and Ukkonen, 2015; Heim et al., 2015; Amid et al., 2016; Jain et al., 2016; Haghiri et al., 2017). Similarity triplets are answers to dissimilarity comparisons of the restricted form

$$d(A,B) \overset{?}{<} d(A,C). \tag{18}$$

Compared to (17), A equals D and serves as an anchor point.
Another well-known type of ordinal data is the directed, but unweighted k-nearest neighbor graph on a data set (Shaw and Jebara, 2009; von Luxburg and Alamgir, 2013; Terada and von Luxburg, 2014; Hashimoto et al., 2015; Kleindessner and von Luxburg, 2015). This graph provides the ordinal dissimilarity relationships

$$d(V,N) < d(V,O)$$

for objects V, N, and O such that N is adjacent to V in the graph, but O is not.

Ordinal distance information in the form ($\star$), which we consider in this work, is similar to statements of the form (compare with Section 1)

$$\textit{Object } A \textit{ is the outlier within the triple of objects } (A,B,C). \tag{$\boxplus$}$$

This type of ordinal data has been studied by Heikinheimo and Ukkonen (2013) and also by Ukkonen et al. (2015). Heikinheimo and Ukkonen (2013) proposed an algorithm for estimating a medoid of a data set based on statements of the kind ($\boxplus$). Their approach is closely related to ours (compare with Algorithm 1): For every fixed data point, they estimate the probability that the data point is the outlier within a triple of three data points containing the fixed data point and two data points chosen uniformly at random from the remaining ones. Then they take the data point with minimal estimated probability as an

estimate of a medoid. However, the conceptual problem with their approach is that the function that it is based on,

$$F(x;P) = 1 - \mathit{Probability}(x \text{ is the outlier within the triple of points } (x,X,Y)), \tag{19}$$

is *not* a valid statistical depth function. It does not satisfy one of the most crucial properties of statistical depth functions, namely maximality at the center for symmetric distributions (see Section 5.2). As a consequence, their approach always fails to return a true medoid for certain data sets, even though given access to the correct statements of the kind ($\boxplus$) for all triples of data points. As we will see in the experiments in Section 6.1.1, Algorithm 1 consistently achieves better results in recovering a true medoid of a data set compared to the method by Heikinheimo and Ukkonen when both methods are given the same number of statements, either of the kind ($\star$) or of the kind ($\boxplus$), as input.
As Heikinheimo and Ukkonen remark, one can adapt their method to the problem of outlier identification by considering data points with high estimated probabilities as outlier candidates—in the same way as Algorithm 2 is related to Algorithm 1. In the experiments in Section 6.1.2 we will compare Algorithm 2 to such an approach.

Dealing with ordinal distance information comes with a critical drawback compared to a standard setting of cardinal distance information. While for a data set comprising n objects there are in total "only" $\Theta(n^2)$ distances between objects, there are $\Theta(n^4)$ different distance comparisons of the form (17). If one only allows for comparisons of the form (18), that is one considers similarity triplets, there are still $\Theta(n^3)$ different comparisons. This is also the order of magnitude for the number of all statements of the kind ($\star$) or ($\boxplus$). Unless n is rather small, in practice it is prohibitive to collect all statements or answers to all different distance comparisons. The hope is that much fewer statements or answers already contain the bulk of usable information due to high redundancy in the ordinal data. This gives rise to distinguishing between a batch setting and an active setting in the study of algorithms for ordinal distance information: while in a batch setting we are given the ordinal data a priori, in an active setting we are allowed to query ordinal relationships, trying to do it in such a way as to exploit redundancy (Jamieson and Nowak, 2011; Tamuz et al., 2011). Our Algorithms 1 to 5 are designed for the general batch setting. We leave it for future work to devise algorithms for the considered problems in an active setting (see Section 7).

5.1.2 Ordinal Embedding

One important and general approach to machine learning in a setting of ordinal distance information is to construct an ordinal embedding of the data set, that is to map data points to points in a Euclidean space $\mathbb{R}^m$ such that the embedding (with respect to the Euclidean interpoint distances) preserves the given ordinal data as well as possible. After doing so, one can simply apply any algorithm designed for vector-valued data to the embedding for solving the task at hand. This approach is justified by theoretical results showing that for a sufficiently large number of given ordinal relationships and data sets that can be perfectly embedded (this means that all available ordinal relationships are preserved) the embedding is uniquely determined up to similarity transformations as the size of the data set goes to

infinity (Kleindessner and von Luxburg, 2014; Arias-Castro, 2015).

The problem of ordinal embedding dates back to the development of ordinal multidimensional scaling in the 1960s (also known as non-metric multidimensional scaling; Shepard, 1962a,b, and Kruskal, 1964a,b, also see the monograph Borg and Groenen, 2005). More recently, it has been studied in the machine learning community resulting in a number of algorithms (Agarwal et al., 2007; Shaw and Jebara, 2009; Tamuz et al., 2011; van der Maaten and Weinberger, 2012; Terada and von Luxburg, 2014; Amid and Ukkonen, 2015; Heim et al., 2015; Amid et al., 2016; Jain et al., 2016). For none of these algorithms theoretical bounds for their complexity are available in the literature, but it is widely known that they are utterly slow and not appropriate when dealing with large data sets and/or many ordinal relationships (this is confirmed by our experiments in Section 6.1.1). Furthermore, these algorithms either solve a non-convex optimization problem or a relaxed version of such one, in both cases involving the risk of finding only a suboptimal solution. Often, their outcome depends on a random initialization of the ordinal embedding. Moreover, the choice of the dimension of the space of the embedding can be crucial and highly influences the running time of the algorithms, as does the amount of noise in the available ordinal data. All these are strong arguments for aiming to solve machine learning problems in a setting of ordinal distance information directly, that is without constructing an ordinal embedding as an intermediate step, and thus for our proposed Algorithms 1 to 5.

5.2 Statistical Depth Functions and Lens Depth Function

Statistical depth functions (see, e.g., Serfling, 2006, Cascos, 2009, Mosler, 2013, or the introduction of the dissertation of Van Bever, 2013, for basic reviews) have been developed to generalize the concept of the univariate median to multivariate distributions. To this end, a depth function is supposed to measure the centrality of all points $x \in \mathbb{R}^m$ with respect to a probability distribution, in the sense that the depth value at x is high if x resides in the "middle" of the distribution and that it is lower the more distant from the mass of the distribution x is located.
The first statistical depth function has been proposed by Tukey (1974). Given a probability distribution P on $\mathbb{R}^m$, the seminal halfspace depth function HD maps every point $x \in \mathbb{R}^m$ to the smallest probability of a closed halfspace containing x, that is

$$HD(x;P) = \inf_{u \in S^{m-1}} P(\{y \in \mathbb{R}^m : u^T(y-x) \geq 0\}),$$

where $S^{m-1} = \{u \in \mathbb{R}^m : u^T u = 1\}$ denotes the unit sphere in $\mathbb{R}^m$. The intuition behind this definition is simplest to understand in case of an absolutely continuous distribution P: in this case $HD(x;P) \leq 1/2$, $x \in \mathbb{R}^m$, and in order for a point x to be considered central with respect to P it should hold that any hyperplane passing through x splits $\mathbb{R}^m$ into two halfspaces of almost equal probability $1/2$. Hence, points x are considered more central the higher their halfspace depth value $HD(x;P)$ is, and any point maximizing $HD(\,\cdot\,;P)$ is called a Tukey median. Figure 4 shows examples of the halfspace depth function for two absolutely continuous distributions on $\mathbb{R}^2$. Note that a depth function can resemble the density function of the underlying distribution only in case of a unimodal distribution—as

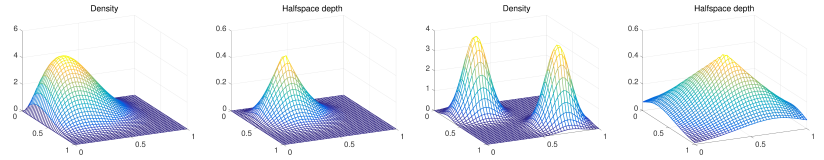


Figure 4: Illustration of the halfspace depth function. Mesh plot of the density and the halfspace depth function of a product of two $Beta(2,4)$-distributions (1st & 2nd plot) and a mixture of two Gaussians (3rd & 4th plot), respectively.

a measure of global centrality depth functions are intended to be unimodal. We will take this up again in Section 6.1.2 and Section 7.
For a univariate and continuous distribution any ordinary median is also a Tukey median. In addition, the halfspace depth function HD satisfies a number of desirable properties:

1. Affine invariance: HD considered as a function in both x and P is invariant under affine transformations.

2. Maximality at the center: for a (halfspace) symmetric distribution the center of symmetry is a Tukey median.

3. Monotonicity with respect to the deepest point: if there is a unique Tukey median μ, $HD(x;P)$ decreases as x moves away along a ray from μ.

4. Vanishing at infinity: $HD(x;P) \to 0$ as $\|x\| \to \infty$.

Even though there is not a unique definition of a statistical depth function, these or closely related properties are typically requested for a function to qualify as depth function. Beside Tukey's halfspace depth, prominent examples of depth functions are simplicial depth (Liu, 1988, 1990), majority depth, projection depth, or Mahalanobis depth (Liu, 1992; Zuo and Serfling, 2000). To the best of our knowledge, the lens depth function (Liu and Modarres, 2011) is the only statistical depth function from the literature that can be evaluated given only ordinal distance information about a data set in an arbitrary semimetric space. Note that the function F defined in (19), which the approach by Heikinheimo and Ukkonen (2013) is based on, is provably not a statistical depth function. It does not satisfy the property of maximality at the center for symmetric distributions. Indeed, as Heikinheimo and Ukkonen observe, in case of a symmetric bimodal distribution in one dimension with the two modes sufficiently far apart, the center of symmetry is in fact a minimizer of F.

We provide some references related to our Algorithms 2 and 3: The idea of considering data points with a small depth value as outliers has been thoroughly studied in the setting of a contamination model in Chen et al. (2009) and Dang and Serfling (2010). In particular, they deal with the question of determining what a *small* depth value is.
The simple max-depth approach to binary classification outlined in Section 3.1 has already been proposed by Liu (1990), using simplicial depth instead of the lens depth function.

It has been theoretically studied in Ghosh and Chaudhuri (2005). Ghosh and Chaudhuri were able to prove that the max-depth approach is consistent, that is it asymptotically achieves Bayes risk, for equally probable and elliptically symmetric classes that only differ in location when using one of several depth functions and dealing with general K-class problems. Working not too well when these assumptions are not satisfied, the max-depth approach has been refined by Li et al. (2012) by allowing for more general classifiers on the DD-plot, thus overcoming some of its original limitations. The DD-plot (depth vs. depth plot; introduced by Liu et al., 1999) is the image of the data under the feature map $x \mapsto (DF(x; Class_1), DF(x; Class_2)) \in \mathbb{R}^2$, where DF denotes the depth function under consideration. Interestingly, Li et al. again only consider the 2-class case and propose a one-vs-one approach for the general case, which is different from our strategy of simply considering

$$x \mapsto (DF(x; Class_1), DF(x; Class_2), \ldots, DF(x; Class_K)) \in \mathbb{R}^K$$

as feature map and subsequently performing classification on $\mathbb{R}^K$.

We conclude this section with some comments about the lens depth function. An early version of the lens depth function has already been mentioned, but not seriously studied, by Lawrence (1996, Section 2.3) and by Bartoszynski et al. (1997). The main reference for the lens depth function is Liu and Modarres (2011), where the lens depth function has been defined and systematically investigated. However, after reading the proofs in detail, we found that there is still an important gap. Liu and Modarres (2011) claim that the lens depth function satisfies the property of maximality at the center for centrally symmetric distributions on $\mathbb{R}^m$ (Theorem 6 in their paper). However, there is an error in their proof. It is *not* true that, conditioning on X_1, the probability of X_2 falling into a region such that $t \in Lens(X_1, X_2)$ holds decreases as $t \in \mathbb{R}^m$ moves away from the center for *all* values of X_1, and hence the monotonicity of the integral is not guaranteed. The same mistake appears in Elmore et al. (2006) and in Section 2.5 of Yang (2014) when showing the property for the spherical depth function and the β-skeleton depth function, respectively. So it has not yet been established that the lens depth function satisfies this essential property of statistical depth functions. We were not able to fix the proof, but we still believe that the statement is correct. At least, unlike for the function F defined in (19), we have not been able to construct any example of a symmetric distribution for which the lens depth function does not attain its maximum at the center.

5.3 k-Relative Neighborhood Graph

The k-RNG belongs to the class of proximity graphs: two vertices are connected if they are in some sense close to each other (see Jaromczyk and Toussaint, 1992, for a basic survey or Bose et al., 2012, for a more recent paper). Beside the k-RNG, Gabriel graphs (Gabriel and Sokal, 1969) and k-NN graphs are prominent examples of proximity graphs.

The 1-RNG, which is simply known as RNG, has been used in a wide range of applications (see Toussaint, 2014, for a review and detailed references). Most interesting for us are its use in classification and clustering as related to our Algorithms 4 and 5, respectively:

Instance-based classification based on the RNG neighborhood, that is inferring a point's label by taking a majority vote of the point's neighbors in the RNG, has been empirically shown to be competitive with the k-NN classifier in Sánchez et al. (1997a) and Toussaint and Berzan (2012). Instance-based classification based on the RNG neighborhood has also been used for prototype selection for the 1-NN classifier (Toussaint et al., 1984; Sánchez et al., 1997b). The RNG has been used for spectral clustering in Correa and Lindstrom (2012) with a strategy of assigning locally adapted edge weights. Our experiments in Section 6.1.4 show that such a strategy is dispensable and that using the k-RNG weighted as in (9), or also unweighted, yields reasonable results as well.

We have mentioned in Section 3.2 and Section 4.4 that a true k-RNG (not an estimated one) is always connected. This follows from the fact that the RNG on a data set $\mathcal{D}$ contains the minimal spanning tree on $\mathcal{D}$ as a subgraph. By minimal spanning tree we mean the minimal spanning tree of the complete graph on $\mathcal{D}$ in which an edge is weighted with the distance between two points. A proof of this property for data points in the Euclidean plane, which readily generalizes to data sets in arbitrary semimetric spaces, can be found in Toussaint (1980). The RNG is guaranteed to be sparse for data sets in the 2-dimensional or 3-dimensional Euclidean space, but it can be dense in higher-dimensional spaces or if d is induced by the 1-norm or the maximum norm (Jaromczyk and Toussaint, 1992). There is a large literature on the question how to efficiently compute a k-RNG on a data set, mainly for data sets in $\mathbb{R}^2$ or $\mathbb{R}^3$ (see the references in Toussaint, 2014), and how to *approximate* the RNG by a graph that is easier to compute (Andrade and de Figueiredo, 2001). We are not aware of any work that deals with *estimating* the k-RNG as we do in this paper.

6. Experiments

We performed several experiments for examining the performance of our proposed Algorithms 1 to 5 and compared them to ordinal embedding approaches. In case of Algorithm 1 and Algorithm 2 we also made a comparison with the methods proposed by Heikinheimo and Ukkonen (2013) explained in Section 5.1.1. Recall that an ordinal embedding approach consists of first constructing an ordinal embedding of a data set $\mathcal{D}$ based on the given ordinal distance information and then solving the problem on the embedding by applying a standard algorithm. For example, in the case of medoid estimation a medoid of an ordinal embedding is computed and the corresponding object is returned as an estimate of a medoid of $\mathcal{D}$. For constructing an ordinal embedding we tried several algorithms: the GNMDS (generalized non-metric multidimensional scaling) algorithm by Agarwal et al. (2007), the SOE (soft ordinal embedding) algorithm by Terada and von Luxburg (2014), and the STE (stochastic triplet embedding) and t-STE (t-distributed stochastic triplet embedding) algorithms by van der Maaten and Weinberger (2012). The GNMDS algorithm and the SOE algorithm can take answers to arbitrary dissimilarity comparisons of the form (17) as input, while the STE and t-STE algorithms are designed only for similarity triplets, that is answers to comparisons (18). The ordinal data that we gave to the embedding algorithms were all the similarity triplets obtained via (2) from a collection of statements of the kind ($\star$) that we provided as input to one of our algorithms. We used the Matlab implementations of GNMDS, STE, and t-STE provided by van der Maaten and Weinberger (2012) and the

R implementation of SOE provided by Terada and von Luxburg (2014). We set all parameters except the dimension m of the space of the embedding to the provided default parameters (for all algorithms the default dimension is two). Note that all algorithms try to iteratively minimize an objective function that measures the amount of violated ordinal relationships, and in doing so their results depend on a random initialization of the ordinal embedding.

We start with presenting experiments on artificial data in Section 6.1. In Section 6.2 we deal with real data consisting of 60 images of cars and ordinal distance information of the kind ($\star$) that we have collected via crowdsourcing in an online survey.

6.1 Artificial Data

In the following, except the plots in Figures 8 and 9, where outliers have to be identified by visual inspection, and one plot in Figure 5, which provides a visualization of available statements per data point, all plots of this section show results averaged over running the experiments for 100 times.

We primarily study the performance of the considered methods with respect to the number of provided input statements, but also with respect to the amount of noise in the provided ordinal data. We consider two different noise models: Noise model I (with parameter $0 \leq errorprob \leq 1$) equals the one described in Section 3.2.1, that is a statement of the kind ($\star$) is incorrect, independently of other statements, with some fixed error probability $errorprob$. In an incorrect statement the two data points that are not most central appear to be most central with probability 1/2 each. In Noise model II (with parameter $noiseparam \geq 0$) we distort the dissimilarity values $d(A, B)$, which then induces a distortion of statements. Concretely, we add Gaussian noise with mean zero and standard deviation $noiseparam \cdot SD$, where SD denotes the standard deviation of all true dissimilarity values $d(A, B)$, $A \neq B \in \mathcal{D}$, independently to each dissimilarity value $d(A, B)$. For choosing input statements we essentially consider two sampling strategies: The first one, referred to as uniform sampling, is to choose input statements uniformly at random without replacement from the set of all statements, that is the set of statements for all triples of data points, which were generated according to the noise model under consideration. When applying this sampling strategy and studying performance as a function of the number of input statements, the rightmost measurement in a plot corresponds to the case that all statements are provided as input. In the experiment presented in Figure 7 the provided statements are chosen uniformly at random with replacement from the set of all statements, but there the set of all statements is so large that in fact this does not make any difference. In these plots the rightmost measurement corresponds to a number of input statements of less than one permil of the number of all statements. In order to illustrate our claim that our algorithms require statements to be sampled only approximately uniformly with respect to a fixed data point (Algorithms 1 to 3), or a fixed pair of data points (Algorithms 4 and 5), we also consider a second sampling strategy, referred to as Sampling II. When sampling according to this strategy, we partition the data set into ten groups. For each group we form a set consisting of all statements, generated according to the noise model under consideration, that comprise at least one data point from the corresponding group. We then sample with replacement by selecting one of the ten sets according to probabilities $i^2 / \sum_{j=1}^{10} j^2$, $i = 1, \ldots, 10$, and choosing a statement from the selected set uniformly at random.
When comparing Algorithm 1 or Algorithm 2 to the corresponding methods by Heikinheimo and Ukkonen (2013) in Sections 6.1.1 and 6.1.2, their methods are given a collection of statements of the kind ($\boxplus$) as input that contains as many statements as the input to our algorithm and is created in a completely analogous way.

6.1.1 Medoid Estimation

We measure performance of a method for medoid estimation by the relative error in the objective D (given in (6)), which is given by

$$\text{relative error} = \frac{D(\text{estimated medoid}) - D(\text{true medoid})}{D(\text{true medoid})}. \tag{20}$$

Figure 5 shows in the first two rows the relative error of Algorithm 1, the method by Heikinheimo and Ukkonen (2013), and the embedding approach, using the various embedding algorithms, as a function of the number of provided input statements and as a function of $errorprob$ (Noise model I) for 100 points from a 2-dimensional Gaussian $N_2(0, I_2)$ and d being the Euclidean metric. Obviously, the embedding approach outperforms Algorithm 1 and the method by Heikinheimo and Ukkonen when dealing only with correct statements, that is $errorprob = 0$, and embedding into the true dimension (1st row, 1st plot). However, it is not superior over Algorithm 1 anymore when $errorprob = 0.3$ and the dimension of the embedding is chosen as five (2nd row, 1st plot). Algorithm 1 consistently outperforms the method by Heikinheimo and Ukkonen. All methods show a similar behavior with respect to $errorprob$ (2nd row, 2nd & 3rd plot). Interestingly, the strongest incline in the error does not occur until the transition from $errorprob = 0.6$ to $errorprob = 0.7$. The bottom row of Figure 5 also shows the relative error of the various methods as a function of the number of provided input statements, but here input statements were sampled according to the strategy Sampling II. Compared to the strategy of sampling statements uniformly at random without replacement from the set of all statements, Algorithm 1 performs slightly worse, but we consider the difference to be negligible. The last plot of the bottom row shows the difference in the two sampling strategies: while in the uniform case, for all data points there is almost the same number of input statements comprising the data point, when sampling according to Sampling II there are data points for which this number is twice as large as for others (the plot is based on a total of 4500 input statements corresponding to the third measurement in the first and second plot of the bottom row).
The biggest advantage of Algorithm 1 (in fact of *all* our proposed algorithms) compared to an ordinal embedding approach becomes obvious from the plots in the third and fourth row of Figure 5, which show the running times of the experiments shown in the plots in the two top rows: For a fixed size $|\mathcal{D}|$ of the data set, like the running times of our proposed algorithms and the method by Heikinheimo and Ukkonen, the running time of the embedding approach with any of the considered embedding algorithms also grows linearly with the number $|\mathcal{S}|$ of input statements (indicated by the orange curves). However, in practice Algorithm 1 and the method by Heikinheimo and Ukkonen are vastly superior in

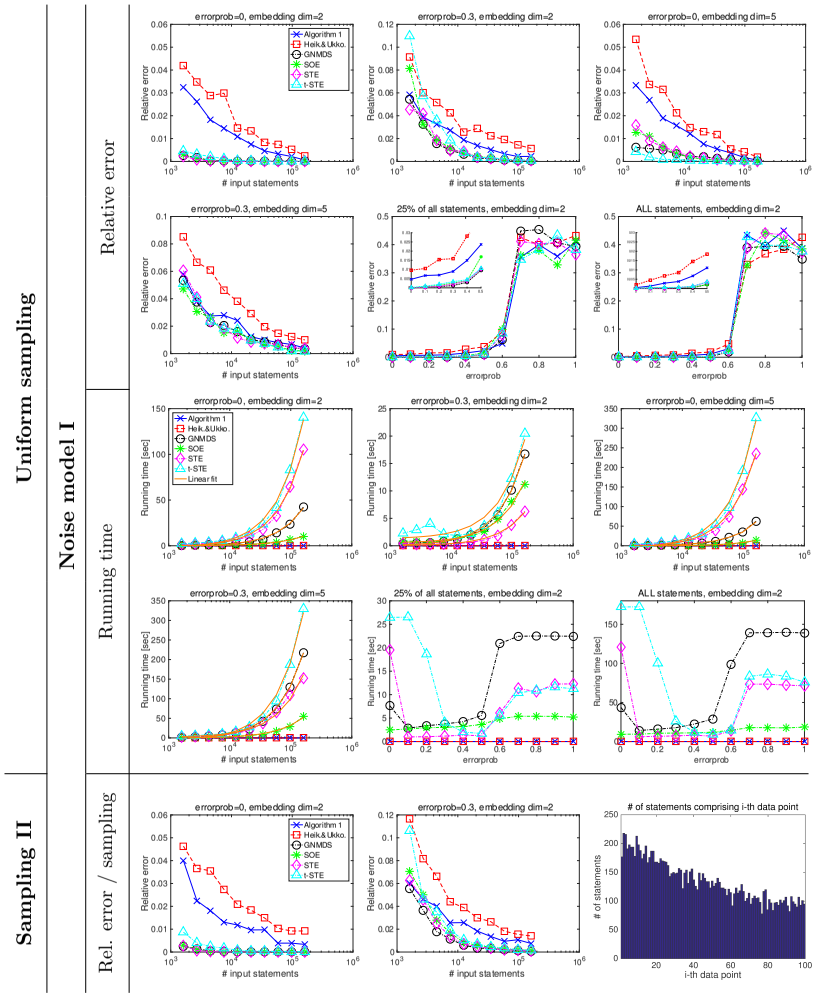


Figure 5: Medoid estimation — 100 points from a 2-dim Gaussian $N_2(0, I_2)$ with Euclidean metric. Relative error (20) and running time as a function of the number of provided statements of the kind ($\star$) or of the kind ($\boxplus$) and as a function of *errorprob* for Algorithm 1, for the method by Heikinheimo and Ukkonen, and for the embedding approach using the various embedding methods.

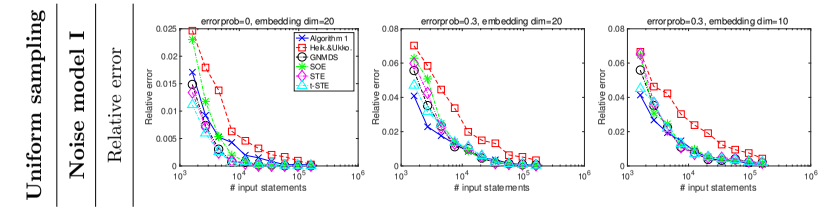


Figure 6: Medoid estimation — 100 points from a 20-dim Gaussian $N_{20}(0, I_{20})$ with Euclidean metric. Relative error (20) as a function of the number of provided statements of the kind ($\star$) or of the kind ($\boxplus$) for Algorithm 1, for the method by Heikinheimo and Ukkonen, and for the embedding approach using the various embedding methods.

terms of running time compared to the embedding approach, even without making use of their potential of simple and highly efficient parallelization. For example, when all statements are provided as input, *errorprob* $= 0$, and the embedding dimension is chosen as two, the running time of the embedding approach is between 10 seconds (when using the SOE algorithm) and 141 seconds (when using the t-STE algorithm), while Algorithm 1 or the method by Heikinheimo and Ukkonen only run for 0.01 seconds (3rd row, 1st plot). Note that the running times of Algorithm 1 and the method by Heikinheimo and Ukkonen are independent of *errorprob* and, of course, of the choice of a dimension of the space of the embedding. The running times of the embedding algorithms tend to increase with the embedding dimension (e.g., differences between the first and the third plot in the third row). The running time of the SOE algorithm also increases with *errorprob* (4th row, 2nd & 3rd plot). For the GNMDS algorithm this holds for *errorprob* ≥ 0.1. The running times of the STE and t-STE algorithms vary non-monotonically with *errorprob*. All experiments shown in Figure 5 were performed in MATLAB R2015a on a MacBook Pro with 2.6 GHz Intel Core i7 and 8 GB 1600 MHz DDR3. Within MATLAB we invoked R 3.2.2 for computing the SOE embedding. In order to make a fair comparison we did not use MEX files in the implementation of Algorithm 1 or the method by Heikinheimo and Ukkonen.

Figure 6 shows almost the same experiments as Figure 5, but this time dealing with 100 points from a 20-dimensional Gaussian $N_{20}(0, I_{20})$. The distance function d again equals the Euclidean metric. In this high-dimensional case the embedding approach cannot be considered superior anymore. In fact, when *errorprob* $= 0.3$ and the number of input statements is small, Algorithm 1 performs best (2nd plot). We omit to show plots of the relative error as a function of *errorprob* since they look very similar to the ones in Figure 5. Time measurements show that the differences in running times between the embedding approach and Algorithm 1 or the method by Heikinheimo and Ukkonen are even more severe compared to Figure 5, as to be expected because of the high embedding dimensions (plots omitted).

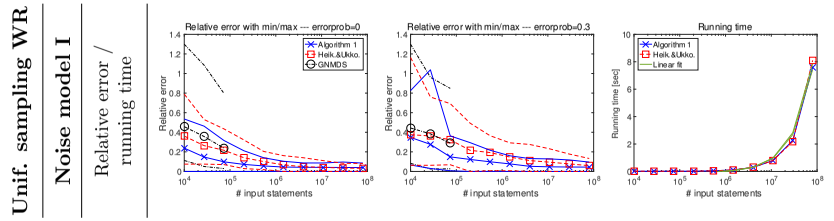


Figure 7: Medoid estimation — 8638 vertices in a collaboration network with shortest-path-distance. 1st & 2nd plot: Relative error (20) as a function of the number of provided statements of the kind ($\star$) or of the kind ($\boxplus$) for Algorithm 1, for the method by Heikinheimo and Ukkonen, and for the embedding approach using GNMDS (for the first three measurements). Average over 100 runs together with the minimum and the maximum of the 100 runs. 3rd plot: The corresponding running times with fitted linear functions.

Finally, we applied Algorithm 1 and the method by Heikinheimo and Ukkonen to a large network with the dissimilarity function d equaling the shortest-path-distance. In this context, a medoid is usually referred to as a "most central point with respect to the closeness centrality measure" (Freeman, 1978). Our data set consists of 8638 vertices, which form the largest connected component of a collaboration network with 9877 vertices that represent authors of papers submitted to arXiv in the High Energy Physics - Theory category and with two vertices being connected if the authors co-authored at least one paper (Leskovec et al., 2007). Comparing against the embedding methods as in the previous experiments on this large data set would have taken months (considering various numbers of input statements and averaging over 100 runs), so we only compared against GNMDS (embedding dimension chosen to equal two) for a small number of input statements. The first and the second plot of Figure 7 show the relative error of Algorithm 1 and the method by Heikinheimo and Ukkonen as a function of the number of provided statements of the kind ($\star$) or of the kind ($\boxplus$). The number of provided statements varies between 10^4 and $8 \cdot 10^7$. The latter is less than one permil of the number of all statements, which is $\binom{8638}{3} \approx 10^{11}$. This number is so large that the set of all statements does by no means fit into the main memory of a single machine. The plots also show the relative error of the embedding approach using the GNMDS algorithm for 10000, 27144, and 73680 input statements. As in the previous experiments, the shown error is the average over 100 runs of the experiment, but here the data set is fixed and the only sort of randomness comes from the input statements (and the random initialization of the ordinal embedding in case of GNMDS). In addition to the average error the plots show the minimum and maximum error of the 100 runs for illustrating the variance in the methods. In both the cases of $errorprob = 0$ (1st plot) and $errorprob = 0.3$ (2nd plot), when the number of input statements is small, Algorithm 1 outperforms the method by Heikinheimo and Ukkonen. Both methods outperform the embedding approach, which might have difficulties due to the data set being non-Euclidean or might struggle with a too small embedding dimension. For comparison, a strategy of

choosing a data point uniformly at random as medoid estimate incurs a relative error of 0.47 in expectation. Even when given only 10000 input statements, when $errorprob = 0$, the error of Algorithm 1 is only about one half of this. The variance seems to be similar for both Algorithm 1 and the method by Heikinheimo and Ukkonen and seems to be significantly larger for the embedding approach. As expected, it decreases as the number of input statements increases. In case of $errorprob = 0$, we also applied GNMDS to the data set providing 10857670 statements as input (corresponding to the eighth measurement in the plots): averaging over 10 runs we obtained an average relative error of 0.28 (which is more than six times larger than the error of Algorithm 1 or the method by Heikinheimo and Ukkonen), where computation took 2.84 hours on average. The third plot of Figure 7 shows the running times of Algorithm 1 and the method by Heikinheimo and Ukkonen as a function of the number of input statements. Both methods have the same running time, which is linear in the number of input statements. The plot does not show the running times of GNMDS at the first three measurements. These were 110, 860, and 879 seconds in case of $errorprob = 0$ and 92, 848, and 898 seconds in case of $errorprob = 0.3$.

6.1.2 Outlier Identification

We started with testing Algorithm 2 and the corresponding method by Heikinheimo and Ukkonen (2013) by applying them to two visualizable data sets containing some obvious outliers. Both of the Figures 8 and 9 show a scatterplot of the points of a data set $\mathcal{D}$ in the Euclidean plane with the "regular" points in black and the outliers in color. For assessing the performance of the two considered methods we plotted the sorted values of $LD(O)$, $O \in \mathcal{D}$, as needed for Algorithm 2 as well as the sorted values of estimated probabilities of being an outlier within a triple of objects as needed for the method by Heikinheimo and Ukkonen (compare with Section 5.1.1). Both methods were provided with the same number of statements as input, either of the kind ($\star$) or of the kind ($\boxplus$). Both Figure 8 and Figure 9 provide several such plots, varying with this number of input statements as well as with the error probability $errorprob$ (we generated statements according to Noise model I). In all the plots, values belonging to outliers have the same color as the corresponding outlier in the scatterplot. The methods are successful if these colored values appear at the very end of the sorted values, either at the lower end for Algorithm 2 or at the upper end for the method by Heikinheimo and Ukkonen, and there is a (preferably large) gap between the colored values and the remaining ones since then it is easy to correctly identify the outliers. There are inlay plots showing the bottom or top ten values for more precise inspection. Note that there is no averaging involved in creating these plots and they may change with every run of the experiment since they depend on the random data set, the random choice of statements that are provided as input, and the random occurrence of incorrect statements.

In Figure 8 the data set consists of 100 points that were drawn from a 2-dimensional Gaussian $N_2(0, I_2)$ and three outliers added by hand. The dissimilarity function d equals the Euclidean metric. We can see that for both methods the values corresponding to the outliers appear at the right place when given all correct statements as input (top row). However, when given only 25 percent of all statements and $errorprob = 0.3$, for Algorithm 2 the estimated lens depth value of the pink outlier ranks only sixth smallest, and thus this outlier

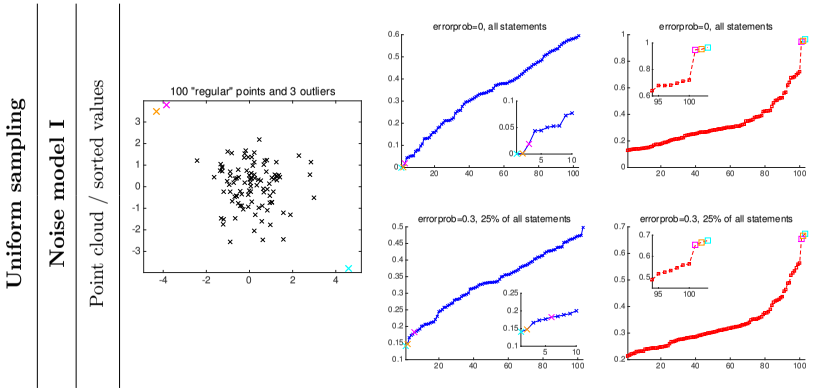


Figure 8: Outlier identification — 100 points from a 2-dim Gaussian $N_2(0, I_2)$ and three outliers added by hand with Euclidean metric. Data set and sorted values of $LD(O)$ as needed for Algorithm 2 (left; in blue) and of estimated probabilities as needed for the method by Heikinheimo and Ukkonen (right; in red).

might not be identified (bottom left). Furthermore, even in the previous situation it might not be possible to correctly infer the number of outliers based on the plot corresponding to Algorithm 2 due to the lack of a clear gap, whereas in both situations this can easily be done for the method by Heikinheimo and Ukkonen. We made similar observations for smaller numbers of provided input statements and other values of *errorprob* too (plots omitted).

In Figure 9 the data set consists of 200 points from a Two-moons data set and four outliers added by hand. Again, d equals the Euclidean metric. Both methods correctly identify the three outliers located quite far apart from the bulk of the data points, and the gap between their values and values belonging to the "regular" data points is large enough to be easily spotted. However, both methods fail to identify the outlier located in-between the two moons (yellow point). The estimated lens depth values or probabilities indicate that this outlier might be the unique medoid—which is indeed the case. For Algorithm 2 this has to be expected and stresses the inherent property of the lens depth function, and statistical depth functions in general, of globally measuring centrality. In doing so, it ignores multimodal aspects of the data (compare with Section 5.2 and Section 7) and cannot be used for identifying outliers that are globally seen at the heart of a data set. At least for the data set of Figure 9 this also holds for the function F defined in (19), which the method by Heikinheimo and Ukkonen is based on. However, for the function F this behavior is not systematic as the example of a symmetric bimodal distribution in one dimension as mentioned in Section 5.2 shows.

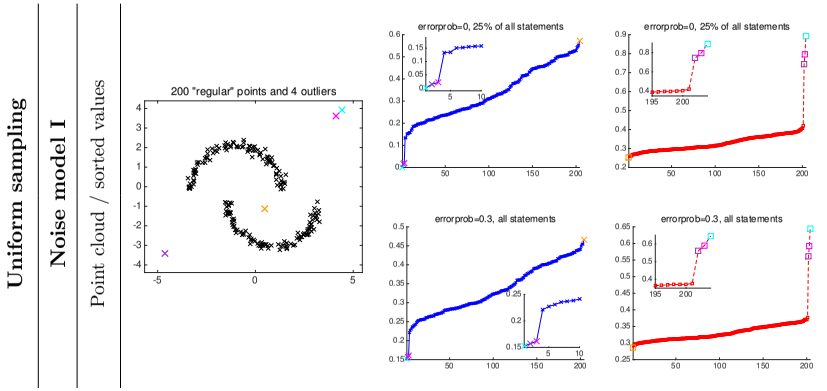


Figure 9: Outlier identification — 200 points from a Two-moons data set and four outliers added by hand with Euclidean metric. Data set and sorted values of $LD(O)$ as needed for Algorithm 2 (left; in blue) and of estimated probabilities as needed for the method by Heikinheimo and Ukkonen (right; in red).

In the last experiment of this section we study Algorithm 2 and the method by Heikinheimo and Ukkonen by using them for outlier identification in a data set consisting of USPS digits. The data set consists of 500 digits chosen uniformly at random from digits 6 and ten outlier digits chosen uniformly at random from the remaining digits. The dissimilarity function d equals the Euclidean metric. We assess the performance of Algorithm 2 and the method by Heikinheimo and Ukkonen by counting how many of the ten outliers are among the ten digits ranked lowest or highest according to the values of $LD(O)$ and estimated probabilities, respectively. Figure 10 shows these numbers as a function of the number of provided input statements in case of uniform sampling and statements generated according to Noise model I (1st row) and in case of Sampling II and statements generated according to Noise model II (2nd row), for $errorprob = 0$ / $noiseparam = 1$ (1st plot), $errorprob = 0.1$ / $noiseparam = 1.5$ (2nd plot), and $errorprob = 0.3$ / $noiseparam = 2$ (3rd plot). We can see that the method by Heikinheimo and Ukkonen performs slightly better in the setting of the first row and that the performance of both methods is essentially the same in the setting of the second row. Most often, the methods can identify three to five outliers, which we consider to be not bad, but not good either. Choosing another digit than 6 for defining the bulk of "regular" points leads to similar results (plots omitted).

To sum up the insights from the experiments shown in Figures 8 to 10, we may conclude that both methods are capable of identifying outliers located lonely and far apart from the bulk of a data set, but should be used with some care in general. The method by Heikinheimo and Ukkonen seems to be superior—which is not very surprising since statements of the

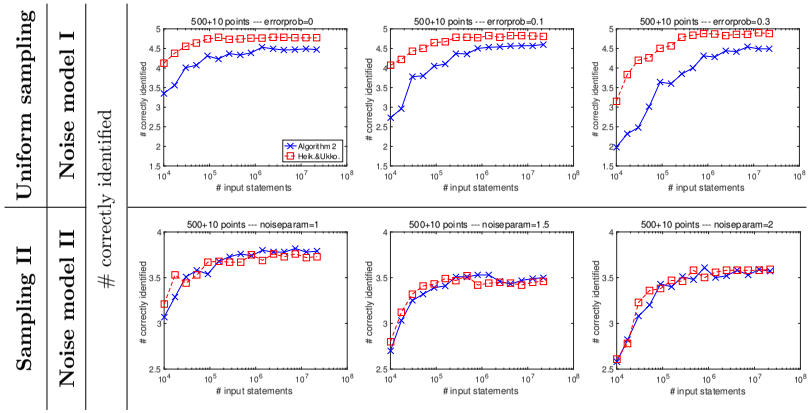


Figure 10: Outlier identification — 500 points from the subset of USPS digits 6 and ten outlier digits with Euclidean metric. Number of correctly ranked outliers as a function of the number of provided statements of the kind ($\star$) or of the kind ($\boxplus$) for Algorithm 2 and for the method by Heikinheimo and Ukkonen.

kind ($\boxplus$) readily inform about outliers within triples of data points. It produces larger and thus easier to spot gaps than Algorithm 2, but is less understood theoretically.

6.1.3 Classification

We compared Algorithms 3 and 4 to an ordinal embedding approach that consists of embedding a data set $\mathcal{D}$ comprising a set $\mathcal{L}$ of labeled data points and a set $\mathcal{U}$ of unlabeled data points into $\mathbb{R}^m$ using the given ordinal distance information and applying a classification algorithm to the embedding. Note that this approach is semi-supervised since it makes use of answers to dissimilarity comparisons involving data points of $\mathcal{U}$ for constructing the embedding of $\mathcal{D}$. Algorithm 3, in contrast, only uses ordinal distance information involving data points of $\mathcal{L}$ for approximately evaluating the feature map (7) on $\mathcal{L}$ and hence is a supervised technique as long as the classifier on top is. Algorithm 4 is a supervised instance-based learning method. Algorithm 3 as well as the embedding approach require an ordinary classifier on top, that is a classifier appropriate for real-valued feature vectors. For simplicity, in the experiments presented here we either used the k-NN classifier or the SVM (support vector machine) algorithm with the standard linear kernel (e.g., Cristianini and Shawe-Taylor, 2000). Both these classification algorithms require to set parameters, which we did by means of 10-fold cross-validation: the parameter k for the k-NN classifier was chosen from the range $1, 3, 5, 7, 11, 15, 23$ and the regularization parameter for the SVM algorithm was chosen from $0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100, 500, 1000$. The ordinal embedding algorithms produce embeddings on an arbitrary scale. Before applying the classification algorithms, we rescaled an ordinal embedding to have diameter 2. The feature embedding

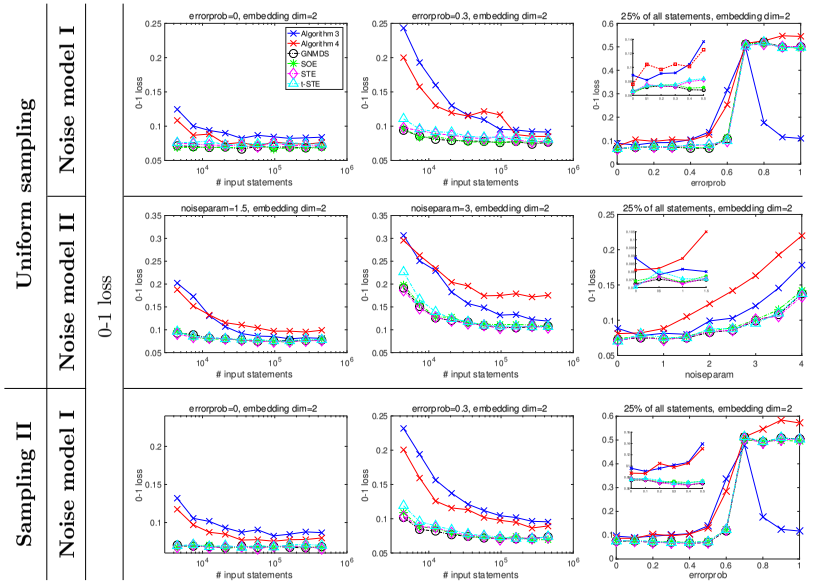


Figure 11: Classification — 100 labeled and 40 unlabeled points from a mixture of two equally probable 2-dim Gaussians $N_2(0, I_2)$ and $N_2((3, 0)^T, I_2)$ with Euclidean metric. SVM algorithm with linear kernel on top of Algorithm 3 as well as on the embedding approach. 0-1 loss (21) as a function of the number of provided statements of the kind ($\star$) and as a function of *errorprob* / *noiseparam* for Algorithm 3, for Algorithm 4, and for the embedding approach using the various embedding methods.

constructed by Algorithm 3 always resides in $[0, 1]^K$ for a K-class classification problem and no rescaling was done here. Algorithm 4 requires to set the parameter k describing which k-RNG it is based on, but this is more subtle: As we have seen in Section 3.2.1, when input statements are incorrect with some error probability $errorprob > 0$ (Noise model I), then our estimation strategy does not estimate the k-RNG anymore, but rather a k'-RNG with $k' = k'(k, errorprob, |\mathcal{D}|)$ depending on the size of the data set as given in (15). We thus have to choose the range of possible values for the parameter k in Algorithm 4 depending on $|\mathcal{D}|$. Furthermore, we cannot use 10-fold cross-validation for choosing the best value within this range since, roughly speaking, this would lead to choosing the best parameter for a data set of size of only 90 percent of $|\mathcal{D}|$. Instead, we used a non-exhaustive variant of leave-one-out cross-validation: we randomly selected a single training point as validation set and repeated this procedure for 20 times, and finally chose the parameter that showed the best performance on average.

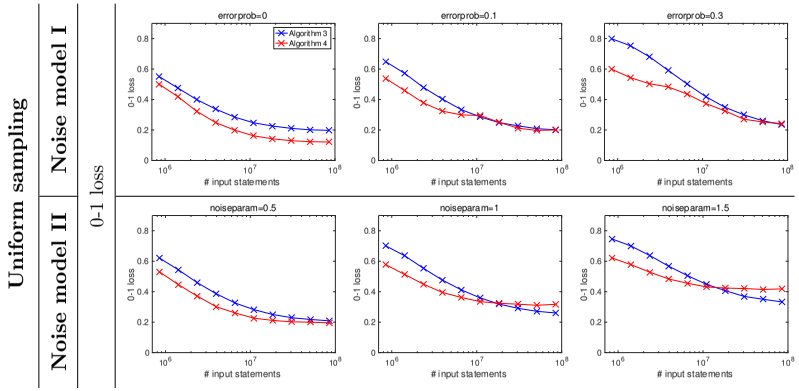


Figure 12: Classification — 300 labeled and 500 unlabeled USPS digits with Euclidean metric. k-NN classifier on top of Algorithm 3. 0-1 loss (21) as a function of the number of provided statements of the kind ($\star$) for Algorithm 3 and for Algorithm 4.

We measure performance of Algorithms 3 and 4 and the embedding approach by considering their incurred 0-1 loss given by

$$\text{0-1 loss} = \frac{1}{|\mathcal{U}|} \cdot \sum_{O \in \mathcal{U}} \mathbb{1}\{\text{predicted label}(O) \neq \text{true label}(O)\}. \tag{21}$$

Figure 11 shows the results for a data set consisting of 100 labeled and 40 unlabeled points from a mixture of two equally probable 2-dimensional Gaussians $N_2(0, I_2)$ and $N_2((3,0)^T, I_2)$ and d being the Euclidean metric. True class labels of the points correspond to which Gaussian they come from. On top of Algorithm 3 as well as on the embedding methods we used the SVM algorithm with the linear kernel. The parameter k for Algorithm 4 was chosen from the range $1, 2, 3, 5, 7, 15, 25, 45, 70$. The dimension of the space of the ordinal embedding was chosen to equal the true dimension two, but we observed similar results when we chose it as five instead (plots omitted). The embedding approach outperforms both Algorithm 3 and Algorithm 4, but their results appear to be acceptable too. Interestingly, other than for Algorithm 4 and the embedding approach, the 0-1 loss incurred by Algorithm 3 studied as a function of *errorprob* (1st & 3rd row, 3rd plot) increases only up to *errorprob* $= 0.7$ and then drops again, finally yielding almost the same result for *errorprob* $= 1$ as for *errorprob* $= 0$. In hindsight, this is not surprising: If *errorprob* $= 1$, and thus every statement is incorrect, and the two possibilities of an incorrect statement are equally likely, as it is the case under Noise model I, then Algorithm 3 approximately evaluates the feature map

$$x \mapsto \left(\frac{1}{2} - \frac{1}{2}LD(x; Class_1), \frac{1}{2} - \frac{1}{2}LD(x; Class_2), \ldots, \frac{1}{2} - \frac{1}{2}LD(x; Class_K)\right) \in \mathbb{R}^K.$$

This feature map coincides with the original one given in (7) up to a similarity transformation and hence gives rise to the same classification results.

In Figure 12 we study the performance of Algorithms 3 and 4 when used for classifying USPS digits. We deal with 800 digits chosen uniformly at random from the set of all USPS digits and randomly split into 300 labeled and 500 unlabeled data points. The dissimilarity function d equals the Euclidean metric. We chose input statements uniformly at random without replacement from the set of all statements, which we generated according to Noise model I (1st row) or Noise model II (2nd row). On top of Algorithm 3 we used the k-NN classifier. The parameter k for Algorithm 4 was chosen from $1, 2, 3, 5, 7, 15, 25, 45, 70, 100, 150, 230, 350$. For small values of *errorprob* or *noiseparam* we consider the results of our proposed algorithms to be satisfactory and useful. Note that in this 10-class classification problem a strategy of random guessing would yield a 0-1 loss of about 0.9. Not surprisingly, we obtained slightly better results when the ratio between labeled and unlabeled data points was chosen as 400/400 instead of 300/500 and slightly worse results when it was chosen as 200/600 (plots omitted).

6.1.4 Clustering

We compared Algorithm 5, both in its weighted and in its unweighted version, to an embedding approach in which we applied spectral clustering to a symmetric k-NN graph on an ordinal embedding of a data set $\mathcal{D}$. We put Gaussian weights $\exp(-\|u_i - u_j\|^2/\sigma^2)$, where u_i and u_j are connected points of the embedding and $\sigma > 0$ is a scaling parameter, on the edges of this k-NN graph. Again, we rescaled an ordinal embedding to have diameter 2. Both in Algorithm 5 and in the ordinal embedding approach we used the normalized version of spectral clustering as stated in von Luxburg (2007) and invented by Shi and Malik (2000). For assessing the quality of a clustering we measure its purity with respect to a ground truth partitioning of the data set $\mathcal{D}$ (e.g., Manning et al., 2008, Chapter 16): if $\mathcal{D}$ consists of L different classes $C_1, \ldots, C_L$ that we would like to recover and the clustering $\mathcal{C}$ comprises K different clusters $U_1, \ldots, U_K$, then the purity of $\mathcal{C}$ is given by

$$\text{purity}(\mathcal{C}) = \text{purity}(\mathcal{C}; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{k=1}^{K} \max_{l=1,..,L} |U_k \cap C_l|. \tag{22}$$

We always have $K/|\mathcal{D}| \leq \text{purity}(\mathcal{C}) \leq 1$, and a high value indicates a good clustering. In the experiments presented in this section, we always provided Algorithm 5 and the embedding approach with the correct number L of clusters as input.

Figure 13 shows the purity of the clusterings produced by Algorithm 5 and the embedding approach when applied to a data set consisting of 100 points from a uniform distribution on two equally sized moons in $\mathbb{R}^2$. The two moons correspond to a ground truth partitioning into two classes. The dissimilarity function d equals the Euclidean metric. The curves shown here are the results obtained by a particular choice of input parameters k and σ: within a reasonably large range of parameter configurations this choice of parameters yielded the best performance on average with respect to the number of input statements. We study the sensitivity of Algorithm 5 with respect to the parameters in another experiment (shown

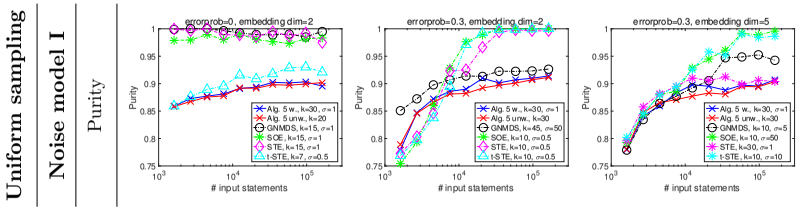


Figure 13: Clustering — 100 points from a uniform distribution on two equally sized moons in $\mathbb{R}^2$ with Euclidean metric. Purity (22) as a function of the number of provided statements of the kind ($\star$) for Algorithm 5 in its weighted and unweighted version and for the embedding approach using the various embedding methods.

in Figure 14). The embedding approach clearly outperforms Algorithm 5 if *errorprob* $= 0$ (1st plot), where three of the considered embedding algorithms achieve significantly higher purity values over the whole range of the number of input statements. However, given the numerous advantages common to all our proposed algorithms compared to an ordinal embedding approach, we consider the performance of Algorithm 5 to be acceptable. For comparison, a random clustering in which data points are randomly assigned to one of two clusters independently of each other with probability one half has an average purity of 0.54. If *errorprob* $= 0.3$, the embedding approach is superior to Algorithm 5 only if the number of input statements is large. Interestingly, there is almost no difference in the performance of the weighted and the unweighted version of Algorithm 5.

The experiment shown in Figure 14 deals with a data set $\mathcal{D}$ consisting of 600 digits chosen uniformly at random from the set of all USPS digits and d being the Euclidean metric. We assume a ground truth partitioning of $\mathcal{D}$ into ten classes according to the digits' values. For various parameter configurations the plots show the purity of the clusterings produced by the weighted (in blue) and unweighted (in red) version of Algorithm 5 as a function of the number of input statements. The plots also show the purity of the clusterings obtained when applying spectral clustering to the true weighted (in light blue) and unweighted (in bronze) k-RNG on $\mathcal{D}$. Note that these two curves only vary with the number of input statements because of random effects in the K-means step of spectral clustering. Although it might look odd at a first glance that the purity achieved by Algorithm 5 is not monotonic with respect to the number of input statements, we can see that the purity is always between 0.67 and 0.7 for a wide range of values of k and σ when *errorprob* $= 0$ (1st row; 2nd row, 1st plot). For comparison, a random clustering in which data points are randomly assigned to one of ten clusters independently of each other with probability one-tenth has an average purity of 0.19. A clustering obtained by applying spectral clustering to a symmetric k-NN graph with Gaussian edge weights $\exp(-d(x_i, x_j)^2/\sigma^2)$ on $\mathcal{D}$ (the true data set—not an ordinal embedding) has an average purity of not higher than 0.75, even for a good choice of k and σ. When *errorprob* $= 0.3$, Algorithm 5 completely fails for small values of k (2nd row, 2nd plot) as has to be expected because of our findings in Section 3.2.1. For k sufficiently large

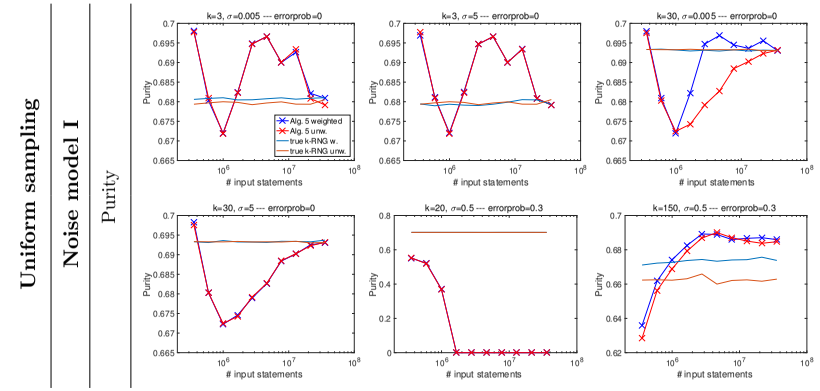


Figure 14: Clustering — 600 USPS digits with Euclidean metric. Purity (22) as a function of the number of provided statements of the kind ($\star$) for Algorithm 5 in its weighted and unweighted version. The light blue curve and the bronze curve show the purity of the clusterings obtained by applying spectral clustering to the true weighted and unweighted k-RNG on the data set.

it finally yields the same purity values as when *errorprob* $= 0$ (2nd row, 3rd plot). In fact, this is true already for $k = 100$ and a wide range of values of σ (plots omitted). Again, both in the case of *errorprob* $= 0$ and in the case of *errorprob* $= 0.3$, there is almost no difference in the performance of the weighted and the unweighted version of Algorithm 5.

6.2 Real Data

We set up an online survey for collecting ordinal distance information of the kind ($\star$) for 60 images of cars, shown in Figure 15. All images were found on Wikimedia Commons (`https://commons.wikimedia.org`) and have been explicitly released into the public domain by their authors. We refer to the set of these images as the car data set. We instructed participants of the survey to determine the most central object within a triple of three shown images according to how they perceive dissimilarity between *cars*. We explicitly stated that they should not judge differences between the *pictures*, like perspective, lighting conditions, or background. Every participant was shown triples of cars in random order (more precisely, triples shown to a participant were drawn uniformly at random without replacement from the set of all possible triples). Also the order of cars within a triple, that is whether a car's image appeared to the left, in the middle, or to the right, was random. One complete round of the survey consisted of 50 shown triples, but we encouraged participants to contribute more than one round, possibly at a later time. There was no possibility of skipping triples, that is even if a participant had no idea which car might be the most central one in a triple, he/she had to make a choice—or quit the current round of the survey. Within the first ten triples every participant was shown a test case triple (shown



Figure 15: Car data set. We collected ordinal distance information of the kind ($\star$) for this data set via an online survey. The framed triple in the first row was used as a test case: T_All and T_All_reduced comprise only statements provided by participants that chose the off-road vehicle as the most central car in this triple. The pictures were found on Wikimedia Commons and have been explicitly released into the public domain by their authors.

within a frame in Figure 15), consisting of an off-road vehicle, a sports car, and a fire truck. We believe the off-road vehicle to be the obvious most central car in this triple and used this test case for checking whether a participant might have got the task of choosing a most central object correctly. The survey was online for about two months and the link to the survey was distributed among colleagues and friends. We took no account of rounds of the survey that were quitted before 30 triples (of the fifty per round) were shown. In doing so, we ended up with 146 rounds (some of them not fully completed) and a total of 7097 statements. It is hard to guess how many different people contributed to these 146 rounds, but assuming an average of three to four rounds per person, which seems to be reasonable according to personal feedback, their number should be around 40. In only 7 out of the 146 rounds the off-road vehicle was not chosen as most central car in the test case triple. We refer to the collection of the total of 7097 statements as the collection All and to its subcollection comprising 6757 statements gathered in the 139 rounds in which the off-road vehicle was chosen as most central car in the test case triple as the collection T_All. From All and T_All we derived two more collections of statements of the kind ($\star$) for the car data set as follows: All_reduced is obtained from All by replacing all statements dealing with the same triple of cars by just one statement about this triple, with the most central car being that car that is most often the most central car in the statements to be replaced. T_All_reduced is derived from T_All analogously. The characteristic values of the collections All, All_reduced, T_All, and T_All_reduced are summarized in Table 1. All survey data and these four collections can be downloaded along with the car data set from `http://www.tml.cs.uni-tuebingen.de/team/luxburg/code_and_data`.

	All	All_reduced	T_All	T_All_reduced
Number of statements	7097	6338	6757	6056
Number of statements in percent of number of triples [$\binom{60}{3} = 34220$]	20.74	18.52	19.75	17.70
Average number of statements in which a car appears	354.85	316.90	337.85	302.80
Minimum number of statements in which a car appears	307	286	292	269
Maximum number of statements in which a car appears	503	347	478	333
Median response time per shown triple (in seconds)	4.02		4.15	

Table 1: Characteristic values of All, All_reduced, T_All, and T_All_reduced. Note that All and T_All contain repeatedly present and contradicting statements.

We applied Algorithms 1 to 5 to the car data set and the statements in All, All_reduced, T_All, or T_All_reduced. In doing so, we assumed a partitioning of the car data set into four subclasses: ordinary cars, sports cars, off-road/sport utility vehicles, and outliers. We considered the fire truck, the motortruck, the tractor, and the antique car as outliers. Looking at Figure 15, there should be no doubts about the other classes.

6.2.1 Medoid Estimation

We applied Algorithm 1 to the car data set as well as to the three classes of ordinary cars, sports cars, and off-road/sport utility vehicles in order to estimate a medoid within these subclasses with the statements in All, All_reduced, T_All, or T_All_reduced. The estimated medoids obtained when working with All or T_All coincide and are shown in Figure 16. The estimated medoids obtained when working with All_reduced or T_All_reduced differ from these only for the whole car data set and the subclass of off-road/sport utility vehicles. Note that for estimating a medoid of a subset of a data set we consider only statements dealing with three objects of the subset. For example, when estimating a medoid of the subclass of sports cars based on the statements in All, we effectively work with 89 out of the 7097 statements in All.

It is interesting to study an ordinal embedding of the car data set. Figure 18 shows an ordinal embedding in the two-dimensional plane that we computed with the SOE algorithm based on the statements in T_All. We cannot only see a grouping of the cars according to the subclasses (compare with Section 6.2.4) and the outer positioning of the outliers (compare with in Section 6.2.2), but also that our medoid estimates are located quite at the center of the corresponding subclasses (with the exception of the subclass of off-road/sport utility vehicles). This confirms the plausibility of our estimates. Note that we observe slightly different embeddings depending on the random initialization in the SOE algorithm.



Figure 16: The estimated medoids for the car data set and the subclasses of ordinary cars, sports cars, and off-road/sport utility vehicles when working with the statements in All or T_All.

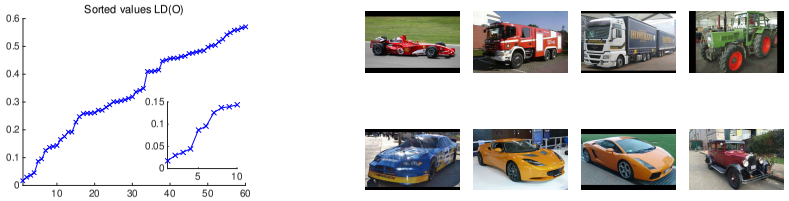


Figure 17: The sorted values $LD(O)$ (O in car data set) as well as the eight cars with smallest values (increasingly ordered) when working with the statements in T_All.

Also, it is not useful to compare the medoid estimates of Algorithm 1 with estimates based on an ordinal embedding since the latter change with every run of the embedding algorithm.

6.2.2 Outlier Identification

We applied Algorithm 2 to the car data set and the statements in All, All_reduced, T_All, and T_All_reduced, respectively. For all of the four collections of statements we obtained very similar results. Figure 17 shows a plot of the sorted values $LD(O)$ (for O being an element of the car data set) as well as the eight cars with smallest values when working with T_All. Looking at the plot it might be reasonable to assume that there are at least four outliers. Indeed, the Formula One car, the fire truck, the motortruck, and the tractor, which appear rather odd in the car data set, are ranked lowest. Also the other cars shown in Figure 17 are quite out of character for the car data set. In the ordinal embedding shown in Figure 18 all these cars are located far outside. These findings support our claim that Algorithm 2 can be used for outlier identification when given only ordinal distance information of the kind ($\star$).

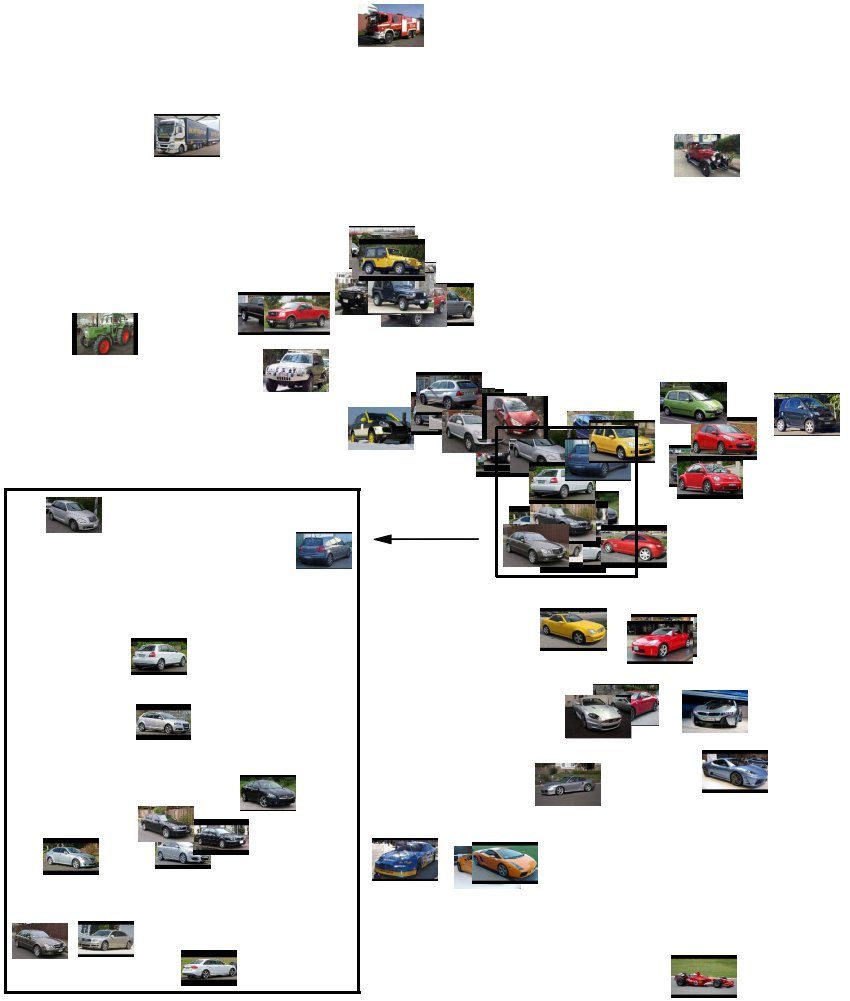


Figure 18: An ordinal embedding of the car data set based on the statements in T_All. A larger version is available on `http://www.tml.cs.uni-tuebingen.de/team/luxburg/code_and_data`.

	All	All_reduced	T_All	T_All_reduced
Number of statements	5624	5121	5349	4886
Number of statements in percent of number of triples $[\binom{56}{3} = 27720]$	20.29	18.47	19.30	17.63

Table 2: Number of statements after removing the fire truck, the motortruck, the tractor, and the antique car from the car data set. Note that All and T_All contain repeatedly present and contradicting statements.

6.2.3 Classification

For setting up a classification problem on the car data set we removed the four outliers (the fire truck, the motortruck, the tractor, and the antique car) and assigned a label to the remaining cars according to which of the three classes of ordinary cars, sports cars, or off-road/sport utility vehicles they belong to. By removing from the collections All, All_reduced, T_All, and T_All_reduced all statements that comprise one or more outliers, we obtained collections of statements of the kind ($\star$) for these 56 labeled cars. A bit sloppy, from now on till the end of Section 6.2, by All, All_reduced, T_All, and T_All_reduced we mean these newly created, reduced collections. Their sizes are given in Table 2.

We randomly selected 16 cars that we used as test points, that is we ignored their labels and predicted them by applying Algorithms 3 and 4 and an embedding approach based on the label information of the remaining 40 labeled cars and the ordinal distance information in All, All_reduced, T_All, or T_All_reduced. In Table 3 we report the average 0-1 loss (see equation (21) for its definition) and its standard deviation, where the average is over hundred random selections of test points, for the considered methods and various classification algorithms on top of Algorithm 3 or the embedding approach. We chose the dimension of the space of the embedding as two. As classifiers on top we used both the k-NN classifier and the SVM algorithm, the latter with the linear as well as with the Gaussian kernel. Since we are dealing with a 3-class classification problem, we combined the SVM algorithm with a one-vs-all strategy. We chose the parameter k for the k-NN classifier and the regularization parameter for the SVM algorithm by means of 10-fold cross-validation from $1, 3, 5, 7, 11, 15$ and $0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100, 500, 1000$, respectively. When using the SVM algorithm with the Gaussian kernel, we chose the kernel bandwidth σ by means of 10-fold cross-validation from $0.01, 0.05, 0.1, 0.5, 1, 5$. The parameter k for Algorithm 4 was chosen from $1, 2, 3, 5, 7, 10, 15$ by means of a non-exhaustive variant of leave-one-out cross-validation as explained in Section 6.1.3. Clearly, the ordinal embedding approach outperforms Algorithms 3 and 4. However, one should judge the performance of our algorithms with regards to their great simplicity compared to the embedding approach. In doing so, we consider the 0-1 loss incurred by Algorithms 3 or 4 to be acceptable. As one might expect, working with T_All or T_All_reduced leads to a slightly lower misclassification rate than working with All or All_reduced.

	All	All_reduced	T_All	T_All_reduced
Alg. 3 with k-NN	0.19 (± 0.11)	0.23 (± 0.12)	0.16 (± 0.10)	0.17 (±0.10)
Alg. 3 with SVM linear	0.17 (± 0.09)	0.16 (± 0.09)	0.13 (± 0.07)	0.16 (± 0.08)
Alg. 3 with SVM Gauss	0.17 (± 0.10)	0.18 (± 0.10)	0.14 (± 0.10)	0.16 (± 0.09)
Algorithm 4	0.15(± 0.09)	0.18 (± 0.10)	0.13 (± 0.09)	0.13 (±0.09)
GNMDS with k-NN	0.05 (± 0.05)	0.05 (± 0.05)	0.04 (± 0.04)	0.04 (±0.04)
GNMDS with SVM linear	0.07 (± 0.07)	0.06 (± 0.06)	0.07 (± 0.07)	0.04 (± 0.05)
GNMDS with SVM Gauss	0.05 (± 0.06)	0.04 (± 0.05)	0.04 (± 0.05)	0.04 (± 0.05)
SOE with k-NN	0.06 (± 0.05)	0.07 (± 0.05)	0.06 (± 0.06)	0.07 (±0.06)
SOE with SVM linear	0.10 (± 0.08)	0.12 (± 0.09)	0.11 (± 0.08)	0.10 (± 0.09)
SOE with SVM Gauss	0.05 (± 0.07)	0.07 (± 0.08)	0.05 (± 0.05)	0.07 (± 0.06)
STE with k-NN	0.05 (± 0.04)	0.03 (± 0.04)	0.05 (± 0.04)	0.04 (± 0.04)
STE with SVM linear	0.07 (± 0.08)	0.06 (± 0.06)	0.08 (± 0.07)	0.06 (± 0.06)
STE with SVM Gauss	0.05 (± 0.05)	0.03 (± 0.05)	0.04 (± 0.05)	0.04 (± 0.05)
t-STE with k-NN	0.09 (± 0.07)	0.10 (± 0.07)	0.06 (± 0.06)	0.08 (± 0.07)
t-STE with SVM linear	0.12 (± 0.09)	0.15 (± 0.11)	0.11 (± 0.09)	0.13 (± 0.09)
t-STE with SVM Gauss	0.09 (± 0.08)	0.08 (± 0.08)	0.07 (± 0.06)	0.08 (± 0.08)

Table 3: Average 0-1 loss (± standard deviation) when predicting labels for a randomly chosen subset of 16 cars (average over 100 choices).

6.2.4 Clustering

Like in the previous Section 6.2.3 we removed the four outliers from the car data set. We then used Algorithm 5 and an ordinal embedding approach for clustering the remaining 56 cars into three clusters, aiming to recover the cars' grouping into classes of ordinary cars, sports cars, and off-road/sport utility vehicles. In the embedding approach we applied spectral clustering to a symmetric k-NN graph with Gaussian edge weights on an ordinal embedding of the data set as we did in Section 6.1.4. Table 4 shows the average purity (see equation (22) for its definition) of the clusterings produced by the considered methods with respect to our assumed ground truth partitioning. The average is over 100 runs of the experiment. Note that clusterings produced by Algorithm 5 and obtained in different runs only differ due to random effects in the K-means step of spectral clustering, while the clusterings produced by the ordinal embedding approach also differ because of the random initialization in the embedding methods. For this reason, standard deviations of the purity values achieved by the embedding approach are much larger than those of the purity values achieved by Algorithm 5 (which are on the order machine epsilon) and are shown in Table 4 too. All methods perform nearly equally well, with the unweighted version of Algorithm 5 slightly inferior compared to the other methods when their parameters are chosen optimally. At least for Algorithm 5 working with the statements in T_All or T_All_reduced yields

	All	All_reduced	T_All	T_All_reduced
Alg. 5 w., $k = 5$, $\sigma = 0.5$	0.82	0.82	0.86	0.88
Alg. 5 w., $k = 5$, $\sigma = 3$	0.84	0.82	0.86	0.86
Alg. 5 w., $k = 10$, $\sigma = 0.5$	0.91	0.84	0.95	0.93
Alg. 5 w., $k = 10$, $\sigma = 3$	0.84	0.91	0.86	0.89
Alg. 5 unw., $k = 5$	0.84	0.82	0.84	0.88
Alg. 5 unw., $k = 10$	0.84	0.84	0.86	0.89
GNMDS, $k = 5$, $\sigma = 0.5$	0.79 ($\pm$ 0.12)	0.83 ($\pm$ 0.12)	0.78 ($\pm$ 0.15)	0.80 ($\pm$ 0.15)
GNMDS, $k = 5$, $\sigma = 3$	0.78 ($\pm$ 0.12)	0.84 ($\pm$ 0.11)	0.76 ($\pm$ 0.14)	0.83 ($\pm$ 0.15)
GNMDS, $k = 10$, $\sigma = 0.5$	0.83 ($\pm$ 0.11)	0.92 ($\pm$ 0.03)	0.93 ($\pm$ 0.04)	0.89 ($\pm$ 0.13)
GNMDS, $k = 10$, $\sigma = 3$	0.78 ($\pm$ 0.12)	0.88 ($\pm$ 0.09)	0.92 ($\pm$ 0.06)	0.95 ($\pm$ 0.03)
SOE, $k = 5$, $\sigma = 0.5$	0.87 ($\pm$ 0.01)	0.87 ($\pm$ 0.04)	0.82 ($\pm$ 0.08)	0.79 ($\pm$ 0.10)
SOE, $k = 5$, $\sigma = 3$	0.82 ($\pm$ 0.09)	0.83 ($\pm$ 0.11)	0.75 ($\pm$ 0.10)	0.73 ($\pm$ 0.10)
SOE, $k = 10$, $\sigma = 0.5$	0.90 ($\pm$ 0.04)	0.90 ($\pm$ 0.03)	0.91 ($\pm$ 0.04)	0.90 ($\pm$ 0.03)
SOE, $k = 10$, $\sigma = 3$	0.93 ($\pm$ 0.03)	0.93 ($\pm$ 0.03)	0.91 ($\pm$ 0.05)	0.89 ($\pm$ 0.08)
STE, $k = 5$, $\sigma = 0.5$	0.75 ($\pm$ 0.11)	0.73 ($\pm$ 0.11)	0.74 ($\pm$ 0.10)	0.73 ($\pm$ 0.11)
STE, $k = 5$, $\sigma = 3$	0.75 ($\pm$ 0.12)	0.76 ($\pm$ 0.11)	0.74 ($\pm$ 0.10)	0.76 ($\pm$ 0.11)
STE, $k = 10$, $\sigma = 0.5$	0.90 ($\pm$ 0.04)	0.87 ($\pm$ 0.01)	0.90 ($\pm$ 0.03)	0.88 ($\pm$ 0.01)
STE, $k = 10$, $\sigma = 3$	0.90 ($\pm$ 0.04)	0.87 ($\pm$ 0.01)	0.77 ($\pm$ 0.14)	0.88 ($\pm$ 0.01)
t-STE, $k = 5$, $\sigma = 0.5$	0.87 ($\pm$ 0.02)	0.87 ($\pm$ 0.02)	0.86 ($\pm$ 0.04)	0.88 ($\pm$ 0.05)
t-STE, $k = 5$, $\sigma = 3$	0.85 ($\pm$ 0.08)	0.89 ($\pm$ 0.03)	0.76 ($\pm$ 0.12)	0.79 ($\pm$ 0.12)
t-STE, $k = 10$, $\sigma = 0.5$	0.92 ($\pm$ 0.03)	0.92 ($\pm$ 0.03)	0.92 ($\pm$ 0.04)	0.91 ($\pm$ 0.04)
t-STE, $k = 10$, $\sigma = 3$	0.94 ($\pm$ 0.03)	0.94 ($\pm$ 0.02)	0.92 ($\pm$ 0.04)	0.91 ($\pm$ 0.06)

Table 4: Average purity ($\pm$ standard deviation) of clusterings produced by the various methods when clustering the 56 cars from the classes of ordinary cars, sports cars, and off-road/sport utility vehicles into three clusters (average over 100 runs).

better results than working with the statements in All or All_reduced, but this does not seem to be the case for the ordinal embedding approach.

7. Discussion, Future Work, and Open Problems

In this paper, we have proposed algorithms for the problems of medoid estimation, outlier identification, classification, and clustering when given only ordinal distance information. We argue that information of the form ($\star$) is particularly useful as it can be related to the lens depth function, which is an instance of a statistical depth function, and k-relative neighborhood graphs. Our algorithms solve the problems by direct approaches instead of constructing an ordinal embedding of a data set as an intermediate step. Thus they avoid

some of the problems inherent in such an embedding approach (discussed in Section 5.1.2). In particular, the running time of our algorithms is lower by several orders of magnitude. In a number of experiments on small data sets we have demonstrated that our algorithms are competitive with or at least not much worse than an embedding approach in terms of the quality of the produced solution. We also performed some experiments on medium-sized data sets, for which there was already no hope to compute an ordinal embedding in somewhat reasonable time, but still our algorithms yielded useful results. Our algorithms are appealingly simple and can easily and highly efficiently be parallelized. Hence, we believe that they are a useful alternative to the generic ordinal embedding approach and applicable in situations in which embedding algorithms are not.

Our work inspires several follow-up questions, we focus on two of them:

- **A more local point of view:** The problems studied in this paper are global problems in the sense that they look at a data set as a whole. In contrast, local problems like density estimation or nearest neighbor search look at single data points and their neighborhoods with respect to the dissimilarity function d, thus spotting only fragments of the data set. The tools used in this paper, the lens depth function and the k-RNG, are global in their nature too. Indeed, as we have seen in Section 6.1.2, the lens depth function cannot detect outliers sitting in-between several modes of a data set since such outliers are globally seen at the heart of the data.

  It is interesting to consider local problems in a setting of ordinal distance information. A concept that becomes attractive then is that of local depth functions: Agostinelli and Romanazzi (2008, 2011) introduced a notion of localized simplicial depth, which can easily be transferred to the lens depth function and is then given by

  $$LD_{\text{local}}(x; \tau, P) = Probability(x \in Lens(X, Y) \wedge d(X, Y) \leq \tau), \tag{23}$$

  where X and Y are independent random variables distributed according to a probability distribution P and $\tau > 0$ is a parameter. Agostinelli and Romanazzi have shown (theoretically for one-dimensional and empirically for multidimensional Euclidean data) that for τ tending to zero their local version of simplicial depth is closely related to the density function of the underlying distribution and that maximizing the local simplicial depth function provides reasonable estimates of the distribution's modes. We believe that such a connection also holds for the local lens depth function (23)—note that in one dimension the lens depth function coincides with the simplicial depth function. Unfortunately, unlike for the ordinary lens depth function, the local lens depth function cannot be evaluated with respect to an empirical distribution of a data set $\mathcal{D}$ given only ordinal distance information of the kind ($\star$) about $\mathcal{D}$. Even if we replace the event "$d(X, Y) \leq \tau$" by the event "$d(X, Y)$ is among the smallest τ distances between data points in $\mathcal{D}$", it is not clear at all how to evaluate or estimate (23). One solution would be to allow for additional ordinal distance information of the general kind (17), that is answers to comparisons $d(X, Y) < d(\widetilde{X}, \widetilde{Y})$, like Ukkonen et al. (2015) do in their paper, but this seems to be a rather unattractive way out. We have tried several heuristics for approximately evaluating the general comparison (17) given only statements of the kind ($\star$), like

  $$Probability(x \in Lens(X, Y) \mid y \in Lens(X, Y)) \approx f(d(x, y))$$

for a monotonically decreasing function $f : \mathbb{R}_0^+ \to [0,1]$, which would be useful since we can easily estimate the probability on the left side. However, none of them was promising. They all suffer from the same problem, namely that the number of data points in $Lens(X,Y)$ can be small for two completely different reasons: either $d(X,Y)$ is small, or $d(X,Y)$ is large, but $Lens(X,Y)$ is located in an area of low probability. Unfortunately, there is no obvious way for distinguishing between these two reasons.

This raises the question whether density estimation or solving any other local problem is possible at all given only ordinal distance information of the kind $(\star)$, and indeed the answer is negative for intrinsically one-dimensional data sets: Consider data points $x_1, \ldots, x_n$ on the real line and assume d to be the Euclidean metric. Then the ordinal distance information given by all statements of the kind $(\star)$ only depends on the order of the data points: given any three data points, the center is always given by the data point sitting in the middle, and any order-preserving transformation of the data points will give rise to exactly the same ordinal distance information. For this reason it is impossible to estimate any local property of an underlying distribution, and ordinal distance information of the kind $(\star)$ comes along with a substantial loss in effective information compared to similarity triplets, that is answers to (18): while, under some assumptions on the data points, all similarity triplets asymptotically uniquely determine the actual positions of the points on the real line up to a similarity transformation (Proposition 10 in Kleindessner and von Luxburg, 2014), all statements of the kind $(\star)$ only determine the ranking of the data points up to inversion. However, such a loss in information does not seem to occur when dealing with data sets of higher intrinsic dimensionality: If we start with data points from $\mathbb{R}^m$ for $m \geq 2$ (again equipped with the Euclidean metric) that are reasonably scattered, collect all statements of the kind $(\star)$, and provide them as input to an ordinal embedding algorithm, then the algorithm is able to almost perfectly recover the configuration of the data points up to a similarity transformation. An example of this happening can be seen in Figure 19 for a set of 100 points in $\mathbb{R}^2$: the blue points are the data points that we start with and the red ones are the recovered data points after a Procrustes analysis, that is aligning the points of the ordinal embedding with the original ones via a similarity transformation. We have observed this phenomenon for a broad variety of point configurations in Euclidean spaces $\mathbb{R}^m$ of arbitrary dimensions $m \geq 2$ and want to state it as a conjecture. We formulate the conjecture similarly to the theorems in Kleindessner and von Luxburg (2014) and Arias-Castro (2015), which state the asymptotic uniqueness property for ordinal data consisting of answers to general dissimilarity comparisons (17) and for similarity triplets, that is answers to comparisons (18).

Conjecture: *Let B_1 and B_2 be two closed and bounded balls in $\mathbb{R}^m$, $m \geq 2$, with arbitrary centers and radii. Let $(x_n)_{n\in\mathbb{N}}$ be a sequence of points $x_n \in B_1$ such that $\{x_n : n \in \mathbb{N}\}$ is dense in B_1. For $n \in \mathbb{N}$, let $X_n = \{x_1, \ldots, x_n\}$ and let $Y_n = \{y_1^n, \ldots, y_n^n\} \subseteq B_2$ be an ordinal embedding of X_n that preserves all statements of the kind $(\star)$ (but not necessarily any other ordinal relationships). Then there exists a sequence $(S_n)_{n\in\mathbb{N}}$ of similarity transformations $S_n : \mathbb{R}^m \to \mathbb{R}^m$ such that*

$$\max_{i=1,\ldots,n} \|S_n(y_i^n) - x_i\| \to 0 \quad as \quad n \to 0.$$

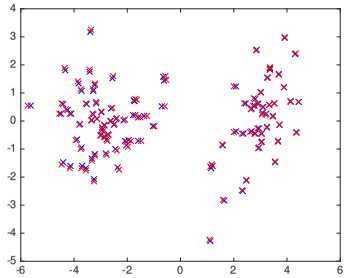


Figure 19: An example illustrating the conjecture of this section. Given all statements of the kind $(\star)$ for the set of blue points, an algorithm for ordinal embedding is able to almost perfectly recover the point configuration up to a similarity transformation (red points—after a Procrustes analysis).

Hence, there is hope: if our conjecture holds, when dealing with a Euclidean data set of known intrinsic dimension, which is greater than one, then all statements of the kind $(\star)$ asymptotically contain all cardinal distance information up to rescaling. At least for such a data set we may hope that, in principle, we are able to solve any local problem that we can solve in a standard machine learning setting of cardinal distance information also in a setting of ordinal distance information of the type $(\star)$. However, it remains an open problem how to solve a local problem in practice except for an embedding approach.

- **Active learning:** Algorithms 1 to 5 can deal with arbitrary collections of statements of the kind $(\star)$ that are gathered *before* the application of the algorithm and are provided as input all at once. However, in many scenarios one might have the chance to actively query statements for intentionally chosen triples of objects. In such a scenario an algorithm for a machine learning task should interact with the process of querying statements and adaptively choose triples of objects for which statements are to be queried in such a way that the task at hand is solved as fast, accurately, cheaply, ... as possible. For the problems of medoid estimation or outlier identification it is easy to adapt Algorithm 1 and Algorithm 2 in order to derive adaptive versions: starting with rough estimates of values $LD(O)$ for every object O in the data set, one could immediately rule out some objects with very small (or high) estimated values and continue improving only estimates of the values of the remaining objects by querying further statements only for them. This strategy has been suggested by Heikinheimo and Ukkonen (2013) for their method for medoid estimation. Studying the questions whether such a strategy comes with any guarantees, whether there might be better alternatives (of course, this depends on what one wants to achieve), or whether similar approaches apply to Algorithms 3 to 5 is left for future work.

Two further follow-up questions are: I. If one is free to choose the particular type of ordinal data that one is working with, for example in a crowdsourcing scenario, then which type is

most appropriate for which problems (in terms that it is both informative for the problem at hand and can easily be provided by the crowd)? II. How can we fix the error in the proof of Theorem 6 in Liu and Modarres (2011) (see Section 5.2)?

Acknowledgments

This work was supported by the Institutional Strategy of the University of Tübingen (Deutsche Forschungsgemeinschaft, ZUK 63).

References

S. Agarwal, J. Wills, L. Cayton, G. Lanckriet, D. Kriegman, and S. Belongie. Generalized non-metric multidimensional scaling. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2007.

C. Agostinelli and M. Romanazzi. Local depth of multivariate data. Technical report, Ca' Foscari University of Venice, 2008.

C. Agostinelli and M. Romanazzi. Local depth. *Journal of Statistical Planning and Inference*, 141(2):817–830, 2011.

E. Amid and A. Ukkonen. Multiview triplet embedding: Learning attributes in multiple maps. In *International Conference on Machine Learning (ICML)*, 2015.

E. Amid, N. Vlassis, and M. Warmuth. t-exponential triplet embedding. arXiv:1611.09957 [cs.AI], 2016.

D. V. Andrade and L. H. de Figueiredo. Good approximations for the relative neighbourhood graph. In *Canadian Conference on Computational Geometry (CCCG)*, 2001.

E. Arias-Castro. Some theory for ordinal embedding. arXiv:1501.02861 [math.ST], 2015.

Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, editors. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. Society for Industrial and Applied Mathematics, 2000.

V. Barnett and T. Lewis. *Outliers in Statistical Data*. Wiley, 1978.

R. Bartoszynski, D. K. Pearl, and J. Lawrence. A multidimensional goodness-of-fit test based on interpoint distances. *Journal of the American Statistical Association*, 92(438): 577–586, 1997.

M. Blum, R. W. Floyd, V. Pratt, R. L. Rivest, and R. E. Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7:448–461, 1973.

I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, 2005.

P. Bose, V. Dujmović, F. Hurtado, J. Iacono, S. Langerman, H. Meijer, V. Sacristán, M. Saumell, and D. R. Wood. Proximity graphs: E, δ, Δ, χ and ω. *International Journal of Computational Geometry and Applications*, 22(5):439–469, 2012.

F. C. Botelho, R. Pagh, and N. Ziviani. Simple and space-efficient minimal perfect hash functions. In *Workshop on Algorithms and Data Structures (WADS)*, 2007.

I. Cascos. Data depth: Multivariate statistics and geometry. In W. S. Kendall and I. Molchanov, editors, *New Perspectives in Stochastic Geometry*. Oxford University Press, 2009.

M. S. Chang, C. Y. Tang, and R. C. T. Lee. Solving the euclidean bottleneck matching problem by k-relative neighborhood graphs. *Algorithmica*, 8(1–6):177–194, 1992.

Y. Chen, X. Dang, H. Peng, and H. L. Bart, Jr. Outlier detection with the kernelized spatial depth function. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(2): 288–305, 2009.

C. D. Correa and P. Lindstrom. Locally-scaled spectral clustering using empty region graphs. In *ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2012.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other Kernel-Based Learning Methods*. Cambridge University Press, 2000.

X. Dang and R. Serfling. Nonparametric depth-based multivariate outlier identifiers, and masking robustness properties. *Journal of Statistical Planning and Inference*, 140(1): 198–213, 2010.

R. T. Elmore, T. P. Hettmansperger, and F. Xuan. Spherical data depth and a multivariate median. In R. Y. Liu, R. Serfling, and D. L. Souvaine, editors, *Data Depth: Robust Multivariate Analysis, Computational Geometry and Applications*. American Mathematical Society, 2006.

L. C. Freeman. Centrality in social networks: Conceptual clarification. *Social Networks*, 1 (3):215–239, 1978.

K. R. Gabriel and R. R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18(3):259–278, 1969.

A. K. Ghosh and P. Chaudhuri. On maximum depth and related classifiers. *Scandinavian Journal of Statistics*, 32(2):327–350, 2005.

T. Hagerup and T. Tholey. Efficient minimal perfect hashing in nearly minimal space. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, 2001.

S. Haghiri, D. Ghoshdastidar, and U. von Luxburg. Comparison based nearest neighbor search. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

T. B. Hashimoto, Y. Sun, and T. S. Jaakkola. Metric recovery from directed unweighted graphs. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

H. Heikinheimo and A. Ukkonen. The crowd-median algorithm. In *Conference on Human Computation and Crowdsourcing (HCOMP)*, 2013.

E. Heim, M. Berger, L. M. Seversky, and M. Hauskrecht. Efficient online relative comparison kernel learning. In *SIAM International Conference on Data Mining (SDM)*, 2015.

L. Jain, K. G. Jamieson, and R. Nowak. Finite sample prediction and recovery bounds for ordinal embedding. In *Neural Information Processing Systems (NIPS)*, 2016.

K. G. Jamieson and R. Nowak. Low-dimensional embedding using adaptively selected ordinal data. In *Conference on Communication, Control, and Computing*, 2011.

J. W. Jaromczyk and G. T. Toussaint. Relative neighborhood graphs and their relatives. *Proceedings of the IEEE*, 80(9):1502–1517, 1992.

M. Kleindessner and U. von Luxburg. Uniqueness of ordinal embedding. In *Conference on Learning Theory (COLT)*, 2014.

M. Kleindessner and U. von Luxburg. Dimensionality estimation without distances. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964a.

J. B. Kruskal. Nonmetric multidimensional scaling: A numerical method. *Psychometrika*, 29(2):115–129, 1964b.

J. Lawrence. *Interpoint Distance Methods for the Analysis of High Dimensional Data*. PhD thesis, The Ohio State University, 1996.

J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1), 2007. Data available on `https://snap.stanford.edu/data/`.

J. Li, J. A. Cuesta-Albertos, and R. Y. Liu. *DD*-classifier: Nonparametric classification procedure based on *DD*-plot. *Journal of the American Statistical Association*, 107(498): 737–753, 2012.

M. Li, X.-C. Lian, J. T.-Y. Kwok, and B.-L. Lu. Time and space efficient spectral clustering via column sampling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

C. Liu, K. Wu, and T. He. Sensor localization with ring overlapping based on comparison of received signal strength indicator. In *IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2004.

R. Y. Liu. On a notion of simplicial depth. *Proceedings of the National Academy of Sciences of the United States of America*, 85(6):1732–1734, 1988.

R. Y. Liu. On a notion of data depth based on random simplices. *The Annals of Statistics*, 18(1):405–414, 1990.

R. Y. Liu. Data depth and multivariate rank tests. In Y. Dodge, editor, *L1-Statistical Analysis and Related Methods*. North Holland, 1992.

R. Y. Liu, J. M. Parelius, and K. Singh. Multivariate analysis by data depth: descriptive statistics, graphics and inference. *The Annals of Statistics*, 27(3):783–840, 1999.

Z. Liu and R. Modarres. Lens data depth and median. *Journal of Nonparametric Statistics*, 23(4):1063–1074, 2011.

C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

K. Mosler. Depth statistics. In C. Becker, R. Fried, and S. Kuhnt, editors, *Robustness and Complex Data Structures: Festschrift in Honour of Ursula Gather*. Springer, 2013.

J. S. Sánchez, F. Pla, and F. J. Ferri. On the use of neighbourhood-based non-parametric classifiers. *Pattern Recognition Letters*, 18(11–13):1179–1186, 1997a.

J. S. Sánchez, F. Pla, and F. J. Ferri. Prototype selection for the nearest neighbour rule through proximity graphs. *Pattern Recognition Letters*, 18(6):507–513, 1997b.

M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *Neural Information Processing Systems (NIPS)*, 2003.

R. Serfling. Depth functions in nonparametric multivariate inference. In R. Y. Liu, R. Serfling, and D. L. Souvaine, editors, *Data Depth: Robust Multivariate Analysis, Computational Geometry and Applications*. American Mathematical Society, 2006.

S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

B. Shaw and T. Jebara. Structure preserving embedding. In *International Conference on Machine Learning (ICML)*, 2009.

R. N. Shepard. The analysis of proximities: Multidimensional scaling with an unknown distance function. I. *Psychometrika*, 27(2):125–140, 1962a.

R. N. Shepard. The analysis of proximities: Multidimensional scaling with an unknown distance function. II. *Psychometrika*, 27(3):219–246, 1962b.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

N. Stewart, G. D. A. Brown, and N. Chater. Absolute identification by relative judgment. *Psychological Review*, 112(4):881–911, 2005.

O. Tamuz, C. Liu, S. Belongie, O. Shamir, and A. T. Kalai. Adaptively learning the crowd kernel. In *International Conference on Machine Learning (ICML)*, 2011.

Y. Terada and U. von Luxburg. Local ordinal embedding. In *International Conference on Machine Learning (ICML)*, 2014. Code available on `https://cran.r-project.org/web/packages/loe`.

G. T. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12(4):261–268, 1980.

G. T. Toussaint. Applications of the relative neighbourhood graph. In *International Conference on Advances in Computing, Communication and Information Technology (CCIT)*, 2014.

G. T. Toussaint and C. Berzan. Proximity-graph instance-based learning, support vector machines, and high dimensionality: An empirical comparison. In *International Conference on Machine Learning and Data Mining (MLDM)*, 2012.

G. T. Toussaint, B. K. Bhattacharya, and R. S. Poulsen. The application of voronoi diagrams to non-parametric decision rules. In *Symposium on the Interface of Computing Science and Statistics*, 1984.

J. W. Tukey. Mathematics and the picturing of data. In *International Congress of Mathematicians (ICM)*, 1974.

A. Ukkonen, B. Derakhshan, and H. Heikinheimo. Crowdsourced nonparametric density estimation using relative distances. In *Conference on Human Computation and Crowdsourcing (HCOMP)*, 2015.

G. Van Bever. *Contributions to Nonparametric and Semiparametric Inference based on Statistical Depth*. PhD thesis, Université libre de Bruxelles, 2013.

L. J. P. van der Maaten and K. Q. Weinberger. Stochastic triplet embedding. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2012. Code available on `http://homepage.tudelft.nl/19j49/ste`.

U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

U. von Luxburg and M. Alamgir. Density estimation from unweighted k-nearest neighbor graphs: a roadmap. In *Neural Information Processing Systems (NIPS)*, 2013.

M. J. Wilber, I. S. Kwak, and S. J. Belongie. Cost-effective hits for relative similarity comparisons. In *Conference on Human Computation and Crowdsourcing (HCOMP)*, 2014.

L. Xiao, R. Li, and J. Luo. Sensor localization based on nonmetric multidimensional scaling. In *International Conference on Sensing, Computing and Automation (ICSCA)*, 2006.

D. Yan, L. Huang, and M. I. Jordan. Fast approximate spectral clustering. In *ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2009.

M. Yang. *Depth Functions, Multidimensional Medians and Tests of Uniformity on Proximity Graphs*. PhD thesis, The George Washington University, 2014.

Y. Zuo and R. Serfling. General notions of statistical depth function. *The Annals of Statistics*, 28(2):461–482, 2000.

Journal of Machine Learning Research 18 (2017) 1-39 Submitted 4/16; Revised 3/17; Published 7/17

Joint Label Inference in Networks

Deepayan Chakrabarti deepay@utexas.edu
*IROM, McCombs School of Business, University of Texas, Austin**

Stanislav Funiak stano@cerebras.net
*Cerebras Systems**

Jonathan Chang slycoder@gmail.com
*Health Coda**

Sofus A. Macskassy sofmac@branch.io
*Branch Metrics**

Editor: Edo Airoldi

Abstract

We consider the problem of inferring node labels in a partially labeled graph where each node in the graph has multiple label *types* and each label type has a large number of possible labels. Our primary example, and the focus of this paper, is the joint inference of label types such as hometown, current city, and employers for people connected by a social network; by predicting these user profile fields, the network can provide a better experience to its users. Existing approaches such as Label Propagation (Zhu et al., 2003) fail to consider interactions between the label types. Our proposed method, called EdgeExplain, explicitly models these interactions, while still allowing scalable inference under a distributed message-passing architecture. On a large subset of the Facebook social network, collected in a previous study (Chakrabarti et al., 2014), EdgeExplain outperforms label propagation for several label types, with lifts of up to 120% for recall@1 and 60% for recall@3.

Keywords: label inference, graphs, social networks, variational methods, label propagation

1. Introduction

The most common classification setting assumes the presence of independent and identically distributed training examples, and attempts prediction on independent test instances drawn from the same distribution. However, the case of dependent examples has drawn much interest as well. This is particularly the case for networks where the nodes comprise the training/testing instances, and the edges between them imply dependence.

Consider the problem of inferring the labels of nodes in a network. For example, given a network of sports-related blogs connected by hyperlinks, we may want to know which blogs discuss baseball, or football, or hockey. The training data consists of known labels for a subset of blogs. The goal is to infer the labels of the remaining blogs. While the content of the blogs can be useful, the hyperlinks between blogs are clearly informative: we expect blogs to link primarily to other blogs discussing the same sport. This is a network label inference problem with only one label type (sport) and only a few labels (baseball, football, and hockey). The question we ask is: *how can we infer missing labels of multiple types, where each label type can be high-dimensional and the different types can be correlated?*

∗. This research was done while the authors were employees at Facebook, Inc.

©2017 Deepayan Chakrabarti, and Stanislav Funiak, and Jonathan Chang, and Sofus A. Macskassy.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-214.html.

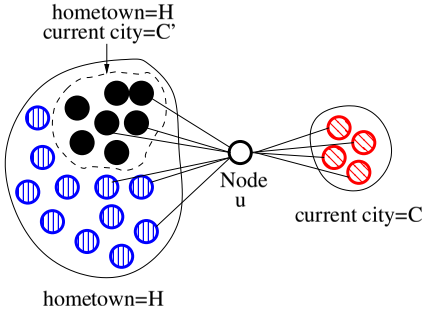


Figure 1: *An example graph of u and her friends:* The hometown friends of u coincidentally contain a subset with current city C'. This swamps the group from u's actual current city C, causing label propagation to infer C' for u. However, our proposed model (called EdgeExplain) correctly explains all friendships by setting the hometown to be H and current city to be C.

The concrete problem considered in this paper is that of inferring multiple fields such as the hometowns, current cities, and employers of users of a social network, where users often only partially fill in their profile, if at all. Here, each user is associated with one label of each *type* (such as hometown, employer, etc.), and the set of possible labels for each type is very high-dimensional. Inference of such label types is important for many ranking and relevance applications. By predicting the profile fields, the social network can make better friend recommendations or show more relevant content. Consequently, accurate predictions can greatly improve the user experience. We show that this inference problem cannot be split up into separate problems, one for each label type, without loss of accuracy. Knowledge of one label type influences inferences regarding the other types. Thus, this *joint inference* problem presents interesting opportunities for modeling and optimization.

A standard method of label inference is label propagation (Zhu and Ghahramani, 2002; Zhu et al., 2003), which tries to set the label probabilities of nodes so that friends have similar probabilities. This is based on the idea of homophily, that is, the more two nodes have in common, the more likely they are to connect (McPherson et al., 2001). However, label propagation assumes only a single category of relationships. It therefore fails to address the complexity of edge formation in networks, where nodes have different reasons to link to each other. As an example, consider the snapshot of a social network in Figure 1, where we want to predict the hometown and current city of node u, given what we know about u and u's neighbors. Here, the labels of node u are completely unknown, but her friends' labels are completely known. Label propagation would treat each label independently and infer the hometown of u to be the most common hometown among her friends, the current city to be the most common current city among friends, and so on. Hence, if the bulk of friends

of u are from her hometown H, then inferences for current city will be dominated by the most common current city among her hometown friends (say, C') and not friends from her actual current city C; indeed, the same will happen for all other label types as well.

Our proposed method, named EdgeExplain, approaches the problem from a different viewpoint, using the following intuition: Two people form an edge in a social network because they share the same label for one or more label types (e.g., both went to the same college). Using this intuition, we can go beyond standard label propagation in the following way: instead of taking the graph as given, and modeling labels as items that propagate over this graph, we consider the labels as factors that can *explain* the observed graph structure. For example, the inferences for u made by label propagation leave u's edges from C completely unexplained. Our proposed method rectifies this, by trying to infer node labels such that for each edge $u \sim v$, we can explain the existence of the edge in terms of a shared label — u and v are friends from the same hometown, or college, or the like. While we are primarily interested in inferring labels, we note that the inferred reason for each edge can be useful by itself. For example, if a new node u joins a network and forms and edge with v, and we can infer that the reason is a shared college, we can recommend other college friends of v as possible new edges for u.

We note that a seemingly simple alternative solution—cluster the graph and then propagate the most common labels within a cluster—is in fact quite problematic. The clustering must also be complex enough to allow many overlapping clusters. This is an active area of research (Xie et al., 2013), but it still entails a significant computational burden. In addition, any clustering based solely on the graph structure ignores labels already available from user profiles, which clearly carry crucial information. A clustering method that tries to use these labels must deal with incomplete and missing labels, significantly increasing its complexity. Instead, our method will be able to operate only on dyads, without having to infer the higher-level clusters themselves. Social network cluster sizes must display wide variation (e.g., the cluster for users living in New York city versus those in small rural areas), and it is unclear if current network community detection algorithms offer good performance across the entire range of community sizes (Leskovec et al., 2010). Hence, we believe that clustering does not readily lend itself to a solution for our problem.

This paper tackles the following five questions.

How can "explaining links" be codified in a model? We propose a probabilistic model, called EdgeExplain, for the social network given the labels of all nodes in the network. The model has two key properties. First, the presence or absence of a link is conditionally independent of all other nodes and edges given the labels of the two endpoints of the link. This enables distributed computation for inference, which is useful for large networks. Second, labels corresponding to all the label types are jointly considered in the model.

How can inference be performed efficiently on large networks? The model likelihood function is non-convex, and only local optima can be found. However, we present several approaches for this problem. The first uses alternating maximization on a relaxation of the objective, such that each iteration solves a convex problem. The second is a variational approach that operates directly on the model objective. We show via simulations the presence of two regimes: while the alternating minimization works best when there is significant missing data, the variational approach works better in the presence of copious and correct information, i.e., when the labels of most neighbors of a node are known with relatively high certainty. This leads to our final hybrid method that combines the two approaches.

How do the model parameters affect accuracy vis-a-vis label propagation? We present an analysis of EdgeExplain that specifies the conditions under which label propagation will be particularly inaccurate. The quality of predictions of label propagation for a user u is shown to depend on three factors: (a) the degree to which any one label type is the preferred "reason" for forming friendships as compared to the other label types, (b) the degree to which any one label dominates among all labels of a particular type, and (c) the number of friends of a user u.

How well does it work in practice? Our iterative methods for inferring labels can be easily implemented in large-scale message-passing architectures. We empirically demonstrate their scalability on a large subset of the Facebook social network (Chakrabarti et al., 2014) and a 5 million node sample of the Google+ network (Gong et al., 2012), using publicly available user profiles and friendships. On Facebook, EdgeExplain significantly outperforms label propagation for several label types, with lifts of up to 120% for recall@1 and 60% for recall@3. On the Google+ sample, which exhibits extreme sparsity of labels, all algorithms are roughly equal. We also present empirical results on a movie network, where EdgeExplain outperforms label propagation as well as other competing methods (Zheleva and Getoor, 2009; Yin et al., 2010; Gong et al., 2014). The accuracy and scalability of EdgeExplain clearly demonstrate its usefulness for label inference on large networks.

How do variational methods compare with relaxation labeling? A subtheme of our paper is the comparison, via simulations as well as empirical evaluation on real-world data, of variational and relaxation labeling techniques. When both are designed to use only per-node parameters (which are well-suited for message-passing architectures), we find that relaxation labeling works better for our problem. This is in spite of the fact that our variational updates are exact, and only the standard approximation of the joint distribution with a product of per-node marginals is made.

The paper is organized as follows. We survey related work in Section 2, and then present our proposed EdgeExplain model in Section 3. For parameter inference, we present two methods: a gradient ascent approach on a relaxed version of the problem (Section 4) and a variational approach for optimizing the original problem objective (Section 5). We then present a detailed analysis of our model in Section 6, focusing on the conditions under which label propagation would fail. Simulations on synthetic data sets (Section 7) confirm our analysis, and also help us identify the conditions under which the two inference mechanisms work best. Empirical evidence demonstrating the effectiveness of EdgeExplain is presented in Section 8. Generalizations of the model are discussed in Section 9. We conclude in Section 10.

2. Related Work

The problem of inferring labels of nodes in a network has been studied extensively in the literature on graph-based semi-supervised learning, statistical relational learning, and latent models for networks. In this section, we highlight the prior work in these fields, and then review recent works that specifically focus on the problem of attribute inference in social networks.

Semi-supervised learning. Our work falls under the paradigm of graph-based semi-supervised learning (Zhu, 2008), where the goal is to estimate a function over the nodes of the graph that satisfies two properties: (1) the function is close (or equal) to the observed labels, and (2) the function is smooth (similar) at adjacent nodes. In their seminal paper, Zhu and Ghahramani (2002) proposed an algorithm called Label Propagation, where the values at the nodes are relaxed to be real-valued, and inference is performed by averaging the values at neighboring nodes. Their follow-up paper (Zhu et al., 2003) formulated Label Propagation as inference in Gaussian random fields, stating additional interpretations as random walks on the graph and electric networks. The Gaussian random field formulation can be viewed as a special case of the regularization framework proposed by Zhou et al. (2003) that allows a tradeoff between edge smoothness and label fitting. This framework was analyzed by Belkin et al. (2004), proving generalization bounds and stability properties. Belkin et al. (2006) placed the problem in the context of reproducing kernel Hilbert spaces to provide further theoretical basis for the algorithms and obtain an out-of-sample extension. Consequently, they can handle both the inductive and transductive settings. Finally, Baluja et al. (2008) and Talukdar and Crammer (2009) introduced abandonment probability into the random walk, which can be viewed as having a special "dummy" label. This formulation addresses a challenge present in some graphs, where the random walk becomes uninformative once it visits a high-degree node. None of these approaches consider interactions between multiple label types, and hence fail to capture the edge formation process in graphs considered here. Although we do assume that observed labels are fixed, our approach could be generalized to handle soft constraints on node labels. Finally, while we do not address the inductive setting, the graphs we consider tend to change slowly, hence we can leverage the standard approach of initializing the iteration with the results of the previous run.

Often, the algorithms for graph-based semi-supervised learning can be expressed as an update rule that scales linearly with the number of nodes and edges of the graph. However, the update and space complexity are also linear in the number of distinct label values; this number can be very large in some applications (e.g., Deng et al., 2009; Shi et al., 2009). To address this challenge, Talukdar and Cohen (2014) use count-min sketch, a randomized data structure that allows them to reduce the time and space complexity to be logarithmic in the number of distinct labels. We face a similar challenge in our work, because the number of distinct values in each field is very large. However, instead of using count-min sketches, we have found it effective to compute a sparse projection of the distribution onto the probability simplex in each update (Kyrillidis et al., 2013).

Statistical relational learning. Our work is also related to the field of statistical relational learning. Here, the goal is to classify an entity such as a web page or a patent into a fixed number of labels (classes). Each entity is typically associated with attributes such as text, and we observe the links between entities. Chakrabarti et al. (1998) noted that it is ineffective to build a classifier based on the attributes of the node and its neighbors directly. Instead, they proposed a naive Bayes classifier that considers only the neighbors' labels (in addition to the local attributes). In order to propagate information through the network, they use an approximation of belief propagation. Lu and Getoor (2003) extended this work with a structured logistic regression model built separately on the local content and the links, using maximum a posteriori probability (MAP) for inference. Macskassy and Provost (2007) show that these approaches can be placed in a single framework, with

different choices of (a) a local classifier that uses a node's attributes alone, (b) a relational classifier that uses the labels at adjacent nodes, and (c) a collective inference procedure that propagates the information through the network. They provide an extensive empirical evaluation of the various combinations of these choices and observe that the best combination, weighted-vote relational neighbor classifier (Macskassy and Provost, 2003) with relaxation labeling (Rosenfeld et al., 1976; Hummel and Zucker, 1983), tends to perform as well as label propagation, which we outperform. We note that these algorithms typically focus on a single label type, whereas we explicitly model the interactions among multiple types. Although there has been some work on understanding how to combine and weigh different edge types for best prediction performance (Macskassy, 2007), the edge types (analogous to our reason for an edge) were given up front; we recover them automatically.

There is also extensive work on probabilistic relational models. Relational Bayesian Networks (RBNs; Koller and Pfeffer, 1998; Friedman et al., 1999) provide representation of organizational structure while maintaining a coherent probabilistic representation of uncertainty. Unfortunately, due to their directed nature, RBNs cannot efficiently represent homophily. Relational Markov Networks (Taskar et al., 2002) and Relational Dependency Networks (Neville and Jensen, 2007) do not impose acyclicity, and could be conceptually applied to the problem considered in this paper. Yet, we cannot use these general formalisms directly, because learning the models with high-cardinality labels would require large amounts of labeled data. Instead, it is our explicit modeling assumptions regarding multiple label types that yield gains in accuracy.

Latent models. Graph structure has been modeled using latent classes (Nowicki and Snijders, 2001; Kemp et al., 2006; Xu et al., 2006; Airoldi et al., 2008) and latent variables (Hoff et al., 2002; Miller et al., 2009; Palla et al., 2012), with an emphasis on link prediction. Nowicki and Snijders (2001) describe a stochastic blockmodel (SBM), where each node belongs to one of a fixed number of clusters, and edge generation is governed entirely by the clusters of the adjacent nodes. Kemp et al. (2006) and Xu et al. (2006) extend this work to an unknown number of clusters using the Chinese restaurant process (Pitman, 2006), while Airoldi et al. (2008) propose a mixed membership stochastic blockmodel (MMSB), where the node class membership varies from one dyad to another. The use of latent variables was initially explored in (Hoff et al., 2002), who consider generative process of edges based on the embedding of the adjacent nodes. The Latent Feature Infinite Relational Model (Miller et al., 2009) uses an infinite number of binary features controlled by an Infinite buffet process (Griffiths and Ghahramani, 2005); the Infinite Latent Attribute Model (Palla et al., 2012) then allows each feature to be partitioned into disjoint subgroups. Similarly to these approaches, in our work, each node is associated with (a finite number of) attributes. However, unlike the literature on latent models, where the latent features can be arbitrary combinations of user attributes and are unlikely to represent the concrete label types we wish to predict, our goal is to make predictions about the label types directly, using the observations made at a subset of the nodes. Several of these models may also not easily scale to large networks.

There is also a growing body of work on simultaneously explaining the connections between documents as well as their word distributions (Nallapati et al., 2008; Chang and Blei, 2010; Ho et al., 2012). Nallapati et al. (2008) propose two models that combine Latent Dirichlet allocation (LDA; Blei et al., 2003) and Mixed membership stochastic blockmodel

(MMSB; Airoldi et al., 2008), where topic proportions generate both word topics and the cluster assignments for linking. Chang and Blei (2010) also combine LDA and linking, but use the entire vector of word topic assignments for linking, thus allowing their model to better infer words from links and vice versa. Finally, Ho et al. (2012) use a hierarchy of topics specified as a nested Chinese restaurant process (Blei et al., 2010), where the link between two nodes is given by the deepest level they share. Although we do not consider the problem of modeling text data, our model permits us to incorporate node attributes, and we show empirical results for an instance where group memberships are used as additional node features. The number of distinct label values in our application is very large (on the order of millions), and we suspect that the latent variables would have to have a large dimension to explain the edges in our graph well.

Attribute inference in social networks. Several of the aforementioned techniques have recently been applied specifically to the problem of inferring node attributes in social networks. Zheleva and Getoor (2009) study the privacy implications of methods that can predict a user's private profile from the public profiles of others in the social network. They compare several algorithms for profile inference, and find that two methods work well when no extra side-information is available. One is collective classification based on the labels of neighboring nodes, where their specific implementation is essentially identical to Label Propagation. The second method (called LINK) predicts a node's labels based on a feature vector consisting of the node-IDs of its neighboring nodes. Thus, a graph with N nodes has N features, which is clearly difficult to scale.

Dong et al. (2014) consider inference of gender (2 categories) and age (4 categories) using a social network. They consider three sets of factors: (a) connections between the attribute values of a node and the node's network characteristics (such as its degree), (b) connections between attributes of node pairs connected by a link, and (c) factors for attributes of social triads. This is a comprehensive model, and is one of the few that considers triadic factors. However, the model requires several parameters for each possible value of the (gender, age-category) attribute vector. This is feasible only when the set of attribute values is extremely restricted (only 8 possibilities in their setting). Our primary problem setting has five attributes, with millions of possible values for each attribute; scaling the method to this setting appears to be extremely difficult.

Yin et al. (2010) propose attribute inference on a combined "Social Attribute Network (SAN)," where both people and attribute values are represented by nodes, and links connect people to their friends as well as to their attribute values (whenever these are known). For nodes with no attribute links, they predict the most likely attributes using a random walk on the SAN. Gong et al. (2014) extend this idea by adapting a variety of traditional link-prediction algorithms to the SAN. They find that two methods work well for attribute prediction on the Google+ SAN. The first performs random walks with restarts (RWR-SAN). The second method uses the number of "common neighbors" (CN-SAN) as a predictor. Specifically, the affinity of user u towards attribute a is measured by the number of neighbors of u who are connected to a in the SAN, and each unlabeled user picks the attribute for which she has the highest affinity. This process is iterated until convergence. Note that this method can be interpreted as a variant of Label Propagation where each unlabeled node is assigned the most common attributes among her friends (in contrast to vanilla Label Propagation where nodes have distributions over attributes).

Thus, recent works confirm the findings of earlier comparative studies (Macskassy and Provost, 2007) that Label Propagation (LP) achieves the best prediction accuracy. We shall show that our method (EdgeExplain) is as accurate as LP on the Google+ data set; LP is equivalent to CN-SAN, which was one of the two best methods for this data set (Gong et al., 2014). However, the Google+ data set suffers from extreme sparsity. On the Facebook data set, EdgeExplain achieves significantly higher accuracy as compared to LP. We also compare EdgeExplain against LP and all the feasible methods mentioned above (LINK, CN-SAN, and RWR-SAN) on a SAN derived from IMDB movie data, and show that EdgeExplain outperforms all of them.

3. Proposed Model

In this section we first build intuition about our model using a running example. Suppose we want to infer the *labels* (e.g., "Palo Alto High School" and "Stanford University") corresponding to several *label types* (e.g., high school and college) for a large collection of users. The available data consist of labels publicly declared by some users, and the (public) social network among users, as defined by their friendship network. While the desired set of label types may depend on the application, here we focus on five label types: hometown, high school, college, current city, and employer.

Our solution exploits three properties of these label types:

(P1) They represent the primary situations where two people can meet and become friends, for example, because they went to the same high school or college.

(P2) These situations are (mostly) mutually exclusive. While there may be occasional friendships sharing, say, hometown and high-school, we make the simplifying assumption that most edges can be explained by only one label type.

(P3) Sharing the same label is a *necessary* but not *sufficient* condition. For example, "We are friends from Chicago" typically implies that the indicated individuals were, at some point in time, co-located in a small area within Chicago (say, lived in the same building, met in the same cafe), but hardly implies that two randomly chosen individuals from Chicago are likely to be friends.

(P1) is a direct result of our application; our desired label types were targeted at friendship formation. Combined with **(P2)**, our five label types can be considered a set of mutually exclusive and exhaustive "reasons" for friendship; while this is not strictly true for high school and hometown, empirical evidence suggests that it is a good approximation (shown later in Section 8) and we defer a discussion on this point to Section 9. However, as **(P3)** shows, we cannot simply cast the labels as features whose mere presence or absence significantly affects the probability of friendship; instead, a more careful analysis is needed.

Formally, we are given a graph, $\mathcal{G} = (V, E)$ and a set of label types $\mathcal{T} = \{t_1, \ldots, t_{|\mathcal{T}|}\}$. For each label type t, let $L(t)$ denote the (high-dimensional) set of labels for that label type. Each node in the graph is associated with binary variables $S_{ut\ell}$, where $S_{ut\ell} = 1$ if node $u \in V$ has label ℓ for label type t. Let $\boldsymbol{S}_V$ and $\boldsymbol{S}_H$ represent the sets of visible and hidden variables, respectively. We want to infer the correct values of $\boldsymbol{S}_H$, given $\boldsymbol{S}_V$ and $\mathcal{G}$.

A popular method for label inference is *label propagation* (Zhu and Ghahramani, 2002; Zhu et al., 2003). For a single label type, this approach represents the labeling by a set of indicator variables $S_{u\ell}$, where $S_{u\ell} = 1$ if node u is labeled as ℓ and 0 otherwise. Zhu et al. (2003) relax the labeling to real-valued variables $f_{u\ell}$ over all nodes u and labels ℓ that are clamped to one (or zero) for nodes known to possess that label (or not). They then define a quadratic energy function that assigns lower energy states to configurations where f at adjacent nodes are similar:

$$E(\boldsymbol{f}) = \frac{1}{2}\sum_{u\sim v} w_{uv} \sum_{\ell} (f_{u\ell} - f_{v\ell})^2. \tag{1}$$

Here, $u \sim v$ means that u and v are linked by an edge, and w_{uv} is a non-negative weight on the edge $u \sim v$. The minimum of Eq. 1 is found by solving the fixed point equations

$$f_{u\ell} = \frac{1}{d_u}\sum_{u\sim v} w_{uv} f_{v\ell},$$

where $d_u = \sum_{u\sim v} w_{uv}$. This procedure encourages $f_{u\ell}$ of nodes connected to clamped nodes to be close to the clamped value and propagates the labels outwards to the rest of the graph. Multiple label types can be handled similarly by minimizing Eq. 1 independently for each type.

While this formulation makes full use of **(P1)** and has the advantage of simplicity, it completely ignores **(P2)**. Intuitively, label propagation assumes that friends tend to be similar in *all* respects (i.e., all label types), whereas what **(P2)** suggests is that each friendship tends to have a single *reason*: an edge $u \sim v$ exists because u and v share the same high school *or* college *or* current city, etc. This highly non-linear function is not easily expressed as a quadratic or similar variant.

Instead, we propose a different probabilistic model, which we call EdgeExplain. As described above, let $\mathcal{G}$ denote the graph, and $\boldsymbol{S}_V$ and $\boldsymbol{S}_H$ represent the sets of visible and hidden variables respectively; the variable $S_{ut\ell}$ is known (visible) if user u has publicly declared the label ℓ for type t, and unknown (hidden) otherwise. We define EdgeExplain as follows:

$$P(\boldsymbol{S}_V, \boldsymbol{S}_H \mid \mathcal{G}) = \frac{1}{Z}\prod_{u\sim v} \underset{t\in\mathcal{T}}{\text{softmax}}(r(u,v,t)) \tag{2}$$

$$r(u,v,t) = \sum_{\ell\in L(t)} S_{ut\ell} S_{vt\ell} \tag{3}$$

$$\underset{t\in\mathcal{T}}{\text{softmax}}(r(u,v,t)) = \sigma\left(\alpha \sum_{t\in\mathcal{T}} r(u,v,t) + c\right), \tag{4}$$

where Z is a normalization constant. Here, $r(u,v,t)$ indicates whether a shared label type t is the *reason* underlying the edge $u \sim v$ (Eq. 3). The $\text{softmax}(r_1, \ldots, r_{|\mathcal{T}|})$ function should have three properties: (a) it should be monotonically non-decreasing in each argument, (b) it should achieve a value close to its maximum as long as any one of its parameters is "high", and also (c) it should be differentiable, for ease of analysis. In Eq. 4, we use the sigmoid function to implement this: $\sigma(x) = 1/(1+e^{-x})$. This monotonically increases from

0 to 1, and achieves values greater than $1-\epsilon$ once x is greater than an ϵ-dependent threshold. In addition, the sigmoid enables fine control of the degree of "explanation" required for each edge (discussed below) and allows for easy extensions to more complex label types and extra features (Section 9), all of which make it our preferred choice for the softmax.

In a nutshell, our modeling assumption can be stated as follows: *It is better to explain as many friendships as possible, rather than to explain a few friendships really well.* Eq. 2 is maximized if the softmax function achieves a high value for each edge $u \sim v$, i.e., if each edge is "explained." This is achieved if the sum $\sum_{t\in\mathcal{T}} r(u,v,t)$ is relatively high, which in turn is satisfied if the product $S_{ut\ell}S_{vt\ell}$ is 1 for even one label ℓ — in other words, when there exists any label ℓ that both u and v share. The parameter α controls the degree of explanation needed for each edge; a small α forces the learning algorithm to be very sure that u and v share one or more label types, while with a large α, a single matching label type is enough. Empirical results shown later in Section 8 prove that large α values perform better (we use $\alpha = 10$ in our evaluation), suggesting that even a single matching label type is enough to explain the edge. The parameter c in Eq. 4 can be thought of as the probability of matching on an unknown label type, distinct from the five we consider. Higher values of c can be used to model uncertainty that the available label types form an *exhaustive* set of reasons for friendships. For our running example in the social network setting, we set $c = 0$ to reflect our belief that the five label types we consider represent the primary reasons for friendship formation (property **(P1)**).

Further intuition can be gained by considering a node u whose labels are completely unknown, but whose friends' labels are completely known (see Figure 1). As we discussed earlier in Section 1, label propagation would infer the hometown of u to be the most common hometown among her friends (i.e., H), the current city to be the most common current city among friends (i.e., C'), and so on. However, such an inference leaves u's friendships from C completely unexplained. Our proposed method rectifies this; Eq. 2 will be maximized by correctly inferring H and C as u's hometown and current city respectively, since H is enough to explain all friendships with the hometown friends, and the marginal extra benefit obtained from explaining these same friendships a little better by using C' as u's current city is outweighed by the significant benefits obtained from explaining all the friendships from C by setting u's current city to be C.

To summarize, Eq. 3 encapsulates property **(P1)** by trying to have matching labels between friends; Eq. 4 models property **(P2)** by enabling any one label type to explain each friendship; and the form of the probability distribution (Eq. 2) uses only existing edges $u \sim v$ and not all node pairs, and thus is not affected when, say, two nodes with Chicago as their current city are not friends, which reflects the idea that matching label types are necessary but not sufficient **(P3)**.

4. Inference via relaxation labeling

We present two methods for inference under EdgeExplain. The first considers a relaxation of the model that yields an objective which can be optimized via projected gradient descent, and this is discussed in the current section. The second method is optimizes a variational lower bound on the likelihood of EdgeExplain, and this is discussed later in Section 5. A

priori, there is little reason to choose one or the other, and we discuss their relative merits via simulations and evaluation on real data (Sections 7 and 8).

The probabilistic description of EdgeExplain in Eqs. 2-4 can be restated as an optimization problem in the variables $S_{ut\ell} \in \{0,1\}$. In the spirit of (Zhu et al., 2003), we propose a relaxation in terms of a real-valued function $\boldsymbol{f}$, with $f_{ut\ell} \in [0,1]$ representing the probability that $S_{ut\ell} = 1$, i.e., the probability that user u has label ℓ for label type t. This yields the following optimization:

$$\underset{\boldsymbol{f}}{\text{Maximize}} \sum_{u \sim v} \log\Big(\underset{t \in \mathcal{T}}{\text{softmax}}(r(u,v,t))\Big) \tag{5}$$

$$\text{where } r(u,v,t) = \sum_{\ell \in L(t)} f_{ut\ell} f_{vt\ell} \tag{6}$$

$$\sum_{\ell \in L(t)} f_{ut\ell} = 1 \quad \forall t \in \mathcal{T} \tag{7}$$

$$f_{ut\ell} \geq 0 \tag{8}$$

where softmax$(\cdot)$ is defined as in Eq. 4, and the equation for $r(\cdot)$ is analogous to Eq. 3 but measures the total probability that u and v have the same label for a given label type t.

The problem is not convex in $\boldsymbol{f}$, but is convex in $\boldsymbol{f}_u = \{f_{ut\ell} | t \in \mathcal{T}, \ell \in L(t)\}$ if the distributions $\boldsymbol{f}_v$ are held fixed for all nodes $v \neq u$. Hence, we propose an iterative algorithm to infer $\boldsymbol{f}$. Given $\boldsymbol{f}_v$ for all $v \neq u$, finding the optimal $\boldsymbol{f}_u$ corresponds to solving the following problem:

$$\underset{\boldsymbol{f}_u}{\text{Maximize}}\ g(\boldsymbol{f}_u) = \sum_{v \in \Gamma(u)} \log\Big(\underset{t \in \mathcal{T}}{\text{softmax}}(r(u,v,t))\Big),$$

where the summation is only over the set $\Gamma(u)$ of the friends of u, and we again restrict $\boldsymbol{f}_u$ to be a set of $|\mathcal{T}|$ probability distributions, one for each label type. We note that $g(\cdot)$ is convex and Lipschitz continuous with constant $L = \alpha \cdot |\Gamma(u)|$, where $|\Gamma(u)|$ is the number of friends of u.

This is a constrained maximization problem with no closed form solution for $\boldsymbol{f}_u$. To solve it, we use projected gradient ascent. This is an iterative method where in each iteration, we take a step in the direction of the gradient, and then project it back to the probability simplex $\Delta = \Big\{f_{ut\ell} \mid f_{ut\ell} \geq 0, \sum_{\ell \in L(t)} f_{ut\ell} = 1 \,\forall t \in \mathcal{T}\Big\}$. Specifically, let ∇g represent the gradient of g, with components given by:

$$\frac{\partial g(\boldsymbol{f}_u)}{\partial f_{ut\ell}} = \sum_{v \in \Gamma(u)} \alpha f_{vt\ell} \cdot \sigma\Big(-\alpha \sum_{t \in \mathcal{T}} \sum_{\ell \in L(t)} f_{ut\ell} f_{vt\ell} - c\Big).$$

Let $\boldsymbol{f}_u^{(k-1)} = \big\{f_{ut\ell}^{(k-1)} | t \in \mathcal{T}, \ell \in L(t)\big\}$ be the estimated probability distributions for each of the $\mathcal{T}$ label types at the end of iteration $k-1$, and let $q_{ut\ell}^{(k)}$ represent the (possibly improper) ending point of the k-th gradient step:

$$\boldsymbol{q}_u^{(k)} = \boldsymbol{f}_u^{(k-1)} + c_k \nabla g,$$

where c_k is a step-size parameter that we could set to a constant $c_k = 1/L$. The point $\boldsymbol{q}_u^{(k)}$ is now projected to the closest point in Δ:

$$\boldsymbol{f}_u^{(k)} = \underset{\boldsymbol{q}' \in \Delta}{\arg\min} \|\boldsymbol{q}_u^{(k)} - \boldsymbol{q}'\|_2.$$

This can be easily achieved in expected linear time over the size of the label set $\sum_t L(t)$ (Duchi et al., 2008). If only sparse distributions can be stored for each label type (say, only the top k labels for each type), the optimal k-sparse projections can be obtained simply by setting to 0 all but the top k labels for each label type, and then projecting on to the simplex (Kyrillidis et al., 2013).

This algorithm converges to a fixed point, and the function values converge to the optimal at a $1/k$ rate (Beck and Teboulle, 2009):

$$g^* - g^{(k)} \leq \frac{L\|\boldsymbol{f}_u^{(0)} - \boldsymbol{f}_u^*\|^2}{2k} \leq \frac{L|\mathcal{T}|}{k},$$

where $\boldsymbol{f}_u^*$ represents the optimal set of probability distributions, and g^* is the optimal function value. An important consequence of the algorithm is that computation of $\boldsymbol{f}_u$ only requires information from $\boldsymbol{f}_v$ for the neighbors v of u. Thus, it is a "local" algorithm that can be easily implemented in distributed message-passing architectures, such as Giraph (Giraph; Ching, 2013).

5. Variational inference

The relaxation labeling approach discussed above is intuitive, but its objective function is neither a bound nor a formal approximation of the likelihood of EdgeExplain. We now present a variational inference procedure where the inferred label probabilities do maximize a lower bound of the likelihood.

From Eqs. 2-4, we have:

$$\begin{aligned} P(S_V, S_H \mid \mathcal{G}) &= \frac{1}{Z} \prod_{u \sim v} \sigma\Big(\alpha \sum_{t \in \mathcal{T}} \sum_{\ell \in L(t)} S_{ut\ell} S_{vt\ell} + c\Big) \\ &= \frac{1}{Z} \prod_{u \sim v} \frac{1}{1 + \exp(-\alpha \sum_{t \in \mathcal{T}} \sum_{\ell \in L(t)} S_{ut\ell} S_{vt\ell} - c)}. \end{aligned}$$

Given a fixed assignment to the visible variables S_V and any distribution $Q(S_H)$ over the hidden variables, we have the inequality:

$$\ln P(S_V \mid \mathcal{G}) \geq -\sum_{S_H} Q(S_H) \ln \frac{Q(S_H)}{P(S_H, S_V \mid \mathcal{G})}. \tag{9}$$

We shall choose a fully factorized distribution for $Q(S_H)$:

$$Q(S_H) = \prod_u \prod_{t \in \mathcal{T}} \prod_{\ell \in L(t)} \mu_{ut\ell}^{S_{ut\ell}},$$

where $\mu_{ut\cdot}$ represents a multinomial distribution over all labels $\ell \in L(t)$ for label type t for user u; for notational convenience, we set $\mu_{ut\ell}$ to 0 or 1 if the user's labels are known. Such factored distributions are common in variational inference. They also have the same number of parameters as the relaxation labeling approach, allowing a fair comparison between the two. With this choice of distribution $Q(\cdot)$, the variational lower bound (Eq. 9) can be written as

$$\begin{aligned}\ln P(S_V \mid \mathcal{G}) \geq & -\sum_{ut\ell} \mu_{ut\ell} \ln \mu_{ut\ell} \\ & -\sum_{u\sim v} \left\langle \ln\left(1+\exp\left(-\alpha \sum_{t\ell} S_{ut\ell} S_{vt\ell} - c\right)\right)\right\rangle_{Q(S_H)} \\ & -\ln Z,\end{aligned} \tag{10}$$

where $\langle\cdot\rangle_Q$ represents expectation with respect to $Q(\cdot)$, and ℓ is summed over $L(t)$. We want to set the variational parameters $\boldsymbol{\mu} = \{\mu_{ut\ell}\ \forall u, t, \ell \in L(t)\}$ to maximize this lower bound.

Define $\eta_{uvt} = \sum_{\ell \in L(t)} \mu_{ut\ell}\mu_{vt\ell}$. Let $\mathbf{w} \in \{0,1\}^{|\mathcal{T}|}$ represent a binary vector of length $|\mathcal{T}|$, with w_t being the t^{th} component and $|\mathbf{w}|$ the number of "ones".

Theorem 1 *Consider one node u and one label type t. Given the parameters $\boldsymbol{\mu} \setminus \{\mu_{ut\cdot}\}$, the distribution $\mu^*_{ut\ell}$ that maximizes the lower bound of Eq. 10 is given by:*

$$\mu^*_{ut\ell} \propto \exp\left\{\sum_{\{v|u\sim v\}} \mu_{vt\ell} \sum_{\mathbf{w}} \phi_{uvt}(\mathbf{w})\right\}$$

$$\phi_{uvt}(\mathbf{w}) = \ln\left(1+e^{-(\alpha|\mathbf{w}|+c)}\right)(-1)^{w_t} \prod_{t'\neq t} \kappa(w_{t'}, \eta_{uvt'})$$

$$\kappa(w_{t'}, \eta_{uvt'}) = \eta_{uvt'}^{w_{t'}}(1-\eta_{uvt'})^{1-w_{t'}}.$$

Proposition 1 $\sum_{\mathbf{w}} \phi_{uvt}(\mathbf{w}) \geq 0$.

Both proofs are deferred to Appendix A.

It follows that the probability of node u having label ℓ for type t under the variational approximation is given by a (normalized, exponentiated) weighted linear combination of the neighboring node labels. For intuition, consider the case of large α. Then, the sum $\sum_{\mathbf{w}} \phi_{uvt}(\mathbf{w})$ is dominated by the case of $\mathbf{w}^{(-t)} = \mathbf{0}$, the all-zero vector:

$$\sum_{\mathbf{w}} \phi_{uvt}(\mathbf{w}) \approx \ln\left(\frac{1+e^{-c}}{1+e^{-\alpha-c}}\right) \prod_{t'\neq t}(1-\eta_{uvt'}).$$

Thus, the highest-probability label of type t for u is the most the common label among friends of u, *unless* the edge $u \sim v$ is already explained by some other label type t', i.e., if $\eta_{uvt'} \approx 1$. Hence, the probabilities $\mu^*_{ut\ell}$ attempt to explain as many friendship edges as possible, which is precisely the expected behavior for large α. Our variational inference method iteratively updates the parameter vector $\boldsymbol{\mu}$ by repeatedly applying Thm. 1.

6. Analysis

We now present a theoretical analysis of EdgeExplain, and its relationship with Label Propagation (LP). Even if node labels for different label types are dependent (as in EdgeExplain) rather than independent (as in LP), the labels inferred by LP could still be correct. We will find the conditions under which LP-based inference yields incorrect results, and hence inference tailored to EdgeExplain can outperform LP.

Simplified model. We will make several simplifications to EdgeExplain to focus on its main aspects. First, we will consider the "hard-threshold" variant of the problem, where even one shared attribute between two friends is enough to explain their friendship; this corresponds to setting $\alpha \to \infty$ in Eq. 4. Such a choice is also justified empirically, as will be shown in Section 8. Second, we will consider the case of one node (the "ego") whose attributes are all unknown, surrounded by N friends whose attributes are all known. This can be thought of as the fundamental inference problem template, with the actual problem using a softer version of this template that associates probabilities with the labels of friends. Finally, Eqs. 2–4 only specify a discriminative model, but we will need a generative model for our analysis. Hence, we devise a simple probabilistic model for generating node labels (represented by the binary variables $\boldsymbol{S}$) that is consistent with properties **(P1)**-**(P3)** outlined in Section 3.

Specifically, consider an "ego network" $\mathcal{G}_u$ consisting of a central node u surrounded by N friends $v_1, \ldots, v_N$. Let $Y_u = \{Y_u(t_1), \ldots, Y_u(t_{|\mathcal{T}|})\}$ denote the labels of u for each of the $|\mathcal{T}|$ label types; these are precisely the labels that are turned "on" in the binary indicator variables $\boldsymbol{S}$ restricted to u. Similarly, let Y_i represent the vector of labels for node v_i. Let $\pi(Y_u, Y_{v_1}, \ldots, Y_{v_N})$ denote the probability of observing these labels. Since $\mathcal{G}_u$ is a tree rooted at u, the labels of the friends are conditionally independent given the labels of the ego: $\pi(Y_u, Y_{v_1}, \ldots, Y_{v_N}) = \pi(Y_u) \cdot \prod \pi(Y_i \mid Y_u)$. Thus, we can generate node labels in the following manner. First, the ego u selects her labels first according to the prior $\pi(Y_u)$. Then, conditioned on Y_u, the labels Y_i for friends v_i are drawn independently of each other. In order to satisfy **(P1)**, this second step must itself be split into two stages. In the first stage, each friend v_i selects a "reason" for her friendship with u by selecting the label type Z_i that v_i shares with u. This shared label type is drawn according to a multinomial distribution q: $q(Z_i = t) = q_t$. Thus, $Y_i(Z_i) = Y_u(Z_i)$ with probability 1. Then, in the second stage, the remaining labels of v_i are set via the distribution $\pi(Y_i \mid Y_i(Z_i))$. Under this setting of Y_u and Y_i, there is a shared label for each edge, and the model π is the most general one that satisfies **(P1)** and the independence assumptions encoded in $\mathcal{G}_u$, and is symmetric across all friends.

This leads to the following simplified problem statement:

Given the network $\mathcal{G}_u$ and the labels Y_i of all friends v_i, predict the labels Y_u of the ego.

Relation to set-cover. At first sight, this appears to be a variant of the well-known set-cover problem where, for each node, we must pick sets of friends sharing some attribute such that the union of these sets covers all the friendships. However, note that we can pick only one set for each label type, since the ego can have only one label for each type Thus, the computational complexity is $O\left(\prod_{t\in\mathcal{T}} |\mathcal{L}_t|\right)$, and since we consider only a limited

number of label types ($|\mathcal{T}| = 5$), the complexity is polynomial. The set-cover problem, on the other hand, can consider any number of sets, and is an NP-hard problem.

Failure conditions for Label Propagation. The correct labels Y_u for the ego offer (one of) the best solutions for EdgeExplain, since each friendship is explained by at least one shared label. LP fails if the ego has a label ℓ of type t (i.e., $Y_u(t) = \ell$), but a different label $\ell' \neq \ell$ of the same type t is shared by more friends; we will call this the event *"LP fails via* (t, ℓ, ℓ')*."* Thus, $P(\text{LP fails}) \geq \max_{t,\ell,\ell'} P(\text{LP fails via } (t, \ell, \ell'))$, and we will develop lower bounds for the latter quantity.

Let us denote by $p_{t,t'}(\ell)$ the probability of friend v_i having the correct label ℓ for label type t given that it shares some other label type t' with the ego u; as discussed earlier, this probability is assumed to be the same for all friends:

$$p_{t,t'}(\ell) \triangleq \pi(Y_i(t) = \ell \mid Y_u \text{ and } \{Z_i = t' \neq t\}).$$

Analogously, let $p_{t,t'}(\ell')$ denote the probability of friend v_i having an incorrect label ℓ'. Furthermore, let $\Delta_{t,\ell,\ell'}$ denote the expected difference of $p_{t,t'}(\ell')$ and $p_{t,t'}(\ell)$ w.r.t. q:

$$\Delta_{t,\ell,\ell'} \triangleq \sum_{t' \neq t} q_{t'} \left(p_{t,t'}(\ell') - p_{t,t'}(\ell) \right) - q_t. \tag{11}$$

Theorem 2 *If $\Delta_{t,\ell,\ell'} > 0$, then for any small ϵ such that $0 < \epsilon < \Delta_{t,\ell,\ell'}$, we have:*

$$P(\textit{LP fails via } (t, \ell, \ell')) \geq \sum_{\{Y_u | Y_u(t) = \ell\}} \pi(Y_u) \cdot \left(1 - \exp\{-0.5N(\Delta_{t,\ell,\ell'} - \epsilon)^2\}\right).$$

The proof is deferred to Appendix A.

Thus, LP is likely to fail when $\Delta_{t,\ell,\ell'}$ is large. This happens when the following two conditions hold: (a) label ℓ is somewhat less likely than ℓ' in the entire population (so that $p_{t,t'}(\ell') - p_{t,t'}(\ell) > 0$), and (b) friendships based on a shared label for label type t are rare (i.e., q_t is small and consequently $q_{t'}$ can be large). Intuitively, under these conditions, the ego can have label ℓ for a rarely shared label type t (say, employer), but enough friends can have a different label ℓ' simply due to its prevalence in the population. It becomes likely that the number of friends with label ℓ' is more than those with ℓ, leading to an erroneous inference by LP. This is precisely the situation illustrated in Figure 1 earlier.

Optimum conditions for EdgeExplain. We shall now derive the conditions which imply the greatest possibility of error by LP. Consider a system with only two label types t and t' (call this TwoLabels). Thus, $q_{t'} = 1 - q_t$. Since LP's probability of failure is monotonically increasing in $p_{t,t'}(\ell')$ (via Eqs. 11 and Thm. 2), the worst case for LP is when all the mass is concentrated in a single alternate label ℓ'. Then without any loss of generality, we can let label type t take only two values ℓ and ℓ', with $p_{t,t'}(\ell') = 1 - p_{t,t'}(\ell)$. We will assume that the labels for the ego and her friends follow the same marginal distribution, so

$$\pi(Y_u(t) = k \mid Y_u(t')) = \pi(Y_i(t) = k \mid Y_u(t') \text{ and } \{Z_i = t'\}) \triangleq p_{t,t'}(k) \qquad \text{for } k \in \{\ell, \ell'\}.$$

Theorem 3 *The lower bound in Theorem 2 under* TwoLabels *is maximized for*

$$\Delta_{t,\ell,\ell'} = O\left(\sqrt{\frac{\log N}{N}}\right), \qquad p_{t,t'}(\ell) = \frac{1 - 2q_t - \epsilon}{2(1 - q_t)} - O\left(\sqrt{\frac{\log N}{N}}\right), \qquad q_t < 0.5.$$

Corollary 4 *For a given N, the optimum conditions for* EdgeExplain *vis-a-vis LP under* TwoLabels *are obtained for $p_{t,t'}(\ell) + q_t - p_{t,t'}(\ell) \cdot q_t \approx const.$*

Both proofs are deferred to Appendix A.

Theorem 3 demonstrates the link between the probability $p_{t,t'}(\ell)$ of a person having label ℓ and the probability q_t of forming a friendship based on a shared label of type t. If $p_{t,t'}(\ell)$ is too large, then it becomes very unlikely that another label ℓ' can be shared by more friends than ℓ. Conversely, if $p_{t,t'}(\ell)$ is too small, the ego will rarely have label ℓ, so there will be fewer situations where LP fails. Setting $p_{t,t'}(\ell) \approx (1 - 2q_t)/(2(1 - q_t))$ achieves the optimal balance between these two. Alternatively, for small $p_{t,t'}(\ell)$ and q_t, the optimum linearly trades off $p_{t,t'}(\ell)$ and q_t (Corollary 4).

A second point of interest is the effect of the number of friends N: the optimal $p_{t,t'}(\ell)$ increases with increasing N. This can be explained by noting that as N increases, any difference in the expected popularities of ℓ' and ℓ becomes more likely to be reflected in their observed frequencies, and hence even a small expected difference $\Delta_{t,\ell,\ell'}$ is sufficient to create a situation where LP fails. However, when N is small, the effect of randomness is greater, and hence a smaller $p_{t,t'}(\ell)$ is needed to ensure that ℓ' is observed more often among the friends than ℓ.

7. Simulations

We seek answers to three questions from our simulation runs: (a) how can we simulate networks according to EdgeExplain, (b) how well do the various inference algorithms (relaxation labeling and variational) perform on such networks, and (c) how does the accuracy of label propagation vary with model parameters?

7.1 Simulating EdgeExplain

This closely follows the TwoLabels setting described in Section 6, i.e., we use two label types t and t'. Generating the simulation graph consists of three stages: (a) label generation, (b) edge generation, and (c) hiding labels.

Label generation: We generate small *neighborhoods* consisting of one node (the "ego") and her friends (the "alters", whose number N is a parameter of the simulation), as follows. First, the ego u selects her labels Y_u from a predefined joint distribution $\pi(Y_u)$ over all labels of all label types. Next, the ego selects, for each friend $v_1, \ldots, v_N$, the label type Z_i that explains their friendship; Z_i is picked from a distribution q over the set of label types. Clearly, this sets the corresponding label $Y_i(Z_i) = Y_u(Z_i)$. Conditioned on this, the labels of v_i for the other label type are given by $p_{t,t'}(\ell) \triangleq \pi(Y_i(t) \mid Y_u \text{ and } Z_i = t' \neq t) = \pi(Y_i(t) \mid Y_i(t') = Y_u(t'), Z_i = t' \neq t)$. We set $p_{t,t'}(\ell) = \pi(Y_i(t) = \ell)$.

Edge generation: To make the model more realistic, we introduce inter-friend links in addition to the links between the ego u and all her friends. For each pair of friends sharing at least one label type, a link is added between them with an inter-friend link probability p_{if}.

Hiding labels: Next, we hide some labels to create the actual train/test instances. All labels of the ego u are hidden. In addition, for each of the N friends, with probability p_h, all labels are hidden. This creates an ego network where some nodes have full information and others have no information. The goal is to infer the hidden labels of u from the network structure and the known labels of some friends.

Parameter choices: In order to create the simplest instance where parameter tradeoffs can be investigated, the labels and their probabilities are set as follows. We allow two possible labels for t, with marginal probabilities p and $1-p$ respectively. We allow three possible labels for t', with marginal probabilities fixed at 0.3, 0.3, and 0.4. Since there are only two label types, the distribution $q(Z_i)$ over label types can be represented via just one number q, with $q_t = q$ and $q_{t'} = 1-q$. Thus, each train/test instance can be indexed by the 4-tuple (p, q, p_{if}, p_h).

We set $N = 1,000$ and run 50 repetitions for each simulation setting. For each simulation, we check if the right label for label type t was inferred, since this is the label type whose label probabilities are varied via the parameter p. The average fraction of correct inferences over all simulations gives the accuracy of the inference technique. The parameter range is restricted to $q < 0.5$ in accordance with Thm. 3; for $q > 0.5$, label type t is shared most often with friends, so none of the inference algorithms face any difficulty. We also keep $p < 0.5$, since the case of $p > 0.5$ is symmetric (we do not differentiate between the two labels for type t).

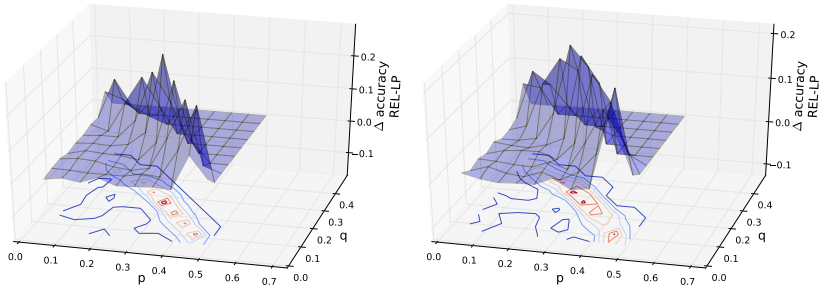
We note two points. First, the only test instance is the ego node u; friends with hidden labels are not considered in the test set since they might have other friends not present in the ego network of u. Second, while we could extend the graph generation process by adding friends of friends of u, the primary driver of inference accuracy is the information available among friends of u. Thus, the ego network with hidden node labels is the basic structure on which the accuracy of inference can be tested.

7.2 Comparison of inference algorithms

We will compare the accuracies of the three inference methods seen so far: relaxation labeling (REL), variational inference (VAR), and label propagation (LP).

Comparison against LP. Figure 2 shows the difference in accuracy between REL and LP as the parameters p, q, and p_h change, with $p_{if} = 0.3$ (the plots are almost identical with $p_{if} = 0$ and $p_{if} = 0.5$). We can make the following observations:

- REL is always at least as accurate as LP, and often much more accurate. Thus, under EdgeExplain, REL dominates LP.
- The parameter settings for which REL outperforms LP the most lie almost on a line in the pq-plane. This corresponds to $p + q \approx$ const., which agrees with Corollary 4.
- The results are unaffected by the probability of hiding labels and the probability of having links between friends. Hence, the difference between EdgeExplain inference



(a) $p_h = 0.0$ (b) $p_h = 0.5$

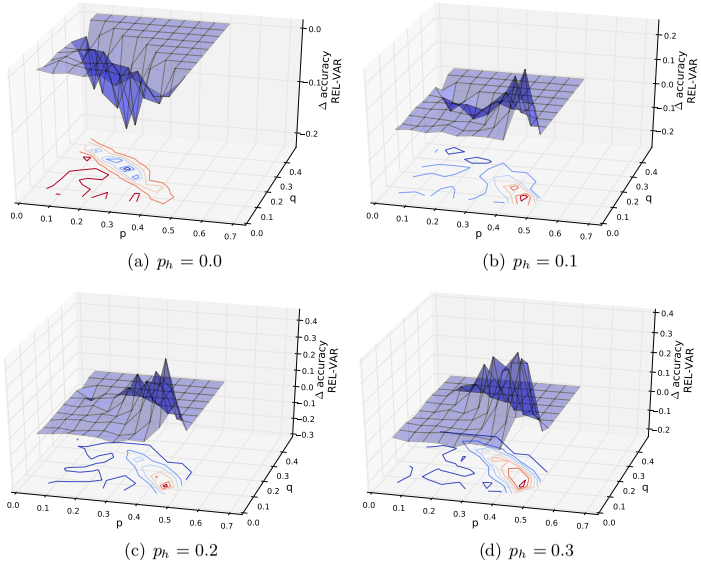
Figure 2: REL *outperforms* LP*:* The REL inference method under EdgeExplain outperforms LP irrespective of the fraction of friends whose labels are hidden. The greatest difference is when $p + q \approx$ const., agreeing with Corollary 4.

and label propagation is primarily a function of the relative probabilities of the various labels (the p) and the chances of various label types being used for forming friendships (the q); again, this is what we saw in Section 6.

Comparison of REL and VAR. Figure 3 compares REL against VAR. We observe the following:

- Although both are algorithms for inference under EdgeExplain, they perform better under different parameter regimes. When the labels of all friends are known perfectly ($p_h = 0$), VAR performs better than REL. However, as p_h increases, information in the neighborhood of the ego becomes more uncertain, and REL performs better for high p_h. The transition point is around $p_h \approx 0.1$, i.e., when 90% of friend labels are known.
- The greatest difference is in the same region of the pq-plane where REL outperformed LP the most.
- With p_h fixed, the greatest *magnitude* of difference occurs for larger values of p.
- Indeed, for large p_h, VAR performs worse than even LP for large p and small q (Figure 4).

The underlying reason is the form of the variational update (Thm. 1), in particular, in the exponentiation which serves to magnify even minor differences in its argument. When p_h is high, there are more errors in labels assigned to friends with missing information. This leads to incorrect estimation of the chances that two nodes share a label (the parameter η_{uvt} in Theorem 1). This error gets compounded by the exponentiation, causing a quick convergence to erroneous locally-optimum labels in VAR.



(a) $p_h = 0.0$ (b) $p_h = 0.1$

(c) $p_h = 0.2$ (d) $p_h = 0.3$

Figure 3: REL *versus* VAR*:* REL is better when more friends hide their labels ($p_h \geq 0.1$). However, when labels of all friends are known ($p_h = 0$), VAR is better.

The problem is even more acute when p is high and q is low: the low q means that label type t is rarely shared among friends, while high p means that both possible labels of type t are common (recall that $p < 0.5$, so a high p corresponds to $p \approx 0.5$). This implies greater chances of mistakes in calculating η_{uvt} in the initial iterations, which is exacerbated by the exponentiation.

However, the positive side of exponentiation is that when friend labels are not noisy (as when $p_h = 0$), VAR converges quickly to the correct labels, while REL may get stuck in local minima during its gradient ascent. This explains why VAR is better for low p_h.

7.3 A hybrid inference algorithm

The previous discussion suggests the following hybrid inference procedure: in any iteration, for any node u, update its label probabilities via VAR if we are quite confident about the labels of most of the friends of u, otherwise use REL for the update. In effect, this tries to use VAR and REL precisely where they worked best in the previous simulations.

Thus, HYBRID counts the fraction of friends of u whose labels are "nearly certain" (for each label type, at least one label has probability greater than θ_1, normalized by the

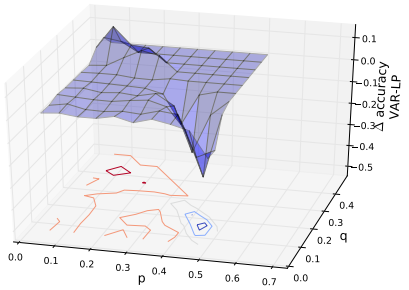


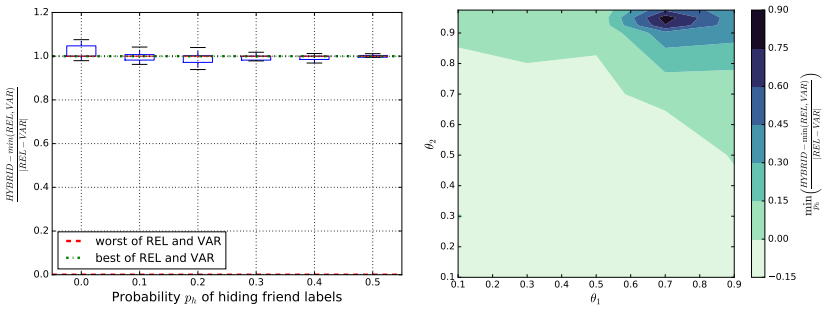
Figure 4: VAR *versus* LP *for one parameter setting:* With $p_h = 0.3$ and $p_{if} = 0.3$, VAR is better than LP for low p , and worse for high p.

Algorithm 1 HYBRID

function HYBRID(initial label-probs, θ_1, θ_2)
  repeat
    for all nodes u **do**
      L_u = number of distinct labels in $\bigcup_{u \sim v}$ label-probs$_t(v)$
      nearly-certain-friends = $|\{v | u \sim v, \max(\text{label-probs}_t(v)) > 2\theta_1/|L_u| \ \forall t \in \mathcal{T}\}|$
      all-friends = $|\{v | u \sim v\}|$
      if nearly-certain-friends $> \theta_2 *$ all-friends **then**
        Update label-probs$_t(u)$ via VAR
      else
        Update label-probs$_t(u)$ via REL
      end if
    end for
  until convergence or maximum iterations
  return label-probs(u)
end function

number of labels in the neighborhood) and applies the variational update (VAR) if this fraction is more than θ_2, otherwise the relaxation labeling update (REL) is used.

Figure 5 compares the accuracy of HYBRID to REL and VAR. Plot (a) shows, for $(\theta_1 = 0.7, \theta_2 = 0.95)$, the lift in accuracy of HYBRID against the others; a lift of 0 corresponds to accuracy equivalent to the worst of the two, while a lift of 1 corresponds to the best of the two. We see that HYBRID is in fact as good as the best of them for all p_h (recall that VAR was better for small p_h, and REL elsewhere). The results are very similar when the probability p_{if} of inter-friend edges is varied. Plot (b) shows a contour plot of the worst-case lift (i.e., the minimum lift among all p_h) over the (θ_1, θ_2) range. While the best setting is $(\theta_1 = 0.7, \theta_2 = 0.95)$, we note that performance is quite robust even without picking the precise maximum.



(a) Lift of HYBRID (b) Lift as a function of (θ_1, θ_2)

Figure 5: HYBRID *performs as well as the best of* REL *and* VAR*:* (a) The lift of HYBRID (with $\theta_1 = 0.7$ and $\theta_2 = 0.95$) over the minimum of REL and VAR is plotted against p_h (restricted to parameter settings where REL and VAR differ in accuracy by at least 0.01); a lift of 0 corresponds to no improvement over the worst of the two, while 1 corresponds to the performance of the better of the two. HYBRID is seen to provide better performance than REL or VAR over the entire range of p_h. (b) The minimum lift over all p_h is plotted for various (θ_1, θ_2) settings, with $\theta_1 = 0.7$, $\theta_2 = 0.95$ performing the best.

8. Empirical evaluation

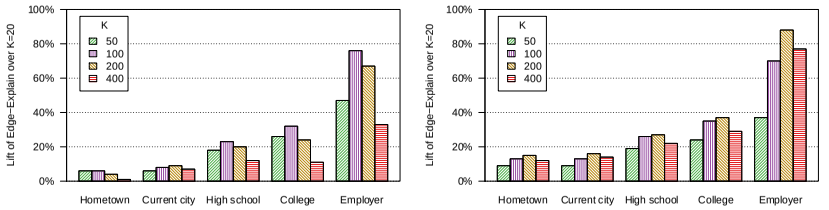
Previously, we provided intuition and examples supporting the claim that EdgeExplain is better suited to inference of our desired label types than label propagation. In this section, we demonstrate this via empirical evidence on two large social network data sets from Facebook and Google+, and a movie network.

8.1 Evaluation on the Facebook network

We performed a study on a previously collected subgraph of the Facebook social network (Chakrabarti et al., 2014). [1] This data set consists of a large number of users and their friendship edges, as well as the hometown, current city, high school, college, and employer for each user, whenever these fields are available and have their visibility set to public. The dimensionality of our five label types range from almost $900K$ to over $6.3M$. We describe below in Implementation Details our process for generating the edges. This forms our base data set.

Evaluation Methodology. The set of users is randomly split into five parts and the accuracy is measured via 5-fold cross-validation, with the known profile information from four folds being used to predict labels for all types for users in the fifth fold. Results over the

1. We worked on a snapshot of data, and there was no interaction with users or their experience on the site.



(a) Recall at 1 (b) Recall at 3

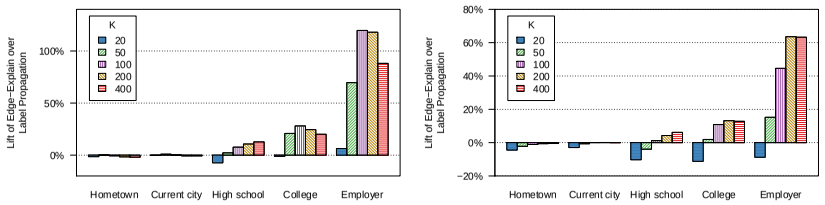
Figure 6: *Recall of* EdgeExplain *for graphs built with different number of friends K:* The plot shows lift in recall with respect to a fixed baseline of EdgeExplain with $K = 20$. Increasing K increases recall up to a point, but then the extra friends introduce noise which hurts accuracy.

various folds are identical to three decimal places. All differences are therefore significant and we do not show variances as they are too small to be noticeable.

For each evaluation, we run inference on the training set and compute a ranking of labels for each label type for each user. This ranking is provided by the label probabilities computed by each label inference method. We then measure recall at the top-1 and top-3 positions, i.e., we measure the fraction of (user, label type) pairs in the test set where the predicted top-ranked label (or any of the top-3 labels) match the actual user-provided label. For reasons of confidentiality, we only present the *lift* in recall values with respect to a clearly specified baseline.

Implementation Details. We implemented EdgeExplain in Giraph (Ching, 2013) which is an iterative graph processing system based on the Bulk Synchronous Processing model (Malewicz et al., 2010; Valiant, 1990). The entire set of nodes is split among 200 machines, and in each iteration, every node u sends the probability distributions ($f_{ut\ell}$ for REL, $\mu_{ut\ell}$ for VAR) to all friends of u. To limit the communication overhead, we implemented two features: (a) for each user u and label type t, the label distribution for each (user, type) pair was clipped to retain only the top 8 entries optimally (Kyrillidis et al., 2013), and (b) the friendship graph is sparsified so as to retain, for each user u, the top K friends whose ages are closest to that of u. This choice of friends is guided by the intuition that friends of similar age are most likely to share certain label types such as high school and college. We find that clipping the distributions makes little difference to accuracy while significantly improving running time. However, the value of K matters significantly, and we detail these effects next.

Accuracy of EdgeExplain. We first show how recall varies as a function of the number of friends K. In the following, EdgeExplain with REL inference is used unless noted otherwise; comparisons with VAR are shown later. Figure 6 shows recall versus K, with recall at $K = 20$ being the baseline. We observe that recall increases up to a certain K and then decreases. The optimal value is $K = 100$ for recall at 1, and $K = 200$ for recall at 3. This demonstrates both the importance and the limits of scalability: increasing the



(a) Recall at 1 (b) Recall at 3

Figure 7: *Lift of* EdgeExplain *over label propagation:* Increasing the number of friends K benefits EdgeExplain much more than label propagation for high school, college, and especially employer.

number of friends enables better inference but beyond a point, more friends increase noise. Thus, $K \approx 100$ friends appear to be enough for inference under EdgeExplain.

Figure 6 also shows an increasing trend from hometown to employer in the degree of improvement obtained over the $K = 20$ baseline. This is because (a) the baseline itself is best for hometown and worst for employer, but also because (b) Facebook users appear to have many more friends from label types other than from their current employer. The effect of this latter observation is that if we only have a small K, it is very likely that the few friends from the same current employer are not included in that limited set of friends (which we empirically verified). As K increases and such same-employer edges become available, EdgeExplain can easily learn the reason for these edges (hence the dramatic increase in recall), but label propagation will likely be confused by the overall distribution of different employers among all friends and therefore does not benefit equally from adding more friends, as we show next.

Comparison with Label Propagation. Figure 7 shows the lift in recall achieved by EdgeExplain over label propagation as we increase K for both. We observe similar performance of both methods for hometown and current city, but increasing improvements for high school, college, and employer. This again points to the first two label types being easier to infer, while the latter three are more difficult. With fewer employer-based friendships, the prototypical example of Figure 1 would also occur frequently, with label propagation likely picking common employers of (say) hometown friends instead of the less common friendships based on the actual employer. By attempting to explain each friendship, EdgeExplain is able to infer the employer even under such difficult circumstances. This ability to perform well even for under-represented label types makes EdgeExplain particularly attractive.

Effect of α. Figure 8 shows that the lift in recall at 1 for various values of the parameter α, with respect to $\alpha = 0.1$. Performance generally improves with increasing α. Results for recall at 3 are qualitatively similar, though the effect is more muted. We find that $\alpha \in [10, 40]$ offer the best results, and EdgeExplain is robust to the specific choice of α within this range. Recall that with large α, a single matching label is enough to explain an edge,

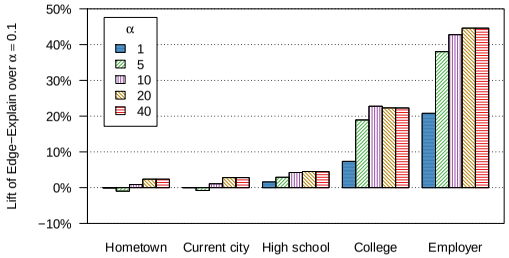
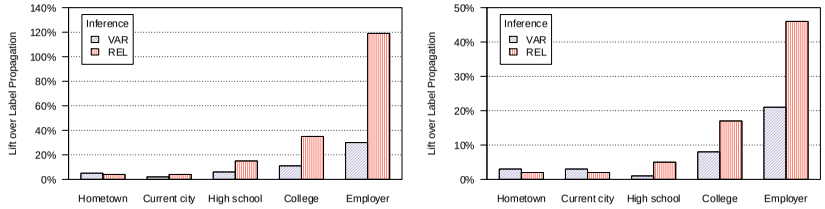


Figure 8: *Effect of α:* Lift in recall at 1 is plotted for different values of α, with respect to $\alpha = 0.1$. The best results are for $\alpha \in [10, 40]$.



(a) Recall at 1 (b) Recall at 3

Figure 9: *Comparison of* VAR *and* REL*:* The lift of each inference method over label propagation propagation is plotted for $K = 100$. REL is seen to outperform VAR, and both are better than LP.

while with small α, multiple matching labels may be needed. Thus, the outperformance of large α provides empirical validation of property **(P2)** (on our network).

Variational inference versus Relaxation Labeling. Our two main methods for inference under EdgeExplain are REL and VAR. Recall that while both use per-node parameters, they attempt to optimize Eqs. 2–4 in different ways. REL relaxes the problem by replacing the hidden variables $S_{ut\ell}$ by their "probabilities" $f_{ut\ell}$, but the relaxed objective is not formally related to the probabilistic model. On the other hand, VAR formally optimizes a lower-bound on the model log-likelihood, but the lower bound may not be very tight (we note, however, that this is the standard variational assumption when per-node parameters are needed).

Figure 9 compares the two inference methods. The lift in recall of each method over LP is shown, with $K = 100$. While both outperform LP, REL is more accurate than VAR. This agrees with our simulation results, where VAR outperformed REL only in a narrow parameter regime, where the labels of most friends of a given node are known with

Table 1: *Lift in recall from using group memberships:* Inclusion of group membership barely improves recall@3, even though it is an orthogonal feature with wide coverage. Thus, information about label types is already encoded in the network structure, and careful modeling via EdgeExplain is sufficient to extract it.

Label Type	Recall at 1	Recall at 3
Hometown	−0.1%	0.7%
Current city	0.4%	1.0%
High school	0.1%	0.8%
College	−0.6%	1.0%
Employer	−2.8%	1.2%

high confidence. Such instances are unlikely in social networks, contributing to the relative weakness of VAR.

Given the better performance of REL on this data set, the following results in this section use REL.

Inclusion of extra features. EdgeExplain easily generalizes to broader settings with multiple user features, such as group memberships, topics of interest, keywords, or pages liked by the user. As an example, consider group memberships of users. Intuitively, if most members of a group come from the same college, then it is likely a college-friends group, and this can aid inference for group members whose college is unknown. This can be easily handled by creating a special node for each group, and creating "friendship" edges between the group node and its members. EdgeExplain will infer labels for the group node as well, and will explain its "friendships" via the college label. This, in turn, will influence college inference for group members with unknown college labels. Group memberships are extensive and provide information that is orthogonal to friendships; hence, *a priori*, one would expect the addition of group membership features to have significant impact on label inference.

Table 1 shows the lift in recall for EdgeExplain when group memberships are used in addition to $K = 100$ friends. While the addition of group memberships increases the size of the graph by $\approx 25\%$, the observed benefits for recall are minor: a maximum lift of only 1.2% for employer inference, and indeed reduced recall at 1 for several label types. Note that the lift in recall would have appeared very significant had we compared it to label propagation with $K = 100$; however, this gain largely disappears when the friendships are considered in the framework of EdgeExplain. Thus, it is not merely the scalability of EdgeExplain, but also the careful modeling of properties **(P1)-(P3)** that makes group membership redundant.

Given the *a priori* expectations of the impact of group memberships, this surprising result suggests that information regarding our label types are already encoded in the structure of the social network and hence the orthogonal information from the group memberships actually turns out to be redundant.

The limits of resolution. Our model theoretically should be able to handle any number of label types, but empirically this may not hold true for our network. How many friends

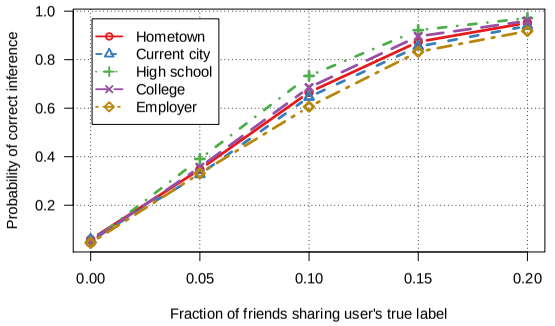


Figure 10: *Probability of correctly inferring (in the top-3) the value of a given label type t for a user, given the fraction of friends with known label for t who actually share the user's label for t:* All label types are broadly similar, with a fraction of 0.1 usually being sufficient for inference. For fraction > 0.2, the plot flattens out.

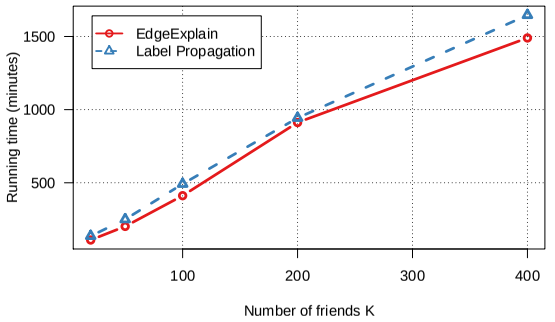
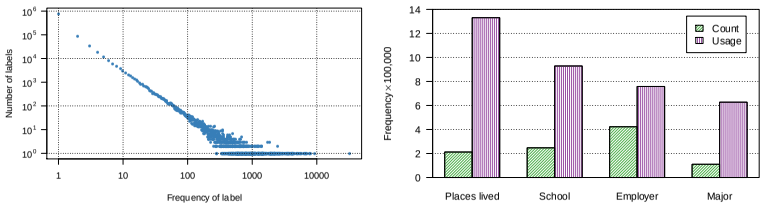


Figure 11: Running time increases linearly with K.

sharing a certain label type (say, the same college) does a user need to have in order to correctly infer the value of that label type? To answer this, we select, for each user, the set of friends whose label for the given label type t is known, and we compute the fraction that actually shared the user's label for t. Figure 10 shows the probability that EdgeExplain correctly infers the user's label as a function of this fraction (i.e., the correct label is among the top 3 predictions). All label types are similar, though high school is somewhat easier and employer harder; having $10 - 15\%$ of friends sharing a user's label is sufficient to infer the label in our graph. Note that certain label types are more likely to be publicly declared than others, and this explains differences in recall observed earlier.



(a) Label usage frequency distribution (b) Number of labels and label usage for each type

Figure 12: Descriptive statistics for Google+.

Running time. Figure 11 shows the wall-clock time as a function of K. The running time should depend linearly on the graph size, which grows almost linearly with K; as expected, the plot is linear, with deviations due to garbage collection stoppages in Java.

8.2 Evaluation on the Google+ network

We also conducted empirical evaluation on data collected from the Google+ social network (Gong et al., 2012). The data set consists of several snapshots of the network, of which we picked the oldest snapshot for our evaluation. This snapshot contains $4,693,129$ users connected by $47,130,325$ links. In addition, the data lists four attributes of each user (if the user has provided this information publicly): their employer, school, major, and the places where they have lived. There are $991,545$ labels, and $4,443,631$ (user, label) pairs in the data set. Gong et al. (2012) compared several methods for attribute inference on this data set, and found that a close variant of Label Propagation (which they called CN-SAN) has the highest precision. Hence, we only compare against Label Propagation.

We first note the extreme sparsity of the data set. There are only 10 friends per person on average, and only 1 label per person. In fact, only $909,669$ people have any labels associated with them at all (i.e., only 19% of the total). The number of people who declared all four of their label types is only $123,023$, or 2.6% of all people. Figure 12 shows some descriptive statistics regarding the labels. The distribution of labels among users (plot (a)) demonstrates a power-law: a few labels are extremely common, while the vast majority occur only once. Plot (b) shows the number of labels, and the usage, for each label type. While the most commonly used labels refer to "places lived," the "employer" label type has the most distinct labels. In other words, information about employers is particularly sparse, as in the Facebook data set.

Figure 13 shows the accuracy of each inference algorithm on all four label types. We see that there is practically no difference between the algorithms. This is due to the sparsity of the data; with so little information, there is hardly enough information among friends for EdgeExplain to differentiate itself from unrestricted homophily, and the most common label among friends is often the only reasonable choice for REL, VAR, or HYBRID (and of course for LP).

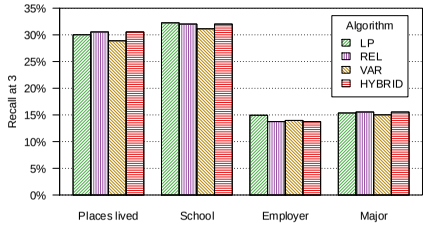


Figure 13: *Recall on the Google+ data set:* All algorithms perform similarly, primarily because the data set is too sparsely labeled.

8.3 Evaluation on a Movie Network

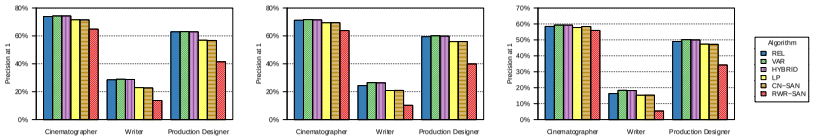
We have so far compared EdgeExplain to Label Propagation (LP), which has been shown to be among the best predictors of node attributes in previous work (see Section 2). We have shown that EdgeExplain works as well as LP on the sparse Google+ data set, and significantly outperforms it on the Facebook data set. Now, we will perform a broader comparison of EdgeExplain against LP as well as other competing methods on a movie network.

We construct the network as follows. Each node is an English-language movie, and has three label types: the writer, the cinematographer, and the production designer.[2] These three figures represent critical behind-the-scenes roles that support the movie's director.[3] We only keep movies with a single label for each label type (i.e., no "joint writers" etc.), so as to simplify testing of prediction accuracy. There are $23,921$ such movies. Two movies are connected by an edge if they have the same writer, or cinematographer, or production designer (they could share several of these label types); note that connecting nodes to their "nearest-neighbors" is a standard technique for converting feature-based data sets into networks, and has been used for LP as well (Zhou et al., 2003). There are $100,098$ such movie pairs that share cinematographers, $20,560$ pairs sharing writers, and $84,328$ pairs sharing production designers. There are $189,828$ edges overall. Thus, by construction, the network structure reflects shared labels, satisfying the basic assumption upon which all label inference methods rest.

We shall compare all inference methods for EdgeExplain (i.e., REL, VAR, and HYBRID) against LP. In addition, we test three algorithms that have been recommended by prior work:

- LINK (Zheleva and Getoor, 2009), where each node is represented by a feature vector encoding the IDs of its neighboring nodes, and a standard classifier is used to predict labels from the feature vector. Given the size of the feature vector ($23,921$ binary

2. The data is available at `ftp://ftp.fu-berlin.de/pub/misc/movies/database/`.
3. We have also experimented with other label types such as director and composer. All results are qualitatively similar and exhibit the same trends.



(a) 5% unlabeled nodes (b) 20% unlabeled nodes (c) 50% unlabeled nodes

Figure 14: Precision@1 for various label inference methods on the movie network.

features) and the size of the output labels (39, 317 labels in total), we choose Naive Bayes as the classifier.

- CN-SAN (Gong et al., 2014), which assigns to each unlabeled node the most popular label of each label type among the node's neighbors, and iterates this process until convergence.

- RWR-SAN (Yin et al., 2010; Gong et al., 2014), which picks labels for each label type based on random walks with restarts on a combined graph that includes movie-movie and movie-label edges. We use a restart probability of 0.15.

Figure 14 shows the precision@1 for different fractions of the network being labeled. All methods achieve higher accuracy for cinematographer and production designer, but lower accuracy for writer. This is because less than 11% of the edges have a shared-writer as the reason for that edge.

All EdgeExplain inference procedures (REL, VAR, and HYBRID) are better than the competing methods for all label types. All EdgeExplain methods are similar for predicting cinematographers and production designers. However, for writer prediction, VAR is better than REL by about 7% on average. This is because edges are formed primarily due to shared cinematographers and production-designers, and these are reasonably easy to infer (for instance, even LP works well for these). Thus, the fraction of "near-certain friends" (i.e., friendships for which the underlying reason is accurately estimated; see Algorithm 1) is high enough that VAR becomes more accurate than REL at predicting writers, as shown earlier in our simulation studies (Figure 5(b)). Note that for the Facebook data set, REL was much better than VAR. This shows the usefulness of HYBRID in tracking the best-performing variant of EdgeExplain.

The differences between the EdgeExplain variants are much less than that between EdgeExplain and the second-best method, which is LP. The greatest difference is again for writer prediction, where HYBRID is always $20-25\%$ better than LP. This mirrors our observations on the Facebook network where the greatest difference between EdgeExplain and LP was in predicting a person's employer; again, friendships formed due to a shared employer are relatively rare.

We again note the fact that LP is the best among the baseline methods. This agrees with previous work by Macskassy and Provost (2007) which showed the high prediction accuracy of LP. Observe that CN-SAN performs about as well as LP; this is expected, since CN-SAN is just LP with a hard label assignment in each iteration. Finally, we note that

LINK performed poorly and is omitted from the plots. We believe this is because both the number of nodes and the number of labels are too large. LINK might be a better option for predicting label types with fewer possible labels such as gender and political affiliation, as in (Zheleva and Getoor, 2009).

9. Generalizations

We now discuss some aspects and generalizations of EdgeExplain that demonstrate its wide applicability.

Related label types. Property **(P2)** assumes that the reasons for friendship formation are mutually exclusive, but this need not be strictly true. For example, some high school friends could be a subset of hometown friends.[4] Let us again consider Figure 1, but with current city replaced by high school. Suppose that the solid-black nodes represent actual high school friends, and we are trying to infer u's high school. If the small cluster on the right did not exist, then Eq. 2 would be maximized by picking the most common high school among u's friends (i.e., the solid-black nodes), even if they are already explained by a shared hometown; thus, EdgeExplain would pick the correct high school. On the other hand, if some friendships would remain unexplained without a shared high school (e.g., the small cluster in Figure 1), then it is not obvious whether we should prefer a high school that explains these edges or a high school that represents a large segment of hometown friends. The parameter α modulates this trade-off, with a higher value of α emphasizing the explanation of all edges as against the explanation of several edges a little better. The choice of α must depend on the characteristics of the social network; for the Facebook network, the best empirical results are achieved for large α (shown in Section 8.1), suggesting that many of our label types are indeed mutually exclusive.

Incorporating edge features. There are several situations where edge-specific features could be useful. First, we may want to give more importance to certain kinds of edges, such as the group-membership edges mentioned above. Second, some features could be important for one label type but not another: e.g., the age difference between friends could be useful for inferring high school but not employer. All these situations can be easily handled by modifying Eq. 3 to include an edge-specific and label type-specific weight. The corresponding modifications to the inference method are trivial.

Nodes with multiple labels. Our original formulation of EdgeExplain (Eqs. 2–4) assumes that each node has a single label for each label type, i.e., for any user u and label type t, only one of the indicator variables $S_{ut\ell}$ is 1. The accuracy of EdgeExplain on real-world data sets shows that this is at least a reasonable approximation. However, if handling multiple labels is critical, two generalizations of EdgeExplain are possible.

First, we can allow $S_{ut\ell}$ to be 1 for multiple labels ℓ of label type t. This leads to a modified relaxation labeling formulation, where the inferred label scores $f_{ut\ell}$ need not sum

4. The relationship between high school and hometown is in fact more complicated. The high school could be within driving distance of the hometown, but not in it; and sometimes even this does not hold.

to 1. Specifically, we choose the label scores as follows:

$$\begin{aligned}
&\underset{\boldsymbol{f}}{\text{Maximize}} \sum_{u\sim v} \log\Big(\underset{t\in\mathcal{T}}{\text{softmax}}(r(u,v,t))\Big) - \gamma \cdot \sum_u \sum_{t\in\mathcal{T}} \left(\sum_\ell f_{ut\ell} - 1\right)_+ \\
&\text{where } r(u,v,t) = \sum_{\ell\in L(t)} f_{ut\ell} f_{vt\ell} \\
&\qquad f_{ut\ell} \geq 0 \\
&\qquad f_{ut\ell} \leq 1
\end{aligned}$$

Thus, the inferred label scores $f_{ut\ell}$ need to sum to 1, but a hinge loss penalty $(\sum_\ell f_{ut\ell} - 1)_+$ is applied whenever it is greater than 1 (x_+ denotes $\max(x, 0)$). Increasing the penalty parameter γ enforces the constraint more strictly, and as $\gamma \to \infty$, we regain our original relaxation labeling formulation (Eqs. 5-8).

The second approach is to modify EdgeExplain, by again allowing $S_{ut\ell}$ to be 1 for multiple labels ℓ, but using a normalization when computing the degree to which a label type "explains" an edge. More precisely, we replace $r(u,v,t) = \sum_\ell S_{ut\ell} S_{vt\ell}$ (Eq. 3) by a probabilistic version:

$$P\left(r(u,v,t) > 0 \mid S_{ut.}, S_{vt.}\right) = \sum_\ell \left(\frac{S_{ut\ell}}{\sum_{\ell'} S_{ut\ell'}} \cdot \frac{S_{vt\ell}}{\sum_{\ell'} S_{vt\ell'}}\right) \tag{12}$$

In addition, we shall let the indicator random variables $S_{ut\ell}$ be drawn independently from Bernoulli distributions: $P(S_{ut\ell} = 1 \mid \mathcal{S}^-) = f_{ut\ell}$, where $\mathcal{S}^-$ represents all indicators except $S_{ut\ell}$. Then, by marginalizing our $S_{ut.}$ and $S_{vt.}$ in Eq. 12, we can express the marginal probability $P(r(u,v,t) > 0)$ in terms of the parameters $\boldsymbol{f}$. In effect, we solve the following optimization problem:

$$\begin{aligned}
&\underset{\boldsymbol{f}}{\text{Maximize}} \sum_{u\sim v} \log\Big(\underset{t\in\mathcal{T}}{\text{softmax}}\left(P(r(u,v,t) > 0)\right)\Big) \\
&\text{where } P\left(r(u,v,t) > 0\right) = \sum_{\ell\in L(t)} E_{S_{ut.}|f_{ut.}}\left[\frac{S_{ut\ell}}{\sum_{\ell'} S_{ut\ell'}}\right] \cdot E_{S_{vt.}|f_{vt.}}\left[\frac{S_{vt\ell}}{\sum_{\ell'} S_{vt\ell'}}\right] \\
&\qquad f_{ut\ell} \geq 0 \\
&\qquad f_{ut\ell} \leq 1
\end{aligned}$$

The first approach has the advantage of being a simple extension to our existing relaxation labeling approach, which already works well. However, it might require careful tuning for the extra parameter γ. The second approach directly incorporates the desired quantities $\boldsymbol{f}$ as model parameters, so it does not require an extra relaxation step. However, it must make a strong independence assumption. We leave a detailed comparison of these approaches for future work.

10. Conclusions

We proposed the problem of jointly inferring multiple correlated label types in a large network and described the problems with existing single-label models. We noted that one particular failure mode of existing methods in our problem setting is that edges are often created for a reason associated with a particular label type (e.g., in a social network, two users may link because they went to the same high school, but they did not go to the same college). We identified three network properties that model this phenomenon: edges are created for a reason **(P1)**, they are generally created only for one reason **(P2)**, and sharing the same value for a label type is necessary but not sufficient for having an edge between two nodes **(P3)**.

We introduced EdgeExplain, which carefully models these properties. We presented two inference methods for EdgeExplain: a relaxation-labeling method and a variational approach, both of which lead to fast iterative inference that is equivalent in running time to basic label propagation. Our empirical evaluation on a large subset of the Facebook graph amply demonstrates the benefits of EdgeExplain, with significant improvements across a set of different label types. Our further analysis validates many of the properties and intuitions we had about modeling networks, primarily that one can achieve significant improvements if one considers and models the *reason* an edge exists. Whether one is interested in inferring one or multiple label types, modeling these explanations will have significant impact on the accuracy of the final predictions.

Appendix A. Proofs

Proof [Theorem 1] Note that $\sum_{t\ell} S_{ut\ell} S_{vt\ell}$ is the number of matching label types between u and v, and hence is an integer between 0 and $|\mathcal{T}|$. Thus,

$$\left\langle \ln\left(1 + e^{-\left(\alpha \sum_{t\ell} S_{ut\ell} S_{vt\ell} + c\right)}\right)\right\rangle = \sum_{k=0}^{|\mathcal{T}|} \ln\left(1 + e^{-(\alpha k + c)}\right) \left\langle I\left\{\sum_{t\ell} S_{ut\ell} S_{vt\ell} = k\right\}\right\rangle.$$

Now, $\sum_{t\ell} S_{ut\ell} S_{vt\ell}$ equals k iff exactly k label types out of $|\mathcal{T}|$ are shared, i.e., there exists a binary vector $\mathbf{w} \in \{0,1\}^{|\mathcal{T}|}$ of length $|\mathcal{T}|$ such that exactly k bits are "one" ($|\mathbf{w}| = k$), and

$$\sum_\ell S_{ut\ell} S_{vt\ell} = \begin{cases} 1 & \text{if } w_t = 1 \\ 0 & \text{otherwise.} \end{cases}$$

Thus,

$$I\left\{\sum_{t\ell} S_{ut\ell} S_{vt\ell} = k\right\} = \sum_{\{\mathbf{w}\,:\,|\mathbf{w}|=k\}} \prod_t M_t^{w_t} (1 - M_t)^{1 - w_t},$$

where $M_t = I\left\{\sum_\ell S_{ut\ell} S_{vt\ell} = 1\right\}$ indicates a shared label type t (the subscripts u and v are dropped for clarity). Thus,

$$\left\langle I\left\{\sum_{t\ell} S_{ut\ell} S_{vt\ell} = k\right\}\right\rangle_Q = \sum_{\{\mathbf{w}\,:\,|\mathbf{w}|=k\}} \prod_t \left\langle M_t^{w_t} (1 - M_t)^{1 - w_t}\right\rangle_Q,$$

using the linearity of expectation and the independence of label types in the factorized distribution Q. Now, the expectation $\langle M_t \rangle_Q$ is just the probability of sharing a label of type t, so $\langle M_t \rangle_Q = \sum_\ell \mu_{ut\ell} \mu_{vt\ell} = \eta_{uvt}$. Then,

$$\kappa(w_t, \eta_{uvt}) \triangleq \left\langle M_t^{w_t} (1 - M_t)^{1 - w_t}\right\rangle_Q = \eta_{uvt}^{w_t} (1 - \eta_{uvt})^{1 - w_t}.$$

We want to find the optimal $\boldsymbol{\mu}$ that maximizes the lower bound of Eq. 10, under the constraint that $\sum_\ell \mu_{ut\ell} = 1$ for all label types t and users u. Combining all the above equations, we want to

$$\begin{aligned}\underset{\boldsymbol{\mu}}{\text{maximize}} \; & -\sum_{ut\ell} \mu_{ut\ell} \ln \mu_{ut\ell} \\ & -\sum_{u\sim v} \sum_{\mathbf{w}} \ln\left(1+e^{-(\alpha|\mathbf{w}|+c)}\right) \prod_t \kappa(w_t, \eta_{uvt}) \\ & -\sum_{ut} \lambda_{ut} \left(\sum_\ell \mu_{ut\ell} - 1\right),\end{aligned}$$

where λ_{ut} are the Lagrange multipliers. Setting the derivatives with respect to $\boldsymbol{\mu}_u$ to zero and holding $\boldsymbol{\mu}_v$ fixed for $v \neq u$, we find:

$$\begin{aligned}\mu^*_{ut\ell} &\propto \exp\left\{\sum_{\{v|u\sim v\}} \mu_{vt\ell} \sum_{\mathbf{w}} \phi(\mathbf{w})\right\} \\ \phi(\mathbf{w}) &= \ln\left(1+e^{-(\alpha|\mathbf{w}|+c)}\right)(-1)^{w_t} \prod_{t'\neq t} \kappa(w_{t'}, \eta_{uvt'}).\end{aligned}$$

Normalizing this expression over all $\ell' \in L(t)$ gives the exact value of $\mu_{ut\ell}$. In practice, only the top few labels for label type t need to be considered. ■

Proof [Proposition 1] Let $\mathbf{w}^{(-t)} \in \{0,1\}^{|\mathcal{T}|-1}$ represent the binary vector $\mathbf{w}$ restricted to all but the t^{th} component. Denoting $\prod_{t'\neq t} \kappa(w_{t'}, \eta_{uvt'})$ by $\chi(\mathbf{w}^{(-t)})$,

$$\begin{aligned}\sum_{\mathbf{w}} \phi_{uv}(\mathbf{w}) &= \sum_{\mathbf{w}} \chi(\mathbf{w}^{(-t)})(-1)^{w_t} \ln\left(1+e^{-(\alpha|\mathbf{w}|+c)}\right) \\ &= \sum_{\mathbf{w}^{(-t)}} \chi(\mathbf{w}^{(-t)}) \ln\left(\frac{1+e^{-\alpha|\mathbf{w}^{(-t)}|-c}}{1+e^{-\alpha(|\mathbf{w}^{(-t)}|+1)-c}}\right) \\ &\geq 0,\end{aligned}$$

where the first equality follows by grouping the two $\mathbf{w}$ terms with the same $\mathbf{w}^{(-t)}$ and summing the contributions for $w_t = 0$ and 1. The inequality holds because the numerator inside the logarithm is greater than the denominator, and $\chi \geq 0$. ■

Proof [Theorem 2] Let $\mathcal{U}$ represent the event that the ego has already selected her labels Y_u, and has chosen label ℓ for label type t. Define the random variable X_i as $+1$ if the friend v_i has label ℓ' for type t, -1 if it is ℓ, and 0 otherwise: $X_i = I_{Y_i(t)=\ell'} - I_{Y_i(t)=\ell}$. Then, LP fails via (t,ℓ,ℓ') if $\sum_i X_i > 0$. We bound the probability of the latter quantity by noting that it is the sum of independent random variables X_i, so a Hoeffding bound applies:

$$P\left(\sum X_i \geq E\left[\sum X_i\right] - Nx \,\middle|\, \mathcal{U}\right) \geq 1 - \exp\{-0.5Nx^2\} \quad (\text{for } x \geq 0). \tag{13}$$

Then, $E[X_i \mid \mathcal{U}]$ is easily seen to be:

$$\frac{E\left[\sum X_i \mid \mathcal{U}\right]}{N} = E[X_i \mid \mathcal{U}] = \sum_{t'\neq t} q_{t'}\left(p_{t,t'}(\ell') - p_{t,t'}(\ell)\right) - q_t = \Delta_{t,\ell,\ell'}.$$

Thus, $\Delta_{t,\ell,\ell'}$ is the expected difference in the rates of occurrences of ℓ' and ℓ; note that it depends on both how frequent the labels ℓ and ℓ' are, and the ego's preferences for choosing friends sharing each label type. Setting $x = E\left[\sum X_i \mid \mathcal{U}\right]/N - \epsilon = \Delta_{t,\ell,\ell'} - \epsilon$ for some small $\epsilon > 0$ in Eq. 13, we find:

$$P\left(\sum X_i > 0 \,\middle|\, \mathcal{U}\right) \geq 1 - \exp\{-0.5N(\Delta_{t,\ell,\ell'} - \epsilon)^2\}.$$

Hence,

$$\begin{aligned}P(\text{LP fails via } (t,\ell,\ell')) &\geq P\left(\sum X_i > 0\right) \\ &= \sum_{\{Y_u|Y_u(t)=\ell\}} \pi(Y_u) \cdot P\left(\sum X_i > 0 \,\middle|\, Y_u \text{ where } Y_u(t) = \ell\right) \\ &= \sum_{\{Y_u|Y_u(t)=\ell\}} \pi(Y_u) \cdot P\left(\sum X_i > 0 \,\middle|\, \mathcal{U}\right) \\ &\geq \sum_{\{Y_u|Y_u(t)=\ell\}} \pi(Y_u) \cdot \left(1 - \exp\{-0.5N(\Delta_{t,\ell,\ell'} - \epsilon)^2\}\right). \end{aligned} \tag{14}$$

■

Proof [Theorem 3] Restating Eq. 14, we find:

$$\begin{aligned}P(\text{LP fails via } (t,\ell,\ell')) &\geq \sum_{\{Y_u|Y_u(t)=\ell\}} \pi(Y_u) \cdot \left(1 - \exp\{-0.5N(\Delta_{t,\ell,\ell'} - \epsilon)^2\}\right) \\ &= \sum \pi(Y_u(t')) \cdot \pi(Y_u(t) = \ell \mid Y_u(t')) \cdot \left(1 - \exp\{-0.5N(\Delta_{t,\ell,\ell'} - \epsilon)^2\}\right) \\ &= \sum \pi(Y_u(t')) \cdot p_{t,t'}(\ell) \cdot \left(1 - \exp\{-0.5N(\Delta_{t,\ell,\ell'} - \epsilon)^2\}\right).\end{aligned}$$

To optimize the lower bound, we set the derivative of the RHS with respect to $p_{t,t'}(\ell)$ to 0. Noting that under TwoLabels, $p_{t,t'}(\ell') = 1 - p_{t,t'}(\ell)$, $q_{t'} = 1 - q_t$, and $\Delta_{t,\ell,\ell'} = q_{t'}(1 - 2p_{t,t'}(\ell)) - q_t = (1-q_t)(1-2p_{t,t'}(\ell)) - q_t$, we find:

$$\begin{aligned}0 &= \left(1 - \exp\{-0.5N(\Delta_{t,\ell,\ell'} - \epsilon)^2\}\right) \\ &\quad - 2Np_{t,t'}(\ell) \cdot \exp\{-0.5N(\Delta_{t,\ell,\ell'} - \epsilon)^2\} \cdot (\Delta_{t,\ell,\ell'} - \epsilon) \cdot (1-q_t) \\ &= 1 - \exp\{-0.5N(\Delta_{t,\ell,\ell'} - \epsilon)^2\} \cdot \left(1 + 2Np_{t,t'}(\ell)(1-q_t)\left((1-q_t)(1-2p_{t,t'}(\ell)) - q_t - \epsilon\right)\right) \\ &= 1 - \exp\{-0.5N(\Delta_{t,\ell,\ell'} - \epsilon)^2\} \times \\ &\qquad \left(1 + \underbrace{2(1-q_t)(1-2q_t-\epsilon)}_{c_1} Np_{t,t'}(\ell) - \underbrace{4(1-q_t)^2}_{c_2} Np_{t,t'}(\ell)^2\right) \\ &= 1 - \exp\{-0.5N \cdot \underbrace{\left((1-2q_t-\epsilon) - 2(1-q_t)p_{t,t'}(\ell)\right)^2}_{c_3}\} \cdot \left(1 + c_1 Np_{t,t'}(\ell) - c_2 Np_{t,t'}(\ell)^2\right).\end{aligned} \tag{15}$$

For large N, this can be satisfied only if $c_3 \approx 0$. Hence, let us express $p_{t,t'}(\ell)$ as $p_{t,t'}(\ell) = (1 - 2q_t - \epsilon)/(2(1 - q_t)) - p' \triangleq \kappa - p'$ for some small p', where $p' > 0$ to ensure that $\Delta_{t,\ell,\ell'} - \epsilon > 0$. Using this in Eq. 15, we find:

$$\begin{aligned}\exp\{2N(1-q_t)^2 p'^2\} &= 1 + c_1 N(\kappa - p') - c_2 N(\kappa - p')^2 \\ &= 1 + N\kappa(c_1 - c_2\kappa) + Np'(2c_2\kappa - c_1) - Np'^2 c_2 \\ &\leq 1 + N\kappa(c_1 - c_2\kappa) + N|(2c_2\kappa - c_1)| \\ \Rightarrow p' &= O\left(\sqrt{\frac{\log N}{N}}\right).\end{aligned}$$

Hence, $\Delta_{t,\ell,\ell'} = (1 - 2q_t) - 2(1 - q_t)p_{t,t'}(\ell) = (1 - 2q_t) - 2(1 - q_t)(\kappa - p') = O\left(\sqrt{\frac{\log N}{N}}\right)$. The requirement that $q_t < 0.5$ is necessitated by $p_{t,t'}(\ell) > 0$. ■

Proof [Corollary 4] When N is fixed, Theorem 3 suggests requiring

$$\begin{aligned}\Delta_{t,\ell,\ell'} &\approx \text{const.} \\ \Rightarrow (1 - q_t)(1 - 2p_{t,t'}(\ell)) - q_t &\approx \text{const.} \\ \Rightarrow 1 - 2\left(p_{t,t'}(\ell) + q_t - p_{t,t'}(\ell) \cdot q_t\right) &\approx \text{const.}\end{aligned}$$

which proves the claim. ■

References

Edoardo M. Airoldi, David M. Blei, Stephen E. Fienberg, and Eric P. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 9(Sep):1981–2014, 2008.

Shumeet Baluja, Rohan Seth, D Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. Video suggestion and discovery for YouTube: Taking random walks through the view graph. In *Proceedings of the 17th International Conference on World Wide Web*, pages 895–904, 2008.

Amir Beck and Marc Teboulle. Gradient-based algorithms with applications to signal-recovery problems. *Convex Optimization in Signal Processing and Communications*, (2): 42–88, 2009.

Mikhail Belkin, Irina Matveeva, and Partha Niyogi. Regularization and semi-supervised learning on large graphs. In *Proceedings of the 17th Annual Conference on Learning Theory*, pages 624–638, 2004.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7(Nov):2399–2434, 2006.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3(Mar):993–1022, 2003.

David M. Blei, Thomas L. Griffiths, and Michael I. Jordan. The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies. *Journal of the ACM*, 57(2), January 2010.

Deepayan Chakrabarti, Stanislav Funiak, Jonathan Chang, and Sofus A. Macskassy. Joint inference of multiple label types in large networks. In *Proceedings of the 31st International Conference on Machine Learning*, pages 874–882, 2014.

Soumen Chakrabarti, Byron Dom, and Piotr Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, pages 307–318, 1998.

Jonathan Chang and David M. Blei. Hierarchical relational models for document networks. *The Annals of Applied Statistics*, 4(1):124–150, 2010.

Avery Ching. Scaling Apache Giraph to a trillion edges. Facebook Engineering blog, 2013.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Computer Vision and Pattern Recognition*, pages 248–255. IEEE, 2009.

Yuxiao Dong, Yang Yang, Jie Tang, Yang Yang, and Nitesh V. Chawla. Inferring user demographics and social strategies in mobile social networks. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 15–24, 2014.

John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the ℓ_1-ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning*, pages 272–279, 2008.

Nir Friedman, Lise Getoor, Daphne Koller, and Avi Pfeffer. Learning probabilistic relational models. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 1300–1309, 1999.

Giraph. The Apache Giraph project. `http://giraph.apache.org`.

Neil Zhenqiang Gong, Wenchang Xu, Ling Huang, Prateek Mittal, Emil Stefanov, Vyas Sekar, and Dawn Song. Evolution of social-attribute networks: Measurements, modeling, and implications using Google+. In *Proceedings of the 2012 ACM Internet Measurement Conference*, pages 131–144, 2012.

Neil Zhenqiang Gong, Ameet Talwalkar, Lester Mackey, Ling Huang, Eui Chul Richard Shin, Emil Stefanov, Elaine (Runting) Shi, and Dawn Song. Joint link prediction and attribute inference using a social-attribute network. *ACM Transactions on Intelligent Systems and Technology*, 5(2):27:1–27:20, April 2014.

Thomas L. Griffiths and Zoubin Ghahramani. Infinite latent feature models and the Indian buffet process. In *Advances in Neural Information Processing Systems 18*, pages 475–482, 2005.

Qirong Ho, Jacob Eisenstein, and Eric P. Xing. Document hierarchies from text and links. In *Proceedings of the 21st International Conference on World Wide Web*, pages 739–749, 2012.

Peter D. Hoff, Adrian E. Raftery, and Mark S. Handcock. Latent space approaches to social network analysis. *Journal of the American Statistical Association*, 97(460):1090–1098, 2002.

Robert A. Hummel and Steven W. Zucker. On the foundations of relaxation labeling processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(3):267–287, 1983.

Charles Kemp, Joshua B. Tenenbaum, Thomas L. Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 381–388, 2006.

Daphne Koller and Avi Pfeffer. Probabilistic frame-based systems. In *Proceedings of the 15th National Conference on Artificial Intelligence*, pages 580–587, 1998.

Anastasios Kyrillidis, Stephen Becker, Volkan Cevher, and Christoph Koch. Sparse projections onto the simplex. In *Proceedings of the 30th International Conference on Machine Learning*, pages 235–243, 2013.

Jure Leskovec, Kevin J. Lang, and Michael W. Mahoney. Empirical comparison of algorithms for network community detection. In *Proceedings of the 19th International Conference on World Wide Web*, pages 631–640, 2010.

Qing Lu and Lise Getoor. Link-based classification. In *Proceedings of the 20th International Conference on Machine Learning*, pages 496–503, 2003.

Sofus A. Macskassy. Improving learning in networked data by combining explicit and mined links. In *Proceedings of the 22nd National Conference on Artificial Intelligence*, pages 590–595, 2007.

Sofus A. Macskassy and Foster Provost. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research*, 8(May):935–983, 2007.

Sofus Attila Macskassy and Foster Provost. A simple relational classifier. In *Proceedings of the Multi-Relational Data Mining Workshop at the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 64–76, 2003.

Grzegorz Malewicz, Matthew H. Austern, Aart J. C. Bik, James C. Dehnert, Ilan Horn, Naty Leiser, and Grzegorz Czajkowski. Pregel: A system for large-scale graph processing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, pages 135–146, 2010.

Miller McPherson, Lynn Smith-Lovin, and James M. Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444, 2001.

Kurt T. Miller, Thomas L. Griffiths, and Michael I. Jordan. Nonparametric latent feature models for link prediction. In *Advances in Neural Information Processing Systems 22*, pages 1276–1284, 2009.

Ramesh Nallapati, Amr Ahmed, Eric P. Xing, and William Cohen. Joint latent topic models for text and citations. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 542–550, 2008.

Jennifer Neville and David Jensen. Relational dependency networks. *Journal of Machine Learning Research*, 8(Mar):653–692, 2007.

Krzysztof Nowicki and Tom A. B. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001.

Konstantina Palla, David A. Knowles, and Zoubin Ghahramani. An infinite latent attribute model for network data. In *Proceedings of the 29th International Conference on Machine Learning*, pages 1607–1614, 2012.

Jim Pitman. Combinatorial Stochastic Processes. Springer-Verlag, Berlin/Heidelberg, 2006.

Azriel Rosenfeld, Robert A. Hummel, and Steven W. Zucker. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man, and Cybernetics*, 6(6):420–433, 1976.

Qinfeng Shi, James Petterson, Gideon Dror, John Langford, Alex Smola, and S. V. N. Vishwanathan. Hash kernels for structured data. *Journal of Machine Learning Research*, 10(Nov):2615–2637, 2009.

Partha Pratim Talukdar and William Cohen. Scaling graph-based semi supervised learning to large number of labels using count-min sketch. In *Proceedings of the 17th International Conference on Artificial Intelligence and Statistics*, pages 940–947, 2014.

Partha Pratim Talukdar and Koby Crammer. New regularized algorithms for transductive learning. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 442–457, 2009.

Benjamin Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In *Proceedings of the 18th Conference Conference on Uncertainty in Artificial Intelligence*, pages 485–492, 2002.

Leslie G. Valiant. A bridging model for parallel computation. *Communications of the ACM*, 33(8):103–111, 1990.

Jierui Xie, Stephen Kelley, and Boleslaw K. Szymanski. Overlapping community detection in networks: The state of the art and comparative study. *ACM Computing Surveys*, 45 (4), 2013.

Zhao Xu, Volker Tresp, Kai Yu, and Hans-Peter Kriegel. Learning infinite hidden relational models. In *ICML Workshop on Open Problems in Statistical Relational Learning*, 2006.

Zhijun Yin, Manish Gupta, Tim Weninger, and Jiawei Han. A unified framework for link recommendation using random walks. In *Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining*, pages 152–159, 2010. ISBN 978-0-7695-4138-9.

Elena Zheleva and Lise Getoor. To join or not to join: The illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th International Conference on World Wide Web*, pages 531–540, 2009.

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in Neural Information Processing Systems 16*, pages 321–328, 2003.

Xiaojin Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin–Madison, July 2008.

Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical report, Carnegie Mellon University, 2002.

Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, pages 912–919, 2003.

Journal of Machine Learning Research 18 (2017) 1-45 Submitted 5/16; Revised 2/17; Published 6/17

Achieving Optimal Misclassification Proportion in Stochastic Block Models

Chao Gao CHAOGAO@GALTON.UCHICAGO.EDU
University of Chicago

Zongming Ma ZONGMING@WHARTON.UPENN.EDU
University of Pennsylvania

Anderson Y. Zhang YE.ZHANG@YALE.EDU
Harrison H. Zhou HUIBIN.ZHOU@YALE.EDU
Yale University

Editor: Sara van de Geer

Abstract

Community detection is a fundamental statistical problem in network data analysis. In this paper, we present a polynomial time two-stage method that provably achieves optimal statistical performance in misclassification proportion for stochastic block model under weak regularity conditions. Our two-stage procedure consists of a refinement stage motivated by penalized local maximum likelihood estimation. This stage can take a wide range of weakly consistent community detection procedures as its initializer, to which it applies and outputs a community assignment that achieves optimal misclassification proportion with high probability. The theoretical property is confirmed by simulated examples.

Keywords: Clustering, Community detection, Minimax rates, Network analysis, Spectral clustering.

1. Introduction

Network data analysis (Wasserman, 1994; Goldenberg et al., 2010) has become an important topic in statistics. In fields such as physics, computer science, social science and biology, one observes a network among a large number of subjects of interest such as particles, computers, people, etc. The observed network can be modeled as an instance of a random graph and the goal is to infer structures of the underlying generating process. A structure of particular interest is *community*: there is a partition of the graph nodes in some suitable sense so that each node belongs to a community. Starting with the proposal of a series of methodologies (Girvan and Newman, 2002; Newman and Leicht, 2007; Handcock et al., 2007; Karrer and Newman, 2011), we have seen a large literature devoted to algorithmic solutions to uncovering community structure. Great advances have also been made in recent years on the theoretical understanding of the problem in terms of statistical consistency and thresholds for detection and exact recoveries. See, for instance, Bickel and Chen (2009); Decelle et al. (2011); Zhao et al. (2012); Mossel et al. (2012, 2013b); Massoulié (2014); Abbe et al. (2014); Mossel et al. (2014); Hajek et al. (2014), among others. The major goal of the present paper is to propose a computationally feasible algorithm for community detection in stochastic block models with adaptive minimax optimal performance.

©2017 Chao Gao, Zongming Ma, Anderson Y. Zhang and Harrison H. Zhou.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-245.html.

To describe network data with community structure, we focus on the stochastic block model (SBM) proposed by Holland et al. (1983). Let $A \in \{0,1\}^{n\times n}$ be the symmetric adjacency matrix of an undirected random graph generated according to an SBM with k communities. The diagonal entries of A are all zeros and each $A_{uv} = A_{vu}$ for $u > v$ is an independent Bernoulli random variable with mean $P_{uv} = B_{\sigma(u)\sigma(v)}$ for some symmetric connectivity matrix $B \in [0,1]^{k\times k}$ and some label function $\sigma : [n] \to [k]$. Here and after, for any positive integer m, $[m] = \{1,\dots,m\}$. In other word, if the u^{th} node and the v^{th} node belong to the i^{th} and the j^{th} community respectively, then $\sigma(u) = i$, $\sigma(v) = j$ and there is an edge connecting u and v with probability B_{ij}. We define a and b through $\min_i B_{ii} = a/n$ and $\max_{i\neq j} B_{ij} = b/n$. Community detection then refers to the problem of estimating the label function σ subject to a permutation of the community labels $\{1,\dots,k\}$. A natural loss function for such an estimation problem is the proportion of wrong labels (subject to a permutation of the label set $[k]$), which we refer to as misclassification proportion from here on.

Literature review The field of community detection in SBMs has been growing fast. Results on fundamental limits and various algorithms for achieving them have been obtained in the literature. We first review the most relevant results prior to our work.

1. *Detection.* In ground breaking works by Mossel et al. (2012, 2013b) and Massoulié (2014), the authors established sharp threshold for the regimes in which it is possible and impossible to achieve a misclassification proportion strictly less than $\frac{1}{2}$ (so that it is better than random guess) when $k = 2$ and both communities are of the same size. This solved the conjecture in Decelle et al. (2011) that was only justified in physics rigor. The necessary and sufficient condition for doing better than random guess is $(a-b)^2 > 2(a+b)$.

2. *Weak consistency.* In the current context, weak consistency means recovering all but a vanishing proportion of the community labels. As was shown in Mossel et al. (2014), the necessary and sufficient condition for achieving weak consistency is $(\sqrt{a}-\sqrt{b})^2 \to \infty$ in the equal-sized two community setting. Conditions of weak consistency in general SBMs were obtained and discussed in Zhang and Zhou (2015); Yun and Proutiere (2014a); Abbe and Sandon (2015a).

3. *Strong consistency.* Abbe et al. (2014); Mossel et al. (2014) established the necessary and sufficient condition for ensuring zero misclassification proportion (usually referred to as "strong consistency") with high probability when $k = 2$ and community sizes are equal. The result was later generalized to general SBMs with possibly unequal community sizes by Abbe and Sandon (2015a) using the notion of Chernoff-Hellinger (CH) divergence.

4. *Minimax optimal rates.* The foregoing three categories of results mainly focused on sharp conditions of achieving detection, weak consistency and strong consistency, respectively. Arguably, what is of more interest to statisticians is the optimal rates of misclassification proportion. When $(\sqrt{a}-\sqrt{b})^2/(k\log k) \to \infty$, it was derived by Zhang and Zhou (2015) that the optimal rate of misclassification proportion takes the

form of

$$\exp\left(-(1+o(1))\frac{nI^*}{k}\right), \tag{1}$$

for a k-community SBM with equal community sizes, where I^* is the Rényi divergence of order $\frac{1}{2}$ between Bern $\left(\frac{a}{n}\right)$ and Bern $\left(\frac{b}{n}\right)$ (Rényi, 1961). See Theorem 1 below for a more general and precise statement of the result. A special case of this result for symmetric SBMs with $k = 2$ was obtained in Yun and Proutiere (2014a).

5. *Algorithms.* Various algorithms have been proposed in the literature to achieve detection, weak consistency and strong consistency. A popular approach is spectral clustering. Its application on network data dates back to Hagen and Kahng (1992); McSherry (2001). Its performance on SBMs has been investigated by Coja-Oghlan (2010); Rohe et al. (2011); Sussman et al. (2012); Fishkind et al. (2013); Qin and Rohe (2013); Joseph and Yu (2013); Lei and Rinaldo (2014); Vu (2014); Chin et al. (2015); Jin (2015); Le et al. (2015), among others. Various ways for refining spectral clustering have been proposed, such as those in Amini et al. (2013); Abbe et al. (2014); Mossel et al. (2014); Lei and Zhu (2014); Yun and Proutiere (2014a); Chin et al. (2015), which lead to strong consistency or convergence rates that are exponential in signal-to-noise ratio, while Mossel et al. (2013a) studied the problem of minimizing a non-vanishing misclassification proportion. However, in the regime of weak consistency, with the exception of Yun and Proutiere (2014a) for equal-sized two community case, these refinement methods cannot attain the optimal misclassification proportion.

   Another important line of research is devoted to the investigation of likelihood-based methods, which was initiated by Bickel and Chen (2009) and later extended to more general settings by Zhao et al. (2012); Choi et al. (2012). To tackle the intractability of optimizing the likelihood function, an EM algorithm using pseudo-likelihood was proposed by Amini et al. (2013). Another way to overcome the intractability of the maximum likelihood estimator (MLE) is by convex relaxation. Various semi-definite relaxations were studied by Cai and Li (2014); Chen and Xu (2014); Amini and Levina (2014), and the sharp threshold for strong consistency can indeed be achieved by semi-definite programming (Hajek et al., 2014, 2015).

Main contribution The current paper proposes a computationally feasible algorithm that provably achieves the optimal misclassification proportion established in Zhang and Zhou (2015) adaptively under weak regularity conditions. It covers the cases of both finite and diverging number of communities and both equal and unequal community sizes, and it achieves both weak and strong consistency in the respective regimes. In addition, the algorithm is guaranteed to compute in polynomial time even when the number of communities diverges with the number of nodes. Since the error bound of the algorithm matches the optimal misclassification proportion (1) in Zhang and Zhou (2015) under weak conditions, it achieves various existing detection boundaries in the literature. For instance, for any fixed number of communities, the procedure is weakly consistent under the necessary and sufficient condition of Mossel et al. (2012, 2013b), and strongly consistent under the necessary and sufficient condition of Abbe et al. (2014); Mossel et al. (2014). Moreover, it could match the optimal misclassification proportion in Zhang and Zhou (2015) even

when k diverges. It is remarkable that the same algorithm can lead to optimal performance in both strong and weak consistency regiems, while most papers in the literature require different algorithms in the two regimes.

The core of the algorithm is a refinement scheme for community detection motivated by penalized MLE, an idea that was previously explored in Amini et al. (2013); Abbe et al. (2014); Mossel et al. (2014); Lei and Zhu (2014); Yun and Proutiere (2014a); Chin et al. (2015). As long as there exists an initial estimator that satisfies a certain weak consistency condition, the refinement scheme is able to obtain an improved estimator that achieves the optimal misclassification proportion with high probability. The key to achieve this goal is to optimize the *local* penalized likelihood function for each node separately. This local optimization step is completely data-driven and has a closed form solution, and hence can be computed very efficiently. The additional penalty term is indispensable as it plays a key role in ensuring the optimal performance when the community sizes are unequal and when the within community and/or between community edge probabilities are unequal.

To obtain a qualified initial estimator, we show that both spectral clustering and its normalized variant could satisfy the desired condition needed for subsequent refinement, though the refinement scheme works for any other method satisfying a certain weak consistency condition. Note that spectral clustering can be considered as a *global* method, and hence our two-stage algorithm runs in a "*from global to local*" fashion. In essence, with high probability, the global stage pinpoints a local neighborhood in which we shall search for solution to each local penalized maximum likelihood problem, and the subsequent local stage finds the desired solution.

Notable results after initial posting of this manuscript After the initial posting of this manuscript on arXiv (arXiv:1505.03772), there have appeared a number of papers with notable and related results. Below, we highlight them with brief discussions:

1. *Detection.* The algorithm proposed in this paper achieves optimal misclassification proportion in the weak and the strong consistency regimes. In the detection regime, optimal misclassification proportion and its statistical-computational gap have been studied in Deshpande et al. (2015); Mossel and Xu (2015); Abbe and Sandon (2015b).

2. *Strong and weak consistency.* Two algorithms were proposed in Abbe and Sandon (2015c) for community detection in general SBMs. One for the strong consistency regime, and the other for weak consistency and detection. The one for strong consistency was shown to achieve the goal all the way to the CH divergence threshold, which is tighter than the strong consistency result in the present paper for asymmetric SBMs. However, the convergence rate for the other algorithm is sub-optimal in the weak consistency regime even for symmetric SBMs. See also the discussion below for the further improvement in Yun and Proutiere (2015).

3. *Weighted/labeled SBMs.* Jog and Loh (2015) extends the results in Zhang and Zhou (2015) to networks with weighted/labeled edges using Rényi divergence of order $\frac{1}{2}$. A more recent paper Yun and Proutiere (2015) studied optimal misclassification proportion for weighted/labeled SBMs. The proposed algorithm in Yun and Proutiere (2015) is able to achieve the CH limit (Abbe and Sandon, 2015a) in the strong consistency regime. It also attains optimal misclassification proportion in the weak consistency

regime with respect to a potentially smaller class than the ones used in our paper. However, their stronger results also require a relatively stronger set of conditions. For example, they require $k = O(1)$. Moreover, for equal-sized two community binary SBMs, with a and b defined at the beginning of this section, Yun and Proutiere (2015) requires $a \asymp b$ and $a - b \asymp b$. In comparison, we do not require the latter for any result in this manuscript. We can even drop the former if we are willing to accept any $1 - \epsilon$ relaxation of the tight constant of the exponent in our error rates. See, for instance, Theorem 12.

4. *Degree-corrected block models.* The paper Gao et al. (2016) studied degree-corrected block models by deriving the minimax rates for misclassification proportion and proposing an adaptive algorithm.

In summary, progress has been made along several different directions after initial posting of the present manuscript. However, none of the aforementioned results dominates those we are to present in the rest of the paper.

Organization and notation The rest of the paper is organized as follows. Section 2 formally sets up the community detection problem and presents the two-stage algorithm. The theoretical guarantees for the proposed method are given in Section 3, followed by numerical results demonstrating its competitive performance on simulated datasets in Section 4. A discussion on the results in the current paper and possible directions for future investigation is included in Section 5. Section 6 presents the proofs of main results with some technical details deferred to the appendix.

We close this section by introducing some notation. For a matrix $M = (M_{ij})$, we denote its Frobenius norm by $\|M\|_{\mathrm{F}} = \sqrt{\sum_{ij} M_{ij}^2}$ and its operator norm by $\|M\|_{\mathrm{op}} = \max_l \lambda_l(M)$, where $\lambda_l(M)$ is its l^{th} singular value. We use M_{i*} to denote its i^{th} row. The norm $\|\cdot\|$ is the usual Euclidean norm for vectors. For a set S, $|S|$ denotes its cardinality. The notation $\mathbb{P}$ and $\mathbb{E}$ are generic probability and expectation operators whose distribution is determined from the context. For two positive sequences $\{x_n\}$ and $\{y_n\}$, $x_n \asymp y_n$ means $x_n/C \le y_n \le C x_n$ for some constant $C \ge 1$ independent of n, while $x_n = o(y_n)$ means $x_n/y_n \to 0$ as $n \to \infty$. Throughout the paper, unless otherwise noticed, we use C, c and their variants to denote absolute constants, whose values may change from line to line.

2. Problem formulation and methodology

In this section, we give a precise formulation of the community detection problem and present a new method for it. The method consists of two stages: initialization and refinement. We shall first introduce the second stage, which is the main algorithm of the paper. It clusters the network data by performing a node-wise penalized neighbor voting based on some initial community assignment. Then, we will discuss several candidates for the initialization step including a new greedy algorithm for clustering the leading eigenvectors of the adjacency matrix or of the graph Laplacian that is tailored specifically for stochastic block models. Theoretical guarantees for the algorithms introduced in the current section will be presented in Section 3.

2.1 Community detection in stochastic block model

Recall that a stochastic block model is completely characterized by a symmetric connectivity matrix $B \in [0,1]^{k\times k}$ and a label vector $\sigma \in [k]^n$. One widely studied parameter space of SBM is

$$\Theta_0(n,k,a,b,\beta) = \Big\{ (B,\sigma) : \sigma : [n] \to [k], |\{u \in [n] : \sigma(u) = i\}| \in \left[\frac{n}{\beta k} - 1, \frac{\beta n}{k} + 1\right], \ \forall i \in [k], \\ B = (B_{ij}) \in [0,1]^{k\times k}, B_{ii} = \frac{a}{n} \text{ for all } i \text{ and } B_{ij} = \frac{b}{n} \text{ for all } i \neq j \Big\} \quad (2)$$

where $\beta \ge 1$ is an absolute constant. This parameter space $\Theta_0(n,k,a,b,\beta)$ contains all SBMs in which the within community connection probabilities are all equal to $\frac{a}{n}$ and the between community connection probabilities are all equal to $\frac{b}{n}$. In the special case of $\beta = 1$, all communities are of nearly equal sizes.

Assuming equal within and equal between connection probabilities can be restrictive. Thus, we also introduce the following larger parameter space

$$\Theta(n,k,a,b,\lambda,\beta;\alpha) = \Big\{ (B,\sigma) : \sigma : [n] \to [k], |\{u \in [n] : \sigma(u) = i\}| \in \left[\frac{n}{\beta k} - 1, \frac{\beta n}{k} + 1\right], \ \forall i \in [k], \\ B = B^T = (B_{ij}) \in [0,1]^{k\times k}, \frac{b}{\alpha n} \le \frac{1}{k(k-1)} \sum_{i\neq j} B_{ij} \le \max_{i\neq j} B_{ij} = \frac{b}{n}, \\ \frac{a}{n} = \min_i B_{ii} \le \max_i B_{ii} \le \frac{\alpha a}{n}, \\ \lambda_k(P) \ge \lambda \text{ with } P = (P_{uv}) = (B_{\sigma(u),\sigma(v)}) \Big\}. \quad (3)$$

Throughout the paper, we treat $\beta \ge 1$ and $\alpha \ge 1$ as absolute constants, while k, a, b and λ should be viewed as functions of the number of nodes n which can vary as n grows. Moreover, we assume $0 < \frac{b}{n} < \frac{a}{n} \le 1 - \epsilon$ throughout the paper for some numeric constant $\epsilon \in (0,1)$. Thus, the parameter space $\Theta(n,k,a,b,\lambda,\beta;\alpha)$ requires that the within community connection probabilities are bounded from below by $\frac{a}{n}$ and the connection probabilities between any two communities are bounded from above by $\frac{b}{n}$. In addition, it requires that the sizes of different communities are comparable. In order to guarantee that $\Theta(n,k,a,b,\lambda,\beta;\alpha)$ is a larger parameter space than $\Theta_0(n,k,a,b,\beta)$, we always require λ to be positive and sufficiently small such that

$$\Theta_0(n,k,a,b,\beta) \subset \Theta(n,k,a,b,\lambda,\beta;\alpha). \quad (4)$$

According to Proposition 24 in the appendix, a sufficient condition for (4) is $\lambda \le \frac{a-b}{2\beta k}$. We assume (4) throughout the rest of the paper.

The labels on the n nodes induce a community structure $[n] = \cup_{i=1}^k \mathcal{C}_i$, where $\mathcal{C}_i = \{u \in [n] : \sigma(u) = i\}$ is the i^{th} community with size $n_i = |\mathcal{C}_i|$. Our goal is to reconstruct this partition, or equivalently, to estimate the label of each node modulo any permutation of label symbols. Therefore, a natural error measure is the misclassification proportion defined as

$$\ell(\widehat{\sigma}, \sigma) = \min_{\pi \in S_k} \frac{1}{n} \sum_{u \in [n]} \mathbf{1}_{\{\widehat{\sigma}(u) \neq \pi(\sigma(u))\}}, \quad (5)$$

where S_k stands for the symmetric group on $[k]$ consisting of all permutations of $[k]$.

2.2 Main algorithm

We now present the main method of the paper – a refinement algorithm for community detection in stochastic block model motivated by penalized local maximum likelihood estimation.

To motivate our proposal, for any SBM in the parameter space $\Theta_0(n,k,a,b,1)$ with equal community size, the MLE for σ (Cai and Li, 2014; Chen and Xu, 2014; Zhang and Zhou, 2015) is

$$\widehat{\sigma} = \underset{\sigma:[n]\to[k]}{\operatorname{argmax}} \sum_{u<v} A_{uv}\mathbf{1}_{\{\sigma(u)=\sigma(v)\}}, \tag{6}$$

which is a combinatorial optimization problem and hence is computationally intractable. However, node-wise optimization of (6) has a simple closed form solution. Suppose the values of $\{\sigma(u)\}_{u=2}^n$ are known and we want to estimate $\sigma(1)$. Then, (6) reduces to

$$\widehat{\sigma}(1) = \underset{i\in[k]}{\operatorname{argmax}} \sum_{\{v\neq 1:\sigma(v)=i\}} A_{1v}. \tag{7}$$

For each $i\in[k]$, the quantity $\sum_{\{v\neq1:\sigma(v)=i\}} A_{1v}$ is the number of neighbors that the first node has in the i^{th} community. Therefore, the most likely label for the first node is the one it has the most connections with when all communities are of equal sizes. In practice, we do not know any label in advance. However, we may estimate the labels of all but the first node by first applying a community detection algorithm σ^0 on the subnetwork excluding the first node and its associated edges, the adjacency matrix of which is denoted by A_{-1} since it is the $(n-1)\times(n-1)$ submatrix of A with its first row and first column removed. Once we estimate the remaining labels, we can apply (7) to estimate $\sigma(1)$ but with $\{\sigma(v)\}_{v=2}^n$ replaced with the estimated labels.

For any $u\in[n]$, let A_{-u} denote the $(n-1)\times(n-1)$ submatrix of A with its u^{th} row and u^{th} column removed. Given any community detection algorithm σ^0 which is able to cluster any graph on $n-1$ nodes into k categories, we present the precise description of our refinement scheme in Algorithm 1.

The algorithm works in two consecutive steps. The first step carries out the foregoing heuristics on a node by node basis. For each fixed node u, we first leave the node out and apply the available community detection algorithm σ^0 on the remaining $n-1$ nodes and the edges among them (as summarized in the matrix $A_{-u}\in\{0,1\}^{(n-1)\times(n-1)}$) to obtain an initial community assignment vector σ_u^0. For convenience, we make σ_u^0 an n-vector by fixing $\sigma_u^0(u)=0$, though applying σ^0 on A_{-u} does not give any community assignment for u. We then assign the label of the u^{th} node according to (10), which is essentially (7) with σ replaced with σ_u^0 except for the additional penalty term. The additional penalty term is added to ensure the optimal performance even when both the diagonal and the off-diagonal entries of the connectivity matrix B are allowed to take different values and the community sizes are not necessarily equal. To determine the penalty parameter ρ_u in an adaptive way as spelled out in (11) – (12), we first estimate the connectivity matrix B based on A_{-u} in (8) – (9). After we obtain the community assignment for u, we organize the assignment for all n vertices into an n-vector $\widehat{\sigma}_u$. We call this step "penalized neighbor voting" since the first term on the RHS of (10) counts the number of neighbors of u in each (estimated) community while the second term is a penalty term proportional to the size of each (estimated) community.

Algorithm 1: A refinement scheme for community detection

Input: Adjacency matrix $A\in\{0,1\}^{n\times n}$,
number of communities k,
initial community detection method σ^0.

Output: Community assignment $\widehat{\sigma}$.

Penalized neighbor voting:

for $u=1$ **to** n **do**

Apply σ^0 on A_{-u} to obtain $\sigma_u^0(v)$ for all $v\neq u$ and let $\sigma_u^0(u)=0$;

Define $\widetilde{\mathcal{C}}_i^u=\{v:\sigma_u^0(v)=i\}$ for all $i\in[k]$; let $\widetilde{\mathcal{E}}_i^u$ be the set of edges within $\widetilde{\mathcal{C}}_i^u$, and $\widetilde{\mathcal{E}}_{ij}^u$ the set of edges between $\widetilde{\mathcal{C}}_i^u$ and $\widetilde{\mathcal{C}}_j^u$ when $i\neq j$;

Define

$$\widehat{B}_{ii}^u=\frac{|\widetilde{\mathcal{E}}_i^u|}{\frac{1}{2}|\widetilde{\mathcal{C}}_i^u|(|\widetilde{\mathcal{C}}_i^u|-1)},\quad \widehat{B}_{ij}^u=\frac{|\widetilde{\mathcal{E}}_{ij}^u|}{|\widetilde{\mathcal{C}}_i^u||\widetilde{\mathcal{C}}_j^u|},\quad \forall i\neq j\in[k], \tag{8}$$

and let

$$\widehat{a}_u=n\min_{i\in[k]}\widehat{B}_{ii}^u \quad\text{and}\quad \widehat{b}_u=n\max_{i\neq j\in[k]}\widehat{B}_{ij}^u. \tag{9}$$

Define $\widehat{\sigma}_u:[n]\to[k]$ by setting $\widehat{\sigma}_u(v)=\sigma_u^0(v)$ for all $v\neq u$ and

$$\widehat{\sigma}_u(u)=\underset{l\in[k]}{\operatorname{argmax}}\sum_{\sigma_u^0(v)=l}A_{uv}-\rho_u\sum_{v\in[n]}\mathbf{1}_{\{\sigma_u^0(v)=l\}} \tag{10}$$

where for

$$t_u=\frac{1}{2}\log\frac{\widehat{a}_u(1-\widehat{b}_u/n)}{\widehat{b}_u(1-\widehat{a}_u/n)}, \tag{11}$$

we define

$$\rho_u=-\frac{1}{2t_u}\log\left(\frac{\frac{\widehat{a}_u}{n}e^{-t_u}+1-\frac{\widehat{a}_u}{n}}{\frac{\widehat{b}_u}{n}e^{t_u}+1-\frac{\widehat{b}_u}{n}}\right), \tag{12}$$

end

Consensus:

Define $\widehat{\sigma}(1)=\widehat{\sigma}_1(1)$. For $u=2,\ldots,n$, define

$$\widehat{\sigma}(u)=\underset{l\in[k]}{\operatorname{argmax}}\left|\{v:\widehat{\sigma}_1(v)=l\}\cap\{v:\widehat{\sigma}_u(v)=\widehat{\sigma}_u(u)\}\right|. \tag{13}$$

Once we complete the above procedure for each of the n nodes, we obtain n vectors $\widehat{\sigma}_u \in [k]^n$, $u = 1, \ldots, n$, and turn to the second step of the algorithm. The basic idea behind the second step is to obtain a unified community assignment by assembling $\{\widehat{\sigma}_u(u) : u \in [n]\}$ and the immediate hurdle is that each $\widehat{\sigma}_u$ is only determined up to a permutation of the community labels. Thus, the second step aims to align these different permutations by (13) before we assemble the $\widehat{\sigma}_u(u)$'s. We call this step "consensus" since we are essentially looking for a consensus on the community labels for n possibly different community assignments, under the assumption that all of them are close to the ground truth up to some permutation.

2.3 Initialization via spectral methods

In this section, we present algorithms that can be used as initializers in Algorithm 1. Note that for any model in (3), the matrix P has rank at most k and $\mathbb{E}A_{uv} = P_{uv}$ for all $u \neq v$. We may first reduce the dimension of the data and then apply some clustering algorithm. Such an approach is usually referred to as spectral clustering (von Luxburg, 2007). The application of spectral clustering on network data goes back to Hagen and Kahng (1992); McSherry (2001), and its performance under the stochastic block model has been investigated by Coja-Oghlan (2010); Rohe et al. (2011); Sussman et al. (2012); Fishkind et al. (2013); Qin and Rohe (2013); Joseph and Yu (2013); Lei and Rinaldo (2014); Vu (2014); Chin et al. (2015); Jin (2015); Le et al. (2015), among others. Technically speaking, spectral clustering refers to the general method of clustering eigenvectors of some data matrix. For random graphs, two commonly used methods are unnormalized spectral clustering (USC) and normalized spectral clustering (NSC). The former refers to clustering the eigenvectors of the adjacency matrix A itself and the latter refers to clustering the eigenvectors of the associated graph Laplacian $L(A)$. To formally define the graph Laplacian, we introduce the notation $d_u = \sum_{v\in[n]} A_{uv}$ for the degree of the u^{th} node. The graph Laplacian operator $L : A \mapsto L(A)$ is defined by $L(A) = ([L(A)]_{uv})$ where $[L(A)]_{uv} = d_u^{-1/2} d_v^{-1/2} A_{uv}$. Although there have been debates and studies on which one works better (see, for example, von Luxburg et al. (2008); Sarkar and Bickel (2013)), for our purpose, both of them can lead to sufficiently decent initial estimators.

The performances of USC and NSC depend critically on the bounds $\|A - P\|_{\text{op}}$ and $\|L(A) - L(P)\|_{\text{op}}$, respectively. However, as pointed out by Chin et al. (2015); Le et al. (2015), the matrices A and $L(A)$ are not good estimators of P and $L(P)$ under the operator norm when the graph is sparse in the sense that $\max_{u,v\in[n]} P_{uv} = o(\log n/n)$. Thus, it is necessary to regularize A and $L(A)$ in order to achieve better performances for USC and NSC. The adjacency matrix A can be regularized by trimming those nodes with high degrees. Define the trimming operator $T_\tau : A \mapsto T_\tau(A)$ by replacing the u^{th} row and the u^{th} column of A with 0 whenever $d_u \geq \tau$, and so $T_\tau(A)$ and A are of the same dimensions. It is argued in Chin et al. (2015) that by removing those high-degree nodes, $T_\tau(A)$ has better convergence properties. Regularization method for graph Laplacian goes back to Amini et al. (2013) and its theoretical properties have been studied by Joseph and Yu (2013); Le et al. (2015). In particular, Amini et al. (2013) proposed to use $L(A_\tau)$ for NSC where $A_\tau = A + \frac{\tau}{n}\mathbf{1}\mathbf{1}^T$ and $\mathbf{1} = (1, 1, ..., 1)^T \in \mathbb{R}^n$. From now on, we use USC(τ) and NSC(τ) to denote unnormalized spectral clustering and normalized spectral clustering with

regularization parameter τ, respectively. Note that the unregularized USC is USC(∞) and the unregularized NSC is NSC(0).

Another important issue in spectral clustering lies in the subsequent clustering method used to cluster the eigenvectors. A popular choice is k-means clustering. However, finding the global solution to the k-means problem is NP-hard (Aloise et al., 2009; Mahajan et al., 2009). Kumar et al. (2004) proposed a polynomial time algorithm for achieving $(1 + \epsilon)$ approximation to the k-means problem for any fixed k, which was utilized in Lei and Rinaldo (2014) to establish consistency for spectral clustering under stochastic block models with a fixed number of communities. However, a closer look at the complexity bound suggests that the smallest possible ϵ is proportional to k. Thus, applying the algorithm and the associated bound in Kumar et al. (2004) directly in our settings can lead to inferior error bounds when $k \to \infty$ as $n \to \infty$. To address this issue for stochastic block models, we propose a greedy clustering method in Algorithm 2. The method is inspired by the fact that the clustering centers in stochastic block models are well separated from each other on the population level. It is straightforward to check that the complexity of Algorithm 2 is polynomial in n.

Algorithm 2: A greedy method for clustering

Input: Data matrix $\widehat{U} \in \mathbb{R}^{n\times k}$, either the leading eigenvectors of $T_\tau(A)$ or that of $L(A_\tau)$,
number of communities k,
critical radius $r = \mu\sqrt{\frac{k}{n}}$ with some constant $\mu > 0$.

Output: Community assignment $\widehat{\sigma}$.

Set $S = [n]$;
for $i = 1$ **to** k **do**
  Let $t_i = \arg\max_{u\in S} \left|\left\{v \in S : \left\|\widehat{U}_{v*} - \widehat{U}_{u*}\right\| < r\right\}\right|$;
  Set $\widehat{\mathcal{C}}_i = \left\{v \in S : \left\|\widehat{U}_{v*} - \widehat{U}_{t_i*}\right\| < r\right\}$;
  Label $\widehat{\sigma}(u) = i$ for all $u \in \widehat{\mathcal{C}}_i$;
  Update $S \leftarrow S\backslash\widehat{\mathcal{C}}_i$.
end
If $S \neq \varnothing$, then for any $u \in S$, set $\widehat{\sigma}(u) = \text{argmin}_{i\in[k]} \frac{1}{|\widehat{\mathcal{C}}_i|}\sum_{v\in\widehat{\mathcal{C}}_i} \left\|\widehat{U}_{u*} - \widehat{U}_{v*}\right\|$.

Last but not least, we would like to emphasize that one needs not limit the initialization algorithm to the spectral methods introduced in this section. As Theorem 4 below shows, Algorithm 1 works for any initialization method that satisfies a weak consistency condition.

3. Theoretical properties

Before stating the theoretical properties of the proposed method, we first review the minimax rate in Zhang and Zhou (2015), which serves as the optimality benchmark. The

minimax risk is governed by the following critical quantity,

$$I^* = -2\log\left(\sqrt{\frac{a}{n}}\sqrt{\frac{b}{n}} + \sqrt{1-\frac{a}{n}}\sqrt{1-\frac{b}{n}}\right), \tag{14}$$

which is the Rényi divergence of order $\frac{1}{2}$ between Bern $\left(\frac{a}{n}\right)$ and Bern $\left(\frac{b}{n}\right)$, i.e., Bernoulli distributions with success probabilities $\frac{a}{n}$ and $\frac{b}{n}$ respectively. Recall that $0 < \frac{b}{n} < \frac{a}{n} \le 1-\epsilon$ is assumed throughout the paper. It can be shown that $I^* \asymp \frac{(a-b)^2}{na}$. Moreover, when $\frac{a}{n} = o(1)$,

$$\begin{aligned} I^* &= (1+o(1))\frac{(\sqrt{a}-\sqrt{b})^2}{n} = (1+o(1))\left[\left(\sqrt{\frac{a}{n}}-\sqrt{\frac{b}{n}}\right)^2 + \left(\sqrt{1-\frac{a}{n}}-\sqrt{1-\frac{b}{n}}\right)^2\right] \\ &= (2+o(1))H^2\left(\text{Bern}\left(\tfrac{a}{n}\right), \text{Bern}\left(\tfrac{b}{n}\right)\right), \end{aligned}$$

where $H^2(P,Q) = \frac{1}{2}\int(\sqrt{\mathrm{d}P} - \sqrt{\mathrm{d}Q})^2$ is the squared Hellinger distance between two distributions P and Q. The minimax rate for the parameter spaces (2) and (3) under the loss function (5) is given in the following theorem.

Theorem 1 (Zhang and Zhou (2015)) *When $\frac{(a-b)^2}{ak\log k} \to \infty$, we have*

$$\inf_{\widehat{\sigma}} \sup_{(B,\sigma)\in\Theta} \mathbb{E}_{B,\sigma}\ell(\widehat{\sigma},\sigma) = \begin{cases} \exp\left(-(1+\eta)\frac{nI^*}{2}\right), & k=2; \\ \exp\left(-(1+\eta)\frac{nI^*}{\beta k}\right), & k \ge 3, \end{cases}$$

for both $\Theta = \Theta_0(n,k,a,b,\beta)$ and $\Theta = \Theta(n,k,a,b,\lambda,\beta;\alpha)$ with any $\lambda \le \frac{a-b}{2\beta k}$ and any $\beta \in [1,\sqrt{5/3})$, where $\eta = \eta_n \to 0$ is some sequence tending to 0 as $n \to \infty$.

Remark 2 *The assumption $\beta \in [1,\sqrt{5/3})$ is needed in Zhang and Zhou (2015) for some technical reason. Here, the parameter β enters the minimax rates when $k \ge 3$ since the worst case is essentially when one has two communities of size $\frac{n}{\beta k}$, while for $k = 2$, the worst case is essentially two communities of size $\frac{n}{2}$. For all other results in this paper, we allow β to be an arbitrary constant no less than* 1.

Remark 3 *The rate in Theorem 1 is optimal in a minimax sense. It is optimal for the worse-case instances in $\Theta_0(n,k,a,b,1)$. More general instance-optimal fundamental limits are referred to Abbe and Sandon (2015a). Details discussion will be given in Section 5.*

To this end, let us show that the two-stage algorithm proposed in Section 2 achieves the optimal misclassification proportion. The essence of the two-stage algorithm lies in the refinement scheme described in Algorithm 1. As long as any initialization step satisfies a certain weak consistency criterion, the refinement step directly leads to a solution with optimal misclassification proportion. To be specific, the initialization step needs to satisfy the following condition.

Condition 1 *There exist constants $C_0, \delta > 0$ and a positive sequence $\gamma = \gamma_n$ such that*

$$\inf_{(B,\sigma)\in\Theta} \min_{u\in[n]} \mathbb{P}_{B,\sigma}\left\{\ell(\sigma,\sigma_u^0) \le \gamma\right\} \ge 1 - C_0 n^{-(1+\delta)}, \tag{15}$$

for some parameter space Θ.

Under Condition 1, we have the following upper bounds regarding the performance of the proposed refinement scheme.

Theorem 4 *Suppose as $n \to \infty$, $\frac{(a-b)^2}{ak\log k} \to \infty$, $a \asymp b$ and Condition 1 is satisfied for*

$$\gamma = o\left(\frac{1}{k\log k}\right) \tag{16}$$

and $\Theta = \Theta_0(n,k,a,b,\beta)$. Then there is a sequence $\eta \to 0$ such that

$$\begin{aligned} \sup_{(B,\sigma)\in\Theta} \mathbb{P}_{B,\sigma}\left\{\ell(\sigma,\widehat{\sigma}) \ge \exp\left(-(1-\eta)\frac{nI^*}{2}\right)\right\} \to 0, \qquad &\text{if } k = 2, \\ \sup_{(B,\sigma)\in\Theta} \mathbb{P}_{B,\sigma}\left\{\ell(\sigma,\widehat{\sigma}) \ge \exp\left(-(1-\eta)\frac{nI^*}{\beta k}\right)\right\} \to 0, \qquad &\text{if } k \ge 3, \end{aligned} \tag{17}$$

where I^* is defined as in (14).

If in addition Condition 1 is satisfied for γ satisfying both (16) *and*

$$\gamma = o\left(\frac{a-b}{ak}\right) \tag{18}$$

and $\Theta = \Theta(n,k,a,b,\lambda,\beta;\alpha)$, then the conclusion in (17) *continues to hold for $\Theta = \Theta(n,k,a,b,\lambda,\beta;\alpha)$.*

Theorem 4 assumes $a \asymp b$. The case when $a \asymp b$ may not hold is considered in Section 5. Compared with Theorem 1, the upper bounds (17) achieved by Algorithm 1 is minimax optimal. The condition (16) for the parameter space $\Theta_0(n,k,a,b,\beta)$ is very mild. When $k = O(1)$, it reduces to $\gamma = o(1)$ and simply means that the initialization should be weakly consistent. For $k \to \infty$, it implies that the misclassification proportion within each community converges to zero. Note that if the initialization step gives wrong labels to all nodes in one particular community, then the misclassification proportion is at least $1/k$. The condition (16) rules out this situation. For the parameter space $\Theta(n,k,a,b,\lambda,\beta;\alpha)$, an extra condition (18) is required. This is because estimating the connectivity matrix B in $\Theta(n,k,a,b,\lambda,\beta;\alpha)$ is harder than in $\Theta_0(n,k,a,b,\beta)$. If we do not pursue adaptive estimation, (18) is not needed.

Remark 5 *Theorem 4 is an adaptive result without assuming the knowledge of a and b. When these two parameters are known, we can directly use a and b in (11) of Algorithm 1. By scrutinizing the proof of Theorem 4, the conditions (16) and (18) can be weakened to $\gamma = o(k^{-1})$ in this case.*

Given the results of Theorem 4, it remains to check the initialization step via spectral clustering satisfies Condition 1. For matrix $P = (P_{uv}) = (B_{\sigma(u)\sigma(v)})$ with (B, σ) belonging to either $\Theta_0(n, k, a, b, \beta)$ or $\Theta(n, k, a, b, \lambda, \beta; \alpha)$, we use λ_k to denote $\lambda_k(P)$. Define the average degree by

$$\bar{d} = \frac{1}{n} \sum_{u \in [n]} d_u. \tag{19}$$

Theorem 6 *Assume $e \le a \le C_1 b$ for some constant $C_1 > 0$ and*

$$\frac{ka}{\lambda_k^2} \le c, \tag{20}$$

for some sufficiently small $c \in (0, 1)$. Consider USC(τ) with a sufficiently small constant $\mu > 0$ in Algorithm 2 and $\tau = C_2 \bar{d}$ for some sufficiently large constant $C_2 > 0$. For any constant $C' > 0$, there exists some $C > 0$ only depending on C', C_1, C_2 and μ such that

$$\ell(\widehat{\sigma}, \sigma) \le C \frac{a}{\lambda_k^2},$$

with probability at least $1 - n^{-C'}$. If k is fixed, the same conclusion holds without assuming $a \le C_1 b$.

Remark 7 *Theorem 6 improves the error bound for spectral clustering in Lei and Rinaldo (2014). While Lei and Rinaldo (2014) requires the assumption $a > C \log n$, our result also holds for $a = o(\log n)$. A result close to ours is that by Chin et al. (2015), but their clustering step is different from Algorithm 2. Moreover, the conclusion of Theorem 6 holds with probability $1 - n^{-C'}$ for an arbitrary large C', which is critical because the initialization step needs to satisfy Condition 1 for the subsequent refinement step to work. On the other hand, the bound in Chin et al. (2015) is stated with probability $1 - o(1)$.*

Remark 8 *For the parameter space $\Theta_0(n, k, a, b, \beta)$, we have $\lambda_k \ge \frac{a-b}{\beta k}$. Then, Theorem 6 implies that consistency is achieved when $\frac{(a-b)^2}{a} \to \infty$ in the case $k = O(1)$, and when $\frac{(a-b)^2}{ak^3} > C$ for some sufficiently large $C > 0$ in the case $k \to \infty$.*

When $k = O(1)$, Theorem 4 and Theorem 6 jointly imply the following result.

Corollary 9 *Consider Algorithm 1 initialized by σ^0 with USC(τ) for $\tau = C\bar{d}$, where C is a sufficiently large constant. Suppose as $n \to \infty$, $k = O(1)$, $\frac{(a-b)^2}{a} \to \infty$ and $a \asymp b$. Then, there exists a sequence $\eta \to 0$ such that*

$$\sup_{(B,\sigma) \in \Theta} \mathbb{P}_{B,\sigma} \left\{ \ell(\sigma, \widehat{\sigma}) \ge \exp\left(-(1 - \eta) \frac{nI^*}{2} \right) \right\} \to 0, \qquad \text{if } k = 2,$$

$$\sup_{(B,\sigma) \in \Theta} \mathbb{P}_{B,\sigma} \left\{ \ell(\sigma, \widehat{\sigma}) \ge \exp\left(-(1 - \eta) \frac{nI^*}{\beta k} \right) \right\} \to 0, \qquad \text{if } k \ge 3,$$

where the parameter space is $\Theta = \Theta_0(n, k, a, b, \beta)$.

Compared with Theorem 1, the proposed procedure achieves the minimax rate under the condition $\frac{(a-b)^2}{a} \to \infty$ and $a \asymp b$. When $k = O(1)$, the condition $\frac{(a-b)^2}{a} \to \infty$ is necessary and sufficient for weak consistency in view of Theorem 1. More general results including the case of $k \to \infty$ are stated and discussed in Section 5.

The following theorem characterizes the misclassification rate of normalized spectral clustering.

Theorem 10 *Assume $e \le a \le C_1 b$ for some constant $C_1 > 0$ and*

$$\frac{ka \log a}{\lambda_k^2} \le c, \tag{21}$$

for some sufficiently small $c \in (0, 1)$. Consider NSC(τ) with a sufficiently small constant $\mu > 0$ in Algorithm 2 and $\tau = C_2 \bar{d}$ for some sufficiently large constant $C_2 > 0$. Then, for any constant $C' > 0$, there exists some $C > 0$ only depending on C', C_1, C_2 and μ such that

$$\ell(\widehat{\sigma}, \sigma) \le C \frac{a \log a}{\lambda_k^2},$$

with probability at least $1 - n^{-C'}$. If k is fixed, the same conclusion holds without assuming $a \le C_1 b$.

Remark 11 *A slightly different regularization of normalized spectral clustering is studied by Qin and Rohe (2013) only for the dense regime, while Theorem 10 holds under both dense and sparse regimes. Moreover, our result also improves that of Le et al. (2015) due to our tighter bound on $\|L(A_\tau) - L(P_\tau)\|_{\text{op}}$ in Lemma 22 below. We conjecture that the $\log a$ factor in both the assumption and the bound of Theorem 10 can be removed.*

Note that Theorem 6 and Theorem 10 are stated in terms of the quantity λ_k. We may specialize the results into the parameter spaces defined in (2) and (3). By Proposition 24, $\lambda_k \ge \frac{a-b}{2\beta k}$ for $\Theta_0(n, k, a, b, \beta)$ and $\lambda_k \ge \lambda$ for $\Theta(n, k, a, b, \lambda, \beta; \alpha)$. The implications of Theorem 6 and Theorem 10 and their use as initialization in for Algorithm 1 are discussed in full details in Section 5.

4. Numerical results

In this section we present the performance of the proposed algorithm on simulated datasets. The experiments cover three different scenarios: (1) dense network with communities of equal sizes; (2) dense network with communities of unequal sizes; and (3) sparse network. Recall the definition of $\bar{d}$ in (19). For each setting, we report results of Algorithm 1 initialized with four different approaches: USC(∞), USC($2\bar{d}$), NSC(0) and NSC($\bar{d}$), the description of which can all be found in Section 2.3. For all these spectral clustering methods, Algorithm 2 was used to cluster the leading eigenvectors. The constant μ in the critical radius definition was set to be 0.5 in all the results reported here. For each setting, the results are based on 100 independent draws from the underlying stochastic block model.

To achieve faster running time, we also ran a simplified version of Algorithm 1. Instead of obtaining n different initializers $\{\sigma_u\}_{u \in [n]}$ to refine each node separately, the simplified

algorithm refines all the nodes with a single initialization on the whole network. Thus, the running time can be reduced roughly by a factor of n. Simulation results below suggest that the simplified version achieves similar performances to that of Algorithm 1 in all the settings we have considered. For the precise description of the simplified algorithm, we refer readers to Algorithm 3 in the appendix.

Balanced case In this setting, we generate networks with 2500 nodes and 10 communities, each of which consists of 250 nodes, and we set $B_{ii} = 0.48$ for all i and $B_{ij} = 0.32$ for all $i \neq j$. Figure 1 shows the boxplots of the number of misclassified nodes. The first four boxplots correspond to the four different spectral clustering methods, in the order of USC(∞), USC($2\bar{d}$), NSC(0) and NSC($\bar{d}$). The middle four correspond to the results achieved by applying the simplified refinement scheme with these four initialization methods, and the last four show the results of Algorithm 1 with these four initialization methods. Regardless of the initialization method, Algorithm 1 or its simplified version reduces the number of misclassified nodes from around 30 to around 5.

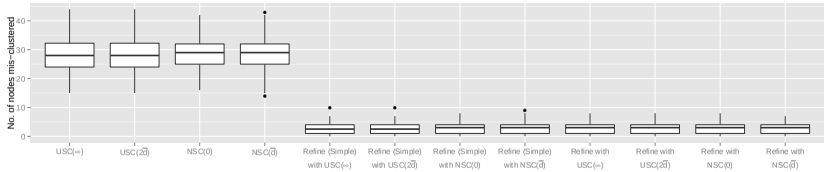


Figure 1: Boxplots of number of misclassified nodes: Balanced case. *Simple* indicates that the simplified version of Algorithm 1 is used instead.

Imbalanced case In this setting, we generate networks with 2000 nodes and 4 communities, the sizes of which are $200, 400, 600$ and 800, respectively. The connectivity matrix is

$$B = \begin{pmatrix} 0.50 & 0.29 & 0.35 & 0.25 \\ 0.29 & 0.45 & 0.25 & 0.30 \\ 0.35 & 0.25 & 0.50 & 0.35 \\ 0.25 & 0.30 & 0.35 & 0.45 \end{pmatrix}.$$

Hence, the within-community edge probability is no smaller than 0.45 while the between-community edge probability is no greater than 0.35, and the underlying SBM is inhomogeneous. Figure 2 shows the boxplots of the number of misclassified nodes obtained by different initialization methods and their refinements, and the boxplots are presented in the same order as those in Figure 1. Similarly, we can see refinement significantly reduces the error.

Sparse case In this setting we consider a much sparser stochastic block model than the previous two cases. In particular, each simulated network has 4000 nodes, divided into 10 communities all of size 400. We set all $B_{ii} = 0.032$ and all $B_{ij} = 0.005$ when

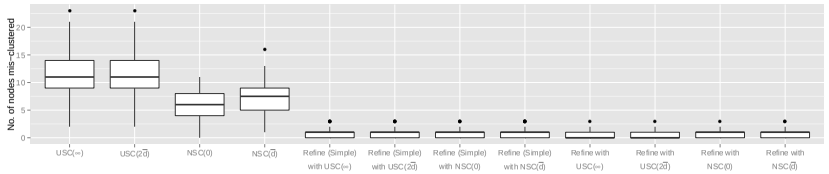


Figure 2: Boxplots of number of misclassified nodes: imbalanced case. *Simple* indicates that the simplified version of Algorithm 1 is used instead.

$i \neq j$. The average degree of each node in the network is around 30. Figure 3 shows the boxplots of the number of misclassified nodes obtained by different initialization methods and their refinements, and the boxplots are presented in the same order as those in Figure 1. Compared with either USC or NSC initialization, refinement reduces the number of misclassified nodes by 50%.

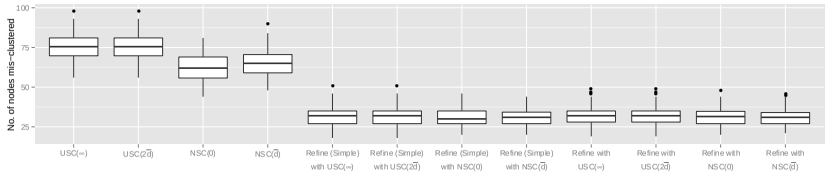


Figure 3: Boxplots of number of misclassified nodes: Sparse case. *Simple* indicates that the simplified version of Algorithm 1 is used instead.

Summary In all three simulation settings, for all four initialization approaches considered, the refinement scheme in Algorithm 1 (and its simplified version) was able to significantly reduce the number of misclassified nodes, which is in agreement with the theoretical properties presented in Section 3.

5. Discussion

In this section, we discuss a few important issues related to the methodology and the theory we have presented in the previous sections.

5.1 Error bounds when $a \asymp b$ may not hold

In Section 3, we established upper bounds on misclassification proportion under the assumption of $a \asymp b$. The following theorem shows that slightly weaker upper bounds can be obtained even when $a \asymp b$ does not hold. To state the result, recall that we assume throughout the paper $\frac{a}{n} \leq 1 - \epsilon$ for some numeric constant $\epsilon \in (0, 1)$.

Theorem 12 *Suppose as $n \to \infty$, $\frac{(a-b)^2}{ak\log k} \to \infty$ and Condition 1 is satisfied for γ satisfying* (16) *and $\Theta = \Theta_0(n,k,a,b,\beta)$. Then for some positive constants c_ϵ and C_ϵ that depend only on ϵ, for any sufficiently small constant $\epsilon_0 \in (0, c_\epsilon)$, if we replace the definition of t_u's in* (11) *with*

$$t_u = \left(\frac{1}{2}\log\frac{\widehat{a}_u(1-\widehat{b}_u/n)}{\widehat{b}_u(1-\widehat{a}_u/n)}\right) \wedge \log\frac{1}{\epsilon_0/2}, \tag{22}$$

then we have

$$\begin{aligned} \sup_{(B,\sigma)\in\Theta} \mathbb{P}_{B,\sigma}\left\{\ell(\sigma,\widehat{\sigma}) \geq \exp\left(-(1-C_\epsilon\epsilon_0)\frac{nI^*}{2}\right)\right\} \to 0, \qquad &\text{if } k = 2, \\ \sup_{(B,\sigma)\in\Theta} \mathbb{P}_{B,\sigma}\left\{\ell(\sigma,\widehat{\sigma}) \geq \exp\left(-(1-C_\epsilon\epsilon_0)\frac{nI^*}{\beta k}\right)\right\} \to 0, \qquad &\text{if } k \geq 3, \end{aligned} \tag{23}$$

where I^* is defined as in (14). *In particular, we can set $C_\epsilon = \frac{10}{3}\frac{2-\epsilon}{\frac{\epsilon}{2}\log\frac{2}{\epsilon}}$ and $c_\epsilon = \min(\frac{1}{10C_\epsilon}, \frac{\epsilon}{2-\epsilon})$.*

If in addition Condition 1 is satisfied for γ satisfying both (16) *and* (18) *and $\Theta = \Theta(n,k,a,b,\lambda,\beta;\alpha)$, then the same conclusion holds for $\Theta = \Theta(n,k,a,b,\lambda,\beta;\alpha)$.*

Compared with the conclusion (17) in Theorem 4, the vanish sequence η in the exponent of the upper bound is replaced by $C_\epsilon\epsilon_0$, which is guaranteed to be smaller than $\min(0.1, \frac{2}{\log(2/\epsilon)})$ and can be driven to be arbitrarily small by decreasing ϵ_0. To achieve this, the t_u's used in defining the penalty parameters in the penalized neighbor voting step need to be truncated at the value $\log\frac{1}{\epsilon_0/2}$.

5.2 Implications of the results

We now discuss some implications of the results in Theorems 4 – 12.

When using USC as initialization for Algorithm 1, we obtain the following results by combining Theorem 4, Theorem 6 and Theorem 12. Recall that $\bar{d}$ is the average degree of nodes in A defined in (19).

Theorem 13 *Consider Algorithm 1 initialized by σ^0 with USC(τ) with $\tau = C\bar{d}$ for some sufficiently large constant $C > 0$. If as $n \to \infty$, $a \asymp b$ and*

$$\frac{(a-b)^2}{ak^3\log k} \to \infty, \tag{24}$$

then there is a sequence $\eta \to 0$ such that (17) *holds with $\Theta = \Theta_0(n,k,a,b,\beta)$. If as $n \to \infty$, $a \asymp b$ and*

$$\frac{\lambda^2}{ak(\log k + a/(a-b))} \to \infty, \tag{25}$$

then (17) *holds for $\Theta = \Theta(n,k,a,b,\lambda,\beta;\alpha)$. If for either parameter space, $a \asymp b$ may not hold but k is fixed and* (24) *or* (25) *holds respectively, then* (23) *holds as long as t_u is replaced by (22) in Algorithm 1.*

Compared with Theorem 1, the minimax optimal performance is achieved under mild conditions. Take $\Theta = \Theta_0(n,k,a,b,\beta)$ for example. For any fixed k, the minimax optimal misclassification proportion is achieved with high probability only under the additional condition of $a \asymp b$. In addition, weak consistency is achieved for fixed k as long as $\frac{(a-b)^2}{a} \to \infty$, regardless of the behavior of $\frac{a}{b}$. This condition is indeed necessary and sufficient for weak consistency. See, for instance, Mossel et al. (2012, 2013b); Yun and Proutiere (2014b); Zhang and Zhou (2015). To achieve strong consistency for fixed k, it suffices to ensure $\ell(\sigma,\widehat{\sigma}) < \frac{1}{n}$ and Theorem 13 implies that it is sufficient to have

$$\liminf_{n\to\infty}\frac{nI^*}{2\log n} > 1, \quad \text{when } k = 2; \quad \liminf_{n\to\infty}\frac{nI^*}{\beta k\log n} > 1, \quad \text{when } k \geq 3, \tag{26}$$

regardless of the behavior of $\frac{a}{b}$. On the other hand, Theorem 1 shows that it is impossible to achieve strong consistency if

$$\limsup_{n\to\infty}\frac{nI^*}{2\log n} < 1, \quad \text{when } k = 2; \quad \limsup_{n\to\infty}\frac{nI^*}{\beta k\log n} < 1, \quad \text{when } k \geq 3. \tag{27}$$

When $\frac{a}{n} = o(1)$, $nI^* = (1+o(1))(\sqrt{a}-\sqrt{b})^2$ and so one can replace nI^* in (26) – (27) with $(\sqrt{a}-\sqrt{b})^2$. In the literature, Abbe et al. (2014) and Mossel et al. (2014) obtained comparable strong consistency results via efficient algorithms for the special case of two communities of equal sizes, i.e., $k = 2$ and $\beta = 1$. Abbe and Sandon (2015a) investigated the case of fixed k and $\beta \geq 1$. Their results give necessary and sufficient conditions for each instance of model parameters. In comparison, our result characterizes minimax optimality through worst case analysis and is less general than those of Abbe and Sandon (2015a). On the other hand, compared with Abbe and Sandon (2015a), we allow any fixed k and any $\beta \geq 1$ without assuming $a \asymp b \asymp \log n$. In the weak consistency regime, in terms of misclassification proportion, for the special case of $k = 2$ and $\beta = 1$, Yun and Proutiere (2014a) achieved the optimal rate for $\Theta_0(n,2,a,b,1)$ when $a \asymp b \asymp a - b$, while the error bounds in other papers are typically off by a constant multiplier on the exponent. In comparison, Theorem 13 provides optimal results (17) and near optimal results (23) for a much broader class of models under much weaker conditions. Last but not least, our algorithm can provably achieve strong consistency and minimax optimal performance even for growing k, which to our limited knowledge, is the first in the literature.

The performance of Algorithm 1 initialized by NSC can be summarized as the following theorem by combining Theorem 4, Theorem 10 and Theorem 12. In this case, the sufficient condition for achieving minimax optimal performance is slightly stronger than when USC is used for initialization.

Theorem 14 *Consider Algorithm 1 initialized by σ^0 with NSC(τ) with $\tau = C\bar{d}$ for some sufficiently large constant $C > 0$. If as $n \to \infty$, $a \asymp b$ and*

$$\frac{(a-b)^2}{ak^3\log k\log a} \to \infty, \tag{28}$$

then there is a sequence $\eta \to 0$ such that (17) *holds with $\Theta = \Theta_0(n,k,a,b,\beta)$. If as $n \to \infty$, $a \asymp b$ and*

$$\frac{\lambda^2}{ak\log a(\log k + a/(a-b))} \to \infty, \tag{29}$$

then (17) *holds for* $\Theta = \Theta(n, k, a, b, \lambda, \beta; \alpha)$. *If for either parameter space,* $a \asymp b$ *may not hold but* k *is fixed and* (28) *or* (29) *holds respectively, then* (23) *holds as long as* t_u *is replaced by (22) in Algorithm 1.*

Last but not least, we would like to point out that when the key parameters a and b are known, we can obtain the desired performance guarantee under weaker conditions as summarized in the following theorem.

Theorem 15 (The case of known a, b**)** *Suppose* a, b *are known. Consider Algorithm 1 initialized by* σ^0 *with USC(*τ*) with* $\tau = Ca$ *for some sufficiently large constant* $C > 0$ *and* $\widehat{a}_u = a$, $\widehat{b}_u = b$ *in* (9) *for all* $u \in [n]$. *If as* $n \to \infty$, $a \asymp b$ *and*

$$\frac{(a-b)^2}{ak^3} \to \infty, \tag{30}$$

then there is a sequence $\eta \to 0$ *such that* (17) *holds with* $\Theta = \Theta_0(n, k, a, b, \beta)$. *If as* $n \to \infty$, $a \asymp b$ *and*

$$\frac{\lambda^2}{ak} \to \infty, \tag{31}$$

then (17) *holds with* $\Theta = \Theta(n, k, a, b, \lambda, \beta; \alpha)$. *If for either parameter space without assuming* $a \asymp b$, (30) *or* (31) *holds respectively, then* (23) *holds if in addition* t_u *is replaced by* (22).

If instead NSC(τ*) is used for initialization with* $\tau = Ca$ *for some sufficiently large constant* $C > 0$, *then the above conclusions hold if we replace* (30) *with* $\frac{(a-b)^2}{ak^3 \log a} \to \infty$ *and* (31) *with* $\frac{\lambda^2}{ak \log a} \to \infty$, *respectively.*

6. Proofs of main results

The main result of the paper, Theorem 4, is proved in Section 6.1. Theorem 6 and Theorem 10 are proved in Section 6.2 and Section 6.3 respectively. The proofs of the remaining results, together with some auxiliary lemmas, are given in the appendix.

6.1 Proof of Theorem 4

We first state a lemma that guarantees the accuracy of parameter estimation in Algorithm 1.

Lemma 16 *Let* $\Theta = \Theta(n, k, a, b, \lambda, \beta; \alpha)$. *Suppose as* $n \to \infty$, $\frac{(a-b)^2}{ak} \to \infty$ *and Condition 1 holds with* γ *satisfying* (16) *and* (18). *Then there is a sequence* $\eta \to 0$ *as* $n \to \infty$ *and a constant* $C > 0$ *such that*

$$\min_{u \in [n]} \inf_{(B,\sigma) \in \Theta} \mathbb{P}\left\{ \min_{\pi \in S_k} \max_{i,j \in [k]} |\widehat{B}^u_{ij} - B_{\pi(i)\pi(j)}| \le \eta \left(\frac{a-b}{n} \right) \right\} \ge 1 - Cn^{-(1+\delta)}. \tag{32}$$

For $\Theta = \Theta_0(n, k, a, b, \beta)$, *the conclusion* (32) *continues to hold even when the assumption* (18) *is dropped.*

Proof 1° Let $\Theta = \Theta(n, k, a, b, \lambda, \beta; \alpha)$. For any community assignments σ_1 and σ_2, define

$$\ell_0(\sigma_1, \sigma_2) = \frac{1}{n} \sum_{u=1}^{n} \mathbf{1}_{\{\sigma_1(u) \neq \sigma_2(u)\}}. \tag{33}$$

Fix any $(B, \sigma) \in \Theta$ and $u \in [n]$. Define event

$$E_u = \left\{ \ell_0(\pi_u(\sigma), \sigma^0_u) \le \gamma \right\}. \tag{34}$$

To simplify notation, assume that $\pi_u = \mathrm{Id}$ is the identity permutation.

Fix any $i \in [k]$. On E_u,

$$n_i \ge |\widetilde{\mathcal{C}}^u_i \cap \mathcal{C}_i| \ge n_i - \gamma_1 n, \ |\widetilde{\mathcal{C}}^u_i \cap \mathcal{C}^c_i| \le \gamma_2 n, \quad \text{where } \gamma_1, \gamma_2 \ge 0 \text{ and } \gamma_1 + \gamma_2 \le \gamma. \tag{35}$$

Let $\mathcal{C}'_i$ be any deterministic subset of $[n]$ such that (35) holds with $\widetilde{\mathcal{C}}^u_i$ replaced by $\mathcal{C}'_i$. By definition, there are at most

$$\sum_{l=0}^{\gamma n} \binom{n_i}{l} \sum_{m=0}^{\gamma n} \binom{n-n_i}{m} \le (\gamma n + 1)^2 \left(\frac{en_i}{\gamma n} \right)^{\gamma n} \left(\frac{en}{\gamma n} \right)^{\gamma n} \le \exp\left\{ 2\log(\gamma n + 1) + 2\gamma n \log \frac{e}{\gamma} \right\}$$
$$\le \exp\left\{ C_1 \gamma n \log \frac{1}{\gamma} \right\}$$

different subsets with this property where $C_1 > 0$ is an absolute constant. Let $\mathcal{E}'_i$ be the edges within $\mathcal{C}'_i$. Then $|\mathcal{E}'_i|$ consists of independent Bernoulli random variables, where at least $(1-\beta\gamma k)^2$ proportion of them follow the $\mathrm{Bern}(B_{ii})$ distribution, at most $(\beta\gamma k)^2$ proportion that are stochastically smaller than $\mathrm{Bern}(\frac{\alpha a}{n})$ and stochastically larger than $\mathrm{Bern}(\frac{a}{n})$, and at most $2\beta\gamma k$ proportion are stochastically smaller than $\mathrm{Bern}(\frac{b}{n})$. Therefore, we obtain that

$$(1-\beta\gamma k)^2 B_{ii} + (\beta\gamma k)^2 \frac{a}{n} \le \mathbb{E}\left[\frac{|\mathcal{E}'_i|}{\frac{1}{2}|\mathcal{C}'_i|(|\mathcal{C}'_i|-1)} \right] \le \max_{t \in [0, \beta\gamma k]} \left\{ (1-t)^2 B_{ii} + t^2 \frac{\alpha a}{n} + 2t \frac{b}{n} \right\}. \tag{36}$$

Note that the LHS is $(1 - (2 + o(1))\beta\gamma k) B_{ii}$. On the other hand, under condition (18), the RHS is attained at $t = 0$ and equals B_{ii} exactly. Thus, we conclude that

$$\left| \mathbb{E}\left[\frac{|\mathcal{E}'_i|}{\frac{1}{2}|\mathcal{C}'_i|(|\mathcal{C}'_i|-1)} \right] - B_{ii} \right| \le C\beta\gamma k \frac{\alpha a}{n} = \eta' \left(\frac{a-b}{n} \right) \tag{37}$$

for some $\eta' \to 0$ that depends only on a, k, α, β and γ, where the last inequality is due to (18).

On the other hand, by Bernstein's inequality, for any $t > 0$,

$$\mathbb{P}\left\{ \left| |\mathcal{E}'_i| - \mathbb{E}|\mathcal{E}'_i| \right| > t \right\} \le 2\exp\left\{ -\frac{t^2}{2(\frac{1}{2}(n_i + \gamma n)^2 \frac{\alpha a}{n} + \frac{2}{3}t)} \right\}.$$

Let

$$t^2 = (n_i+\gamma n)^2\frac{\alpha a}{n}(C_1\gamma n\log\gamma^{-1}+(3+\delta)\log n)\vee(2C_1\gamma n\log\gamma^{-1}+2(3+\delta)\log n)^2$$
$$\lesssim \left(\frac{n}{k}\sqrt{a\gamma\log\gamma^{-1}}+\gamma n\log\gamma^{-1}\right)^2,$$

where we the second inequality holds since $\frac{\log x}{x}$ is monotone decreasing as x increases and so $\gamma\log\gamma^{-1}\geq\frac{1}{n}\log n$ for any $\gamma\geq\frac{1}{n}$, which is the case of most interest since $\gamma<\frac{1}{n}$ leads to $\gamma=0$ and so the initialization is already perfect. Even when $\gamma=0$, we can still continue to the following arguments by replacing every γ with $\frac{1}{n}$ and all the steps continue to hold. Thus, we obtain that for positive constant $C_{\alpha,\beta,\delta}$ that depends only on α,β and δ,

$$\mathbb{P}\left\{\left||\mathcal{E}_i'|-\mathbb{E}|\mathcal{E}_i'|\right|>C_{\alpha,\beta,\delta}\left(\frac{n}{k}\sqrt{a\gamma\log\gamma^{-1}}+\gamma n\log\gamma^{-1}\right)\right\}\leq\exp\left\{-C_1\gamma n\log\gamma^{-1}\right\}n^{-(3+\delta)}. \tag{38}$$

Thus, with probability at least $1-\exp\left\{-C_1\gamma n\log\gamma^{-1}\right\}n^{-(3+\delta)}$,

$$\left|\frac{|\mathcal{E}_i'|}{\frac{1}{2}|\mathcal{C}_i'|(|\mathcal{C}_i'|-1)}-\mathbb{E}\frac{|\mathcal{E}_i'|}{\frac{1}{2}|\mathcal{C}_i'|(|\mathcal{C}_i'|-1)}\right|\leq C_{\alpha,\beta,\delta}\left(\frac{k}{n}\sqrt{a\gamma\log\gamma^{-1}}+\frac{k^2\gamma\log\gamma^{-1}}{n}\right)=\eta'\left(\frac{a-b}{n}\right), \tag{39}$$

where $\eta'\to0$ depends only on a,k,α,β,γ and δ. Here, the last inequality holds since

$$k\sqrt{a\gamma\log\gamma^{-1}}=\sqrt{ak}\sqrt{k\gamma\log\gamma^{-1}},$$

where $\sqrt{ak}\ll a-b$ since $\frac{(a-b)^2}{ak}\to\infty$ and $k\gamma\log\gamma^{-1}=O(1)$, and

$$k^2\gamma\log\gamma^{-1}=k\gamma\log\gamma^{-1}\cdot k\lesssim k\ll\frac{(a-b)^2}{a}\lesssim a-b.$$

We combine (37) and (39) and apply the union bound to obtain that for a sequence $\eta\to0$ that depends only on a,k,α,β,γ and δ, with probability at least $1-n^{-(3+\delta)}$

$$\left|\frac{|\widetilde{\mathcal{E}}_i^u|}{\frac{1}{2}|\widetilde{\mathcal{C}}_i^u|(|\widetilde{\mathcal{C}}_i^u|-1)}-B_{ii}\right|\leq\eta\left(\frac{a-b}{n}\right). \tag{40}$$

The proof for B_{ij} estimation is analogous and hence is omitted. A final union bound on $i,j\in[k]$ leads to the desired claim since all the constants and vanishing sequences in the above analysis depend only on $a,b,k,\alpha,\beta,\gamma$ and δ, but not on u, B or σ.

2° If $\Theta=\Theta_0(n,k,a,b,\beta)$, then condition (18) on γ is no longer needed. This is because (36) can be replaced by

$$\min_{t\in[0,\beta\gamma k]}\left\{(1-t)^2\frac{a}{n}+2t(1-t)\frac{b}{n}+t^2\frac{b}{n}\right\}$$
$$\leq\mathbb{E}\left[\frac{|\mathcal{E}_i'|}{\frac{1}{2}|\mathcal{C}_i'|(|\mathcal{C}_i'|-1)}\right]\leq\max_{t\in[0,\beta\gamma k]}\left\{(1-t)^2\frac{a}{n}+t^2\frac{a}{n}+2t(1-t)\frac{b}{n}\right\},$$

where the LHS equals $\frac{a}{n}-(1-\beta\gamma k(1+o(1)))\frac{a-b}{n}=\frac{a}{n}+o(\frac{a-b}{n})$ and the RHS equals $\frac{a}{n}$. Thus, no additional condition is needed to guarantee (37) in the foregoing arguments. This completes the proof. ■

The next two lemmas establish the desired error bound for the node-wise refinement.

Lemma 17 *Let Θ_0 be defined as in (2) and $k\geq2$. Suppose as $n\to\infty$, $\frac{(a-b)^2}{ak}\to\infty$ and $a\asymp b$. If there exists two sequences $\gamma=o(1/k)$ and $\eta=o(1)$, constants $C,\delta>0$ and permutations $\{\pi_u\}_{u=1}^n\subset S_k$ such that*

$$\inf_{(B,\sigma)\in\Theta_0}\min_{u\in[n]}\mathbb{P}\left\{\ell_0(\pi_u(\sigma),\sigma_u^0)\leq\gamma,\ |\widehat{a}_u-a|\leq\eta(a-b),\ |\widehat{b}_u-b|\leq\eta(a-b)\right\}\geq1-Cn^{-(1+\delta)}. \tag{41}$$

Then for $\widehat{\sigma}_u(u)$ defined as in (10) with $\rho=\rho_u$ in (12), there is a sequence $\eta'=o(1)$ such that for $k=2$,

$$\sup_{(B,\sigma)\in\Theta_0}\max_{u\in[n]}\mathbb{P}\left\{\widehat{\sigma}_u(u)\neq\pi_u(\sigma(u))\right\}\leq(k-1)\exp\left\{-(1-\eta')\frac{nI^*}{2}\right\}+Cn^{-(1+\delta)},$$

and for $k\geq3$,

$$\sup_{(B,\sigma)\in\Theta_0}\max_{u\in[n]}\mathbb{P}\left\{\widehat{\sigma}_u(u)\neq\pi_u(\sigma(u))\right\}\leq(k-1)\exp\left\{-(1-\eta')\frac{nI^*}{\beta k}\right\}+Cn^{-(1+\delta)}.$$

Proof In what follows, let E_u denote the event in (41). For the sake of brevity, we let $p=a/n$, $q=b/n$, $\widehat{p}_u=\widehat{a}_u/n$ and $\widehat{q}_u=\widehat{b}_u/n$. Moreover, let $\sigma_u=\pi_u(\sigma)$, $n_i=|\{v:\sigma_u(v)=i\}|$, $m_i=|\{v:\sigma_u^0(v)=i\}|$ and $m_i'=|\{v:\sigma_u^0(v)=\sigma_u(v)=i\}|$. Without loss of generality, let $\sigma_u(u)=1$.

Then we have

$$\mathbb{P}\left\{\widehat{\sigma}_u(u)\neq1\text{ and }E_u\right\}\leq\sum_{l\neq1}\mathbb{P}\left\{E_u\text{ and }\sum_{\sigma_u(v)=l}A_{uv}-\sum_{\sigma_u(v)=1}A_{uv}\geq\rho_u(m_l-m_1)\right\}=\sum_{l\neq1}p_l. \tag{42}$$

Now we bound each p_l. By the independence structure and Chernoff bound, we have

$$\begin{aligned}p_l &\leq \mathbb{E}\Big\{\exp\left(-t_u\rho_u(m_l-m_1)\right)(qe^{t_u}+1-q)^{m_l'}(pe^{t_u}+1-p)^{m_l-m_l'}\\ &\qquad(pe^{-t_u}+1-p)^{m_1'}(qe^{-t_u}+1-q)^{m_1-m_1'}\mathbf{1}_{\{E_u\}}\Big\} &(43)\\ &\leq \mathbb{E}\left\{\exp\left(-t_u\rho_u(m_l-m_1)\right)(qe^{t_u}+1-q)^{m_l}(pe^{-t_u}+1-p)^{m_1}\mathbf{1}_{\{E_u\}}\right\} &(44)\\ &\quad\times\mathbb{E}\left\{\left(\frac{pe^{t_u}+1-p}{qe^{t_u}+1-q}\right)^{m_l-m_l'}\left(\frac{qe^{-t_u}+1-q}{pe^{-t_u}+1-p}\right)^{m_1-m_1'}\mathbf{1}_{\{E_u\}}\right\}. &(45)\end{aligned}$$

We are going to give bounds for the terms in (44) and (45) respectively. Before doing that, we need some preparatory inequalities. Define t^* through the equation

$$e^{t^*}=\sqrt{\frac{p(1-q)}{q(1-p)}}.$$

Then, on the event E_u,

$$e^{t_u - t^*} + e^{t^* - t_u} \le \exp(C_1 \eta), \tag{46}$$

for some constant $C_1 > 0$. Moreover,

$$|e^{t_u} - 1| \vee |e^{-t_u} - 1| \le C_2 \frac{p-q}{p} = C_2 \frac{a-b}{a}, \tag{47}$$

for some constant $C_2 > 0$. Therefore, for the term in (44), on the event E_u,

$$\begin{aligned} & \exp\left(-t_u \rho_u (m_l - m_1)\right) \left(qe^{t_u} + 1 - q\right)^{m_l} \left(pe^{-t_u} + 1 - p\right)^{m_1} \\ = \; & \exp\left(-t_u \rho_u (m_l - m_1)\right) \left(qe^{t_u} + 1 - q\right)^{(m_l - m_1)/2} \left(pe^{-t_u} + 1 - p\right)^{(m_1 - m_l)/2} \qquad (48) \\ & \times \left(qe^{t_u} + 1 - q\right)^{(m_1 + m_l)/2} \left(pe^{-t_u} + 1 - p\right)^{(m_1 + m_l)/2}. \qquad (49) \end{aligned}$$

By (46), the term in (49) is upper bounded by

$$\begin{aligned} & \left(pq + (1-p)(1-q) + \sqrt{pq}\sqrt{(1-p)(1-q)}(e^{t_u - t^*} + e^{t^* - t_u})\right)^{\frac{m_1 + m_l}{2}} \\ & \le \exp\left(-(1+o(1))\frac{m_1 + m_l}{2} I^*\right) \le \exp\left(-(1+o(1))\frac{n_1 + n_l}{2} I^*\right). \end{aligned}$$

By (47), the term in (48) is upper bounded by

$$\begin{aligned} & \exp\left(-t_u \rho_u (m_l - m_1)\right) \left(qe^{t_u} + 1 - q\right)^{(m_l - m_1)/2} \left(pe^{-t_u} + 1 - p\right)^{(m_1 - m_l)/2} \\ = \; & \exp\left(\frac{m_1 - m_l}{2}\left(\log \frac{pe^{-t_u} + 1 - p}{qe^{t_u} + 1 - q} - \log \frac{\widehat{p}_u e^{-t_u} + 1 - \widehat{p}_u}{\widehat{q}_u e^{t_u} + 1 - \widehat{q}_u}\right)\right) \\ \le \; & \exp\left(\frac{|m_1 - m_l|}{2}\left(|e^{-t_u} - 1||\widehat{p}_u - p| + |e^{t_u} - 1||\widehat{q}_u - q|\right)\right) \\ \le \; & \exp\left(o\left(\frac{n}{k}\frac{(p-q)^2}{p}\right)\right) \\ = \; & \exp\left(o(1)\frac{n_1 + n_l}{2} I^*\right). \end{aligned}$$

Therefore, we can upper bound (44) as

$$\mathbb{E}\left\{e^{-t_u \rho_u (m_l - m_1)} (qe^{t_u} + 1 - q)^{m_l} (pe^{-t_u} + 1 - p)^{m_1} \mathbf{1}_{\{E_u\}}\right\} \le \exp\left(-(1+o(1))\frac{n_1 + n_l}{2} I^*\right). \tag{50}$$

Now we provide an upper bound for (45). By (47), on E_u,

$$\frac{pe^{t_u} + 1 - p}{qe^{t_u} + 1 - q} = 1 + \frac{(p-q)(e^{t_u} - 1)}{qe^{t_u} + 1 - q} \le 1 + O\left(\frac{(p-q)^2}{p}\right) \le \exp\left(O\left(\frac{(p-q)^2}{p}\right)\right),$$

and

$$\frac{qe^{-t_u} + 1 - q}{pe^{-t_u} + 1 - p} = 1 + \frac{(p-q)(1 - e^{-t_u})}{pe^{-t_u} + 1 - p} \le 1 + O\left(\frac{(p-q)^2}{p}\right) \le \exp\left(O\left(\frac{(p-q)^2}{p}\right)\right).$$

Therefore,

$$\mathbb{E}\left\{\left(\frac{pe^{t_u} + 1 - p}{qe^{t_u} + 1 - q}\right)^{m_l - m_l'} \left(\frac{qe^{-t_u} + 1 - q}{pe^{-t_u} + 1 - p}\right)^{m_1 - m_1'} \mathbf{1}_{\{E_u\}}\right\} \le \exp\left(o(1)\frac{n_1 + n_l}{2} I^*\right). \tag{51}$$

By combining (50) and (51), we have

$$p_l \le \exp\left(-(1+o(1))\frac{n_1 + n_l}{2} I^*\right). \tag{52}$$

Using (42), this implies

$$\mathbb{P}\left\{\widehat{\sigma}_u(u) \ne 1 \text{ and } E_u\right\} \le (k-1)\exp\left(-(1+o(1))\min_{l \ne 1}\left(\frac{n_1 + n_l}{2}\right) I^*\right),$$

and so

$$\mathbb{P}\left\{\widehat{\sigma}_u(u) \ne 1\right\} \le (k-1)\exp\left(-(1+o(1))\min_{l \ne 1}\left(\frac{n_1 + n_l}{2}\right) I^*\right) + Cn^{-(1+\delta)}.$$

When $k = 2$, $\min_{l \ne 1}\left(\frac{n_1 + n_l}{2}\right) = \frac{n}{2}$, and when $k \ge 3$, $\min_{l \ne 1}\left(\frac{n_1 + n_l}{2}\right) \ge \frac{n}{\beta k}$. Thus, the proof is complete. ■

Lemma 18 *Let Θ be defined as in (3) and $k \ge 2$. Suppose as $n \to \infty$, $\frac{(a-b)^2}{ak} \to \infty$ and $a \asymp b$. If there exists two sequences $\gamma = o\left(\frac{a-b}{ak}\right)$ and $\eta = o(1)$, constants $C, \delta > 0$ and permutations $\{\pi_u\}_{u=1}^n \subset S_k$ such that (41) holds. Then for $\widehat{\sigma}_u(u)$ defined as in (10) with $\rho = \rho_u$ in (12), the conclusions of Lemma 17 continue to hold.*

Proof The proof is similar to that of Lemma 17 and we use the same notation as there. First, we give a bound for p_l defined in (42). Let $X_j \sim \text{Bern}(q)$, $Y_j \sim \text{Bern}(p)$ and $Z_j \sim \text{Bern}(\alpha p)$, $j \ge 1$, be mutually independent. Then, a stochastic order argument gives

$$\begin{aligned} p_l \; \le \; & \mathbb{E}\left[\mathbb{P}\left\{\sum_{j=1}^{m_l'} X_j + \sum_{j=1}^{m_l - m_l'} Z_j - \sum_{j=1}^{m_1'} Y_j \ge \rho(m_l - m_1) \text{ and } E_u \middle| A_{-u}\right\}\right] \\ \le \; & \mathbb{E}\left\{\exp\left(-t_u \rho_u (m_l - m_1)\right)\left(qe^{t_u} + 1 - q\right)^{m_l}\left(pe^{-t_u} + 1 - p\right)^{m_1} \mathbf{1}_{\{E_u\}}\right\} \qquad (53) \\ & \times \mathbb{E}\left\{\left(\frac{1}{qe^{t_u} + 1 - q}\right)^{m_l - m_l'} \left(\frac{1}{pe^{-t_u} + 1 - p}\right)^{m_1 - m_1'} \right. \qquad (54) \\ & \left. (\alpha p e^{t_u} + 1 - \alpha p)^{m_l - m_l'} \mathbf{1}_{\{E_u\}}\right\}. \end{aligned}$$

Note that the term in (53) is the same as that in (44), and thus it can be upper bounded by (50) as before. To bound for (54), observe that by (47),

$$\frac{1}{qe^{t_u} + 1 - q} \le \exp\left(q|e^{t_u} - 1|\right) \le \exp\left(O(p-q)\right),$$

$$\frac{1}{pe^{-t_u}+1-p} \le \exp\left(Cp|e^{-t_u}-1|\right) \le \exp\left(O(p-q)\right)$$

and

$$\alpha p e^{t_u} + 1 - \alpha p \le \exp\left(\alpha p|e^{t_u}-1|\right) \le \exp\left(O(p-q)\right).$$

Thus, under the assumption $\gamma = o\left(\frac{p-q}{kp}\right)$, the term (54) is bounded by $\exp\left(o(1)\frac{n_1+n_l}{2}I^*\right)$. The remaining proof is the same as that of Lemma 17. ■

Finally, we need a lemma to justify the consensus step in Algorithm 1.

Lemma 19 *For any community assignments σ and σ': $[n] \to [k]$, such that for some constant $C \ge 1$*

$$\min_{l\in[k]} |\{u : \sigma(u) = l\}|, \min_{l\in[k]} |\{u : \sigma'(u) = l\}| \ge \frac{n}{Ck}, \quad \text{and} \quad \min_{\pi\in S_k} \ell_0(\sigma, \pi(\sigma')) < \frac{1}{Ck}.$$

Define map $\xi : [k] \to [k]$ as

$$\xi(i) = \operatorname*{argmax}_l \left|\{u : \sigma(u) = l\} \cap \{u : \sigma'(u) = i\}\right|, \quad \forall i \in [k]. \tag{55}$$

Then $\xi \in S_k$ and $\ell_0(\sigma, \xi(\sigma')) = \min_{\pi\in S_k} \ell_0(\sigma, \pi(\sigma'))$.

Proof By the definition in (55), we obtain

$$\xi = \operatorname*{argmin}_{\xi':[k]\to[k]} \ell_0(\sigma, \xi'(\sigma')), \quad \text{and} \quad \ell_0(\sigma, \xi(\sigma')) \le \min_{\pi\in S_k} \ell_0(\sigma, \pi(\sigma')) < \frac{1}{Ck}.$$

Thus, what remains to be shown is that $\xi \in S_k$, i.e., $\xi(l_1) \ne \xi(l_2)$ for any $l_1 \ne l_2$. To this end, note that if for some $l_1 \ne l_2$, $\xi(l_1) = \xi(l_2)$, then there would exist some $l_0 \in [k]$ such that for any $l \in [k]$, $\xi(l) \ne l_0$, and so

$$\ell_0(\sigma, \xi(\sigma')) \ge \frac{1}{n} \sum_{u:\sigma(u)=l_0} \mathbf{1}_{\{\sigma(u)\ne\xi(\sigma'(u))\}} = \frac{|\{u : \sigma(u) = l_0\}|}{n} \ge \frac{1}{Ck}.$$

This is in contradiction to the second last display, and hence $\xi \in S_k$. This completes the proof. ■

Proof [Proof of Theorem 4] Let $\Theta = \Theta(n, k, a, b, \lambda, \beta; \alpha)$, and fix any $(B, \sigma) \in \Theta$. For any $u \in [n]$, by Condition 1 and the fact that σ_u^0 and $\widehat{\sigma}_u$ differ only at the community assignment of u, for $\gamma' = \gamma + 1/n$, there exists some $\pi_u \in S_k$ such that

$$\mathbb{P}\left\{\ell_0(\sigma, \pi_u^{-1}(\widehat{\sigma}_u)) \le \gamma_n'\right\} \ge 1 - C_0 n^{-(1+\delta)}. \tag{56}$$

Without loss of generality, we assume $\pi_1 = \mathrm{Id}$ is the identity map. Now for any fixed $u \in \{2, \ldots, n\}$, define map $\xi_u : [k] \to [k]$ as in (55) with σ and σ' replaced by $\widehat{\sigma}_1$ and $\widehat{\sigma}_u$. Then by definition

$$\widehat{\sigma}(u) = \xi_u(\widehat{\sigma}_u(u)). \tag{57}$$

In addition, (56) implies with probability at least $1 - Cn^{-(1+\delta)}$, we have

$$\ell_0(\sigma, \widehat{\sigma}_1) \le \gamma' \quad \text{and} \quad \ell_0(\sigma, \pi_u^{-1}(\widehat{\sigma}_u)) \le \gamma'.$$

So the triangle inequality implies $\ell_0(\widehat{\sigma}_1, \pi_u^{-1}(\widehat{\sigma}_u)) \le 2\gamma'$ and hence the condition of Lemma 19 is satisfied. Thus, Lemma 19 implies

$$\mathbb{P}\left\{\xi_u = \pi_u^{-1}\right\} \ge 1 - Cn^{-(1+\delta)}. \tag{58}$$

When $k \ge 3$, Lemma 16, (16) and (18) imply that the condition of Lemma 18 is satisfied, which in turn implies that for a sequence $\eta' = o(1)$,

$$\begin{aligned}
\mathbb{P}\{\widehat{\sigma}(u) \ne \sigma(u)\} &= \mathbb{P}\{\xi_u(\widehat{\sigma}_u(u)) \ne \sigma(u)\} \\
&\le \mathbb{P}\left\{\xi_u(\widehat{\sigma}_u(u)) \ne \sigma(u), \xi_u = \pi_u^{-1}\right\} + \mathbb{P}\left\{\xi_u \ne \pi_u^{-1}\right\} \\
&\le \mathbb{P}\{\widehat{\sigma}_u(u) \ne \pi_u(\sigma(u))\} + \mathbb{P}\left\{\xi_u \ne \pi_u^{-1}\right\} \\
&\le Cn^{-(1+\delta)} + (k-1)\exp\left\{-(1-\eta')\frac{nI^*}{\beta k}\right\}.
\end{aligned}$$

Set

$$\eta = \eta' + \beta\sqrt{\frac{k}{nI^*}} = o(1) \tag{59}$$

where the last inequality holds since $\frac{nI^*}{k} \asymp \frac{(a-b)^2}{ak} \to \infty$. Thus, Markov's inequality leads to

$$\begin{aligned}
&\mathbb{P}\left\{\ell_0(\sigma, \widehat{\sigma}) > (k-1)\exp\left\{-(1-\eta)\frac{nI^*}{\beta k}\right\}\right\} \\
&\quad\le \frac{1}{(k-1)\exp\left\{-(1-\eta)\frac{nI^*}{\beta k}\right\}} \frac{1}{n}\sum_{u=1}^n \mathbb{P}\{\widehat{\sigma}(u) \ne \sigma(u)\} \\
&\quad\le \exp\left\{-(\eta-\eta')\frac{nI^*}{\beta k}\right\} + \frac{Cn^{-(1+\delta)}}{(k-1)\exp\left\{-(1-\eta)\frac{nI^*}{\beta k}\right\}} \\
&\quad\le \exp\left\{-\sqrt{\frac{nI^*}{k}}\right\} + \frac{Cn^{-(1+\delta)}}{(k-1)\exp\left\{-(1-\eta)\frac{nI^*}{\beta k}\right\}}.
\end{aligned}$$

If $(k-1)\exp\left\{-(1-\eta)\frac{nI^*}{\beta k}\right\} \ge n^{-(1+\delta/2)}$, then

$$\mathbb{P}\left\{\ell_0(\sigma, \widehat{\sigma}) > (k-1)\exp\left\{-(1-\eta)\frac{nI^*}{\beta k}\right\}\right\} \le \exp\left\{-\sqrt{\frac{nI^*}{k}}\right\} + Cn^{-\delta/2} = o(1).$$

If $(k-1)\exp\left\{-(1-\eta)\frac{nI^*}{\beta k}\right\} < n^{-(1+\delta/2)}$, then

$$\begin{aligned}
\mathbb{P}\left\{\ell_0(\sigma, \widehat{\sigma}) > (k-1)\exp\left\{-(1-\eta)\frac{nI^*}{\beta k}\right\}\right\} &= \mathbb{P}\{\ell_0(\sigma, \widehat{\sigma}) > 0\} \le \sum_{u=1}^n \mathbb{P}\{\widehat{\sigma}(u) \ne \sigma(u)\} \\
&\le n(k-1)\exp\left\{-(1-\eta)\frac{nI^*}{\beta k}\right\} + Cn^{-\delta} \le Cn^{-\delta/2} = o(1).
\end{aligned}$$

Here, the second last inequality holds since $\eta > \eta'$ and so $(k-1)\exp\{-(1-\eta')nI^*/(\beta k)\} < (k-1)\exp\{-(1-\eta)nI^*/(\beta k)\} < n^{-(1+\delta/2)}$. We complete the proof for the case of $\Theta(n,k,a,b,\lambda,\beta;\alpha)$ and $k \geq 3$ by noting that $(k-1)\exp\left\{-(1-\eta)\frac{nI^*}{\beta k}\right\} = \exp\left\{-(1-\eta'')\frac{nI^*}{\beta k}\right\}$ for another sequence $\eta'' = o(1)$ under the assumption $\frac{(a-b)^2}{ak\log k} \to \infty$ and no constant or sequence in the foregoing arguments involves B, σ or u. When $\Theta = \Theta(n,k,a,b,\lambda,\beta;\alpha)$ and $k = 2$, the foregoing arguments continue to hold with β and k replaced with 1 and 2 respectively.

When $\Theta = \Theta_0(n,k,a,b,\beta)$, we can run the foregoing arguments with Lemma 18 replaced by Lemma 17 to reach the conclusion in (17), which does not require condition (18). This completes the proof. ■

6.2 Proof of Theorem 6

The following lemma is critical to establish the result of Theorem 6. Its proof is given in the appendix. Let us introduce the notation $O(k_1,k_2) = \{V \in \mathbb{R}^{k_1\times k_2} : V^TV = I_{k_2}\}$ for $k_1 \geq k_2$.

Lemma 20 *Consider a symmetric adjacency matrix $A \in \{0,1\}^{n\times n}$ and a symmetric matrix $P \in [0,1]^{n\times n}$ satisfying $A_{uu} = 0$ for all $u \in [n]$ and $A_{uv} \sim Bernoulli(P_{uv})$ independently for all $u > v$. For any $C' > 0$, there exists some $C > 0$ such that*

$$\|T_\tau(A) - P\|_{\mathrm{op}} \leq C\sqrt{np_{\max}+1},$$

with probability at least $1-n^{-C'}$ uniformly over $\tau \in [C_1(np_{\max}+1), C_2(np_{\max}+1)]$ for some sufficiently large constants C_1, C_2, where $p_{\max} = \max_{u\geq v} P_{uv}$.

Lemma 21 *For $P = (P_{uv}) = (B_{\sigma(u)\sigma(v)})$, we have SVD $P = U\Lambda U^T$, where*

$$U = Z\Delta^{-1}W,$$

with $\Delta = diag(\sqrt{n_1}, ..., \sqrt{n_k})$, $Z \in \{0,1\}^{n\times k}$ is a matrix with exactly one nonzero entry in each row at $(i,\sigma(i))$ taking value 1 and $W \in O(k,k)$.

Proof Note that

$$P = ZBZ^T = Z\Delta^{-1}\Delta B\Delta(Z\Delta^{-1})^T,$$

and observe that $Z\Delta^{-1} \in O(n,k)$. Apply SVD to the matrix $\Delta B\Delta^T = W\Lambda W^T$ for some $W \in O(k,k)$, and then we have $P = U\Lambda U^T$ with $U = Z\Delta^{-1}W \in O(k,k)$. ■

Proof [Proof of Theorem 6] Under the current assumption, $\mathbb{E}\tau \in [C_1'a, C_2'a]$ for some large C_1' and C_2'. Using Bernstein's inequality, we have $\tau \in [C_1a, C_2a]$ for some large C_1 and C_2 with probability at least $1-e^{-C'n}$. When (20) holds, by Lemma 20, we deduce that the k^{th} eigenvalue of $T_\tau(A)$ is lower bounded by $c_1\lambda_k$ with probability at least $1-n^{-C'}$ for some small constant $c_1 \in (0,1)$. By Davis–Kahan's sin-theta theorem (Davis and Kahan, 1970),

we have $\|\widehat{U} - UW_1\|_{\mathrm{F}} \leq C\frac{\sqrt{k}}{\lambda_k}\|T_\tau(A) - P\|_{\mathrm{op}}$ for some $W_1 \in O(k,k)$ and some constant $C > 0$. Applying Lemma 21, we have

$$\|\widehat{U} - V\|_{\mathrm{F}} \leq C\frac{\sqrt{k}}{\lambda_k}\|T_\tau(A) - P\|_{\mathrm{op}}, \tag{60}$$

where $V = Z\Delta^{-1}W_2 \in O(n,k)$ for some $W_2 \in O(k,k)$. Combining (60), Lemma 20 and the conclusion $\tau \in [C_1a, C_2a]$, we have

$$\|\widehat{U} - V\|_{\mathrm{F}} \leq \frac{C\sqrt{k}\sqrt{a}}{\lambda_k}, \tag{61}$$

with probability at least $1-n^{-C'}$. The definition of V implies that

$$\|V_{u*} - V_{v*}\| = \sqrt{\frac{1}{n_u}+\frac{1}{n_v}}\mathbf{1}_{\{\sigma(u)\neq\sigma(v)\}}. \tag{62}$$

In other words, define $Q = \Delta^{-1}W_2 \in \mathbb{R}^{k\times k}$ and we have $V_{u*} = Q_{\sigma(u)*}$ for each $u \in [n]$. Hence, for $\sigma(u) \neq \sigma(v)$, $\left\|Q_{\sigma(u)*} - Q_{\sigma(v)*}\right\| = \|V_{u*} - V_{v*}\| \geq \sqrt{\frac{2k}{\beta n}}$. Recall the definition $r = \mu\sqrt{\frac{k}{n}}$ in Algorithm 2. Define the sets

$$T_i = \left\{u \in \sigma^{-1}(i) : \left\|\widehat{U}_{u*} - Q_{i*}\right\| < \frac{r}{2}\right\}, \quad i \in [k].$$

By definition, $T_i \cap T_j = \varnothing$ when $i \neq j$, and we also have

$$\cup_{i\in[k]} T_i = \left\{u \in [n] : \left\|\widehat{U}_{u*} - V_{u*}\right\| < \frac{r}{2}\right\}. \tag{63}$$

Therefore,

$$\left|\left(\cup_{i\in[k]}T_i\right)^c\right|\frac{r^2}{4} \leq \sum_{u\in[n]}\left\|\widehat{U}_{u*} - V_{u*}\right\|^2 \leq \frac{C^2ka}{\lambda_k^2},$$

where the last inequality is by (61). After rearrangement, we have

$$\left|\left(\cup_{i\in[k]}T_i\right)^c\right| \leq \frac{4C^2na}{\mu^2\lambda_k^2}. \tag{64}$$

In other words, most nodes are close to the centers and are in the set (63). Note that the sets $\{T_i\}_{i\in[k]}$ are disjoint. Suppose there is some $i \in [k]$ such that $|T_i| < |\sigma^{-1}(i)| - \left|\left(\cup_{i\in[k]}T_i\right)^c\right|$, we have $\left|\cup_{i\in[k]}T_i\right| = \sum_{i\in[k]}|T_i| < n - \left|\left(\cup_{i\in[k]}T_i\right)^c\right| = \left|\cup_{i\in[k]}T_i\right|$, which is impossible. Thus, the cardinality of T_i for each $i \in [k]$ is lower bounded as

$$|T_i| \geq |\sigma^{-1}(i)| - \left|\left(\cup_{i\in[k]}T_i\right)^c\right| \geq \frac{n}{\beta k} - \frac{4C^2na}{\mu^2\lambda_k^2} > \frac{n}{2\beta k}, \tag{65}$$

where the last inequality above is by the assumption (20). Intuitively speaking, except for a negligible proportion, most data points in $\{\widehat{U}_{u*}\}_{u\in[n]}$ are very close to the population

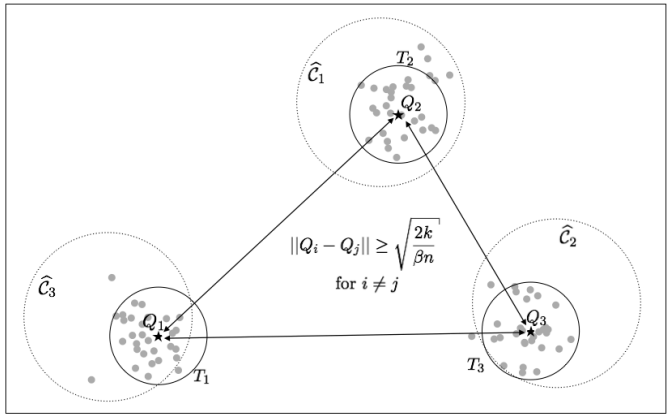


Figure 4: The schematic plot for the proof of Theorem 6. The balls $\{T_i\}_{i\in[k]}$ are centered at $\{Q_i\}_{i\in[k]}$, and the centers are at least $\sqrt{\frac{2k}{\beta n}}$ away from each other. The balls $\{\widehat{\mathcal{C}}_i\}_{i\in[k]}$ intersect with large proportions of $\{T_i\}_{i\in[k]}$, and their subscripts do not need to match due to some permutation.

centers $\{Q_{i*}\}_{i\in[k]}$. Since the centers are at least $\sqrt{\frac{2k}{\beta n}}$ away from each other and $\{T_i\}_{i\in[k]}$ and $\{\widehat{C}_i\}_{i\in[k]}$ are both defined through the critical radius $r = \mu\sqrt{\frac{k}{n}}$ for a small μ, each $\widehat{C}_i$ should intersect with only one T_i (see Figure 4). We claim that there exists some permutation π of the set $[k]$, such that for $\widehat{C}_i$ defined in Algorithm 2,

$$\widehat{\mathcal{C}}_i \cap T_{\pi(i)} \neq \varnothing \quad \text{and} \quad |\widehat{\mathcal{C}}_i| \geq |T_{\pi(i)}| \quad \text{for each } i \in [k]. \tag{66}$$

In what follows, we first establish the result of Theorem 6 by assuming (66). The proof of (66) will be given in the end. Note that for any $i \neq j$, $T_{\pi(i)} \cap \widehat{\mathcal{C}}_j = \varnothing$, which is deduced from the fact that $\widehat{\mathcal{C}}_j \cap T_{\pi(j)} \neq \varnothing$ and the definition of $\widehat{\mathcal{C}}_j$. Therefore, $T_{\pi(i)} \subset \widehat{\mathcal{C}}_j^c$ for all $j \neq i$. Combining with the fact that $T_{\pi(i)} \cap \widehat{\mathcal{C}}_i^c \subset \widehat{\mathcal{C}}_i^c$, we get $T_{\pi(i)} \cap \widehat{\mathcal{C}}_i^c \subset (\cup_{i\in[k]}\widehat{\mathcal{C}}_i)^c$. Therefore,

$$\cup_{i\in[k]} \left(T_{\pi(i)} \cap \widehat{\mathcal{C}}_i^c\right) \subset \left(\cup_{i\in[k]}\widehat{\mathcal{C}}_i\right)^c. \tag{67}$$

Since $T_i \cap T_j = \varnothing$ for $i \neq j$, we deduce from (67) that

$$\sum_{i\in[k]} \left|T_{\pi(i)} \cap \widehat{\mathcal{C}}_i^c\right| \leq \left|\left(\cup_{i\in[k]}\widehat{\mathcal{C}}_i\right)^c\right|. \tag{68}$$

By definition, $\widehat{\mathcal{C}}_i \cap \widehat{\mathcal{C}}_j = \varnothing$ for $i \neq j$, we deduce from (66) that

$$\left|\left(\cup_{i\in[k]}\widehat{\mathcal{C}}_i\right)^c\right| = n - \sum_{i\in[k]} |\widehat{\mathcal{C}}_i| \leq n - \sum_{i\in[k]} |T_i| = \left|\left(\cup_{i\in[k]}T_i\right)^c\right|. \tag{69}$$

Combining (68), (69) and (64), we have

$$\sum_{i\in[k]} \left|T_{\pi(i)} \cap \widehat{\mathcal{C}}_i^c\right| \leq \frac{4C^2na}{\mu^2\lambda_k^2}. \tag{70}$$

Since for any $u \in \cup_{i\in[k]}(\widehat{\mathcal{C}}_i \cap T_{\pi(i)})$, we have $\widehat{\sigma}(u) = i$ when $\sigma(u) = \pi(i)$, the mis-classification rate is bounded as

$$\begin{aligned}
\ell_0(\widehat{\sigma}, \pi^{-1}(\sigma)) &\leq \frac{1}{n}\left|\left(\cup_{i\in[k]}(\widehat{\mathcal{C}}_i \cap T_{\pi(i)})\right)^c\right| \\
&\leq \frac{1}{n}\left(\left|\left(\cup_{i\in[k]}(\widehat{\mathcal{C}}_i \cap T_{\pi(i)})\right)^c \cap \left(\cup_{i\in[k]}T_i\right)\right| + \left|\left(\cup_{i\in[k]}T_i\right)^c\right|\right) \\
&\leq \frac{1}{n}\left(\sum_{i\in[k]}\left|T_{\pi(i)} \cap \widehat{\mathcal{C}}_i^c\right| + \left|\left(\cup_{i\in[k]}T_i\right)^c\right|\right) \\
&\leq \frac{8C^2a}{\mu^2\lambda_k^2},
\end{aligned}$$

where the last inequality is from (70) and (64). This proves the desired conclusion.

Finally, we are going to establish the claim (66) to close the proof. We use mathematical induction. For $i = 1$, it is clear that $|\widehat{\mathcal{C}}_1| \geq \max_{i\in[k]}|T_i|$ holds by the definition of $\widehat{\mathcal{C}}_1$. Suppose $\widehat{\mathcal{C}}_1 \cap T_i = \varnothing$ for all $i \in [k]$, and then we must have

$$\left|\left(\cup_{i\in[k]}T_i\right)^c\right| \geq |\widehat{\mathcal{C}}_1| \geq \max_{i\in[k]}|T_i| \geq \frac{n}{2\beta k},$$

where the last inequality is by (65). This contradicts (64) under the assumption (20). Therefore, there must be a $\pi(1)$ such that $\widehat{\mathcal{C}}_1 \cap T_{\pi(1)} \neq \varnothing$ and $|\widehat{\mathcal{C}}_1| \geq |T_{\pi(1)}|$. Moreover,

$$\begin{aligned}
|\widehat{\mathcal{C}}_1^c \cap T_{\pi(1)}| &= |T_{\pi(1)}| - |T_{\pi(1)} \cap \widehat{\mathcal{C}}_1| \\
&\leq |\widehat{\mathcal{C}}_1| - |T_{\pi(1)} \cap \widehat{\mathcal{C}}_1| \\
&= |\widehat{\mathcal{C}}_1 \cap T_{\pi(1)}^c| \\
&\leq \left|\left(\cup_{i\in[k]}T_i\right)^c\right|,
\end{aligned}$$

where the last inequality is because $T_{\pi(1)}$ is the only set in $\{T_i\}_{i\in[k]}$ that intersects $\widehat{\mathcal{C}}_1$ by the definitions. By (64), we get

$$|\widehat{\mathcal{C}}_i^c \cap T_{\pi(i)}| \leq \frac{4C^2na}{\mu^2\lambda_k^2}, \tag{71}$$

for $i = 1$.

Now suppose (66) and (71) are true for $i = 1, ..., l-1$. Because of the sizes of $\{\widehat{\mathcal{C}}_i\}_{i\in[l-1]}$ and the fact that $\{T_i\}_{i\in[k]}$ are mutually exclusive, we have

$$\left(\cup_{i=1}^{l-1}\widehat{\mathcal{C}}_i\right) \cap \left(\cup_{i\in[k]\setminus\cup_{i=1}^{l-1}\{\pi(i)\}}T_i\right) = \varnothing.$$

Therefore, for the set S in the current step, $\cup_{i\in[k]\setminus\cup_{i=1}^{l-1}\{\pi(i)\}}T_i \subset S$. By the definition of $\widehat{\mathcal{C}}_l$, we have $|\widehat{\mathcal{C}}_l| \geq \max_{i\in[k]\setminus\cup_{i=1}^{l-1}\{\pi(i)\}}|T_i| \geq \frac{n}{2\beta k}$. Suppose $\widehat{\mathcal{C}}_l \cap T_{\pi(i)} \neq \varnothing$ for some $i = 1, ..., l-1$. Then, this $T_{\pi(i)}$ is the only set in $\{T_i\}_{i\in[k]}$ that intersects $\widehat{\mathcal{C}}_l$ by their definitions. This implies that

$$|\widehat{\mathcal{C}}_l| \leq |\widehat{\mathcal{C}}_l \cap T_{\pi(i)}| + \left|\left(\cup_{i\in[k]}T_i\right)^c\right|.$$

Since $\widehat{\mathcal{C}}_l \cap \widehat{\mathcal{C}}_{\pi(i)} = \varnothing$, $|\widehat{\mathcal{C}}_l \cap T_{\pi(i)}| \leq |\widehat{\mathcal{C}}_i^c \cap T_{\pi(i)}|$ is bounded by (71). Together with (64), we have

$$|\widehat{\mathcal{C}}_l| \leq \frac{8C^2na}{\mu^2\lambda_k^2},$$

which contradicts $|\widehat{\mathcal{C}}_l| \geq \frac{n}{2\beta k}$ under the assumption (20). Therefore, we must have $\widehat{\mathcal{C}}_l \cap T_{\pi(i)} = \varnothing$ for all $i = 1, ..., l-1$. Now suppose $\widehat{\mathcal{C}}_l \cap T_{\pi(i)} = \varnothing$ for all $i \in [k]$, we must have

$$\left|\left(\cup_{i\in[k]}T_i\right)^c\right| \geq |\widehat{\mathcal{C}}_l| \geq \frac{n}{2\beta k},$$

which contradicts (64). Hence, $\widehat{\mathcal{C}}_l \cap T_{\pi(l)} \neq \varnothing$ for some $\pi(l) \in [k]\setminus\cup_{i=1}^{l-1}\{\pi(i)\}$, and (66) is established for $i = l$. Moreover, (71) can also be established for $i = l$ by the same argument that is used to prove (71) for $i = 1$. The proof is complete. ■

6.3 Proof of Theorem 10

Define $P_\tau = P + \frac{\tau}{n}\mathbf{1}\mathbf{1}^T$. The proof of the following lemma is given in the appendix.

Lemma 22 *Consider a symmetric adjacency matrix $A \in \{0,1\}^{n\times n}$ and a symmetric matrix $P \in [0,1]^{n\times n}$ satisfying $A_{uu} = 0$ for all $u \in [n]$ and $A_{uv} \sim Bernoulli(P_{uv})$ independently for all $u > v$. For any $C' > 0$, there exists some $C > 0$ such that*

$$\|L(A_\tau) - L(P_\tau)\|_{\text{op}} \leq C\sqrt{\frac{\log(e(np_{\max}+1))}{np_{\max}+1}},$$

with probability at least $1-n^{-C'}$ uniformly over $\tau \in [C_1(np_{\max}+1), C_2(np_{\max}+1)]$ for some sufficiently large constants C_1, C_2, where $p_{\max} = \max_{u\geq v}P_{uv}$.

Lemma 23 *Consider $P = (P_{uv}) = (B_{\sigma(u)\sigma(v)})$. Let the SVD of the matrix $L(P_\tau)$ be $L(P_\tau) = U\Sigma U^T$, with $U \in O(n,k)$ and $\Sigma = diag(\sigma_1, ..., \sigma_k)$. For $V = UW$ with any $W \in O(r,r)$, we have $\|V_{u*} - V_{v*}\| = \sqrt{\frac{1}{n_u}+\frac{1}{n_v}}$ when $\sigma(u) \neq \sigma(v)$ and $V_{u*} = V_{v*}$ when $\sigma(u) = \sigma(v)$. Moreover, $\sigma_k \geq \frac{\lambda_k}{2\tau}$ as long as $\tau \geq np_{\max}$.*

Proof The first part is Lemma 1 in Joseph and Yu (2013). Define $\bar{d}_v = \sum_{u\in[n]}P_{uv}$ and $\bar{D}_\tau = \text{diag}(\bar{d}_1+\tau, ..., \bar{d}_n+\tau)$. Then, we have $L(P_\tau) = \bar{D}_\tau^{-1/2}P_\tau\bar{D}_\tau^{-1/2}$. Note that P_τ has an SBM structure so that it has rank at most k, and the k^{th} eigenvalue of P_τ is lower bounded by λ_k. Thus, we have

$$\sigma_k \geq \frac{\lambda_k}{\max_{u\in[n]}\bar{d}_u + \tau}.$$

Observe that $\max_{u\in[n]}\bar{d}_u \leq np_{\max} \leq \tau$, and the proof is complete. ■

Proof [Proof of Theorem 10] As is shown in the proof of Theorem 6, $\tau \in [C_1a, C_2a]$ for some large C_1, C_2 with probability at least $1-e^{-C'n}$. By Davis–Kahan's sin-theta theorem (Davis and Kahan, 1970), we have $\|\widehat{U} - UW\|_{\text{F}} \leq C_1\frac{\sqrt{k}}{\sigma_k}\|L(A_\tau) - L(P_\tau)\|_{\text{op}}$ for some $W \in O(r,r)$ and some constant $C_1 > 0$. Let $V = UW$ and apply Lemma 22 and Lemma 23, we have

$$\|\widehat{U} - V\|_{\text{F}} \leq \frac{C\sqrt{k}\sqrt{a\log a}}{\lambda_k}, \tag{72}$$

with probability at least $1-n^{-C'}$. Note that by Lemma 23, V satisfies (62). Replace (61) by (72), and follow the remaining proof of Theorem 6, the proof is complete. ■

Appendix A. A simplified version of Algorithm 1

We give in Algorithm 3 the precise description of the simplified version of Algorithm 1 that we have used in Section 4.

A.1 Additional simulation results

We report some additional simulation results comparing the performances of Algorithm 1 and Algorithm 3. In particular, we consider networks with 400 nodes and 2 communities with four different sets of community sizes, within and between community connection probabilities and initialization methods, and the simulation results are reported in Figure 5 on page 34. From Figure 5 on page 34 we can see the performances of both algorithms remain similar across all these different settings.

Appendix B. Proofs of Theorem 12

Proof [Proof of Theorem 12] Let us consider $\Theta = \Theta_0(n,k,a,b,\beta)$ and the case of $\Theta(n,k,a,b,\lambda,\beta;\alpha)$ is similar except that the condition (18) is needed to establish the counterpart of Lemma 18. The proof essentially follows the same steps as those in the proof of Theorem 4. First, we note that Lemma 16 continues to hold since it does not need the assumption of a/b being bounded. Thus, the first job is to establish the counterpart of Lemma 17 with η' replaced with $C_\epsilon\frac{2\epsilon_0}{3}$. As before, let $p = a/n$ and $q = b/n$.

To this end, we first proceed in the same way to obtain (42) – (52). Without loss of generality, let us consider the case where $t^* > \log\frac{2}{\epsilon_0}$ and $t_u = \log\frac{2}{\epsilon_0}$ since otherwise we can essentially repeat the proof of Theorem 4. Note that this implies $\frac{a}{b} > (\frac{2}{\epsilon_0})^2$. In this case, with the new t_u in (22), we have on the event E_u,

$$(qe^{t_u}+1-q)(pe^{-t_u}+1-p) = e^{-I'}$$

Algorithm 3: A simplified refinement scheme for community detection

Input: Adjacency matrix $A \in \{0,1\}^{n\times n}$,
number of communities k,
initial community detection method σ^0.

Output: Community assignment $\widehat{\sigma}$.

Initialization:

Apply σ^0 on A to obtain $\sigma^0(u)$ for all $u \in [n]$;

Define $\widetilde{\mathcal{C}}_i = \{v : \sigma^0(v) = i\}$ for all $i \in [k]$; let $\widetilde{\mathcal{E}}_i$ be the set of edges within $\widetilde{\mathcal{C}}_i$, and $\widetilde{\mathcal{E}}_{ij}$ the set of edges between $\widetilde{\mathcal{C}}_i$ and $\widetilde{\mathcal{C}}_j$ when $i \neq j$;

Define

$$\widehat{B}_{ii} = \frac{|\widetilde{\mathcal{E}}_i|}{\frac{1}{2}|\widetilde{\mathcal{C}}_i|(|\widetilde{\mathcal{C}}_i|-1)}, \quad \widehat{B}_{ij} = \frac{|\widetilde{\mathcal{E}}_{ij}|}{|\widetilde{\mathcal{C}}_i||\widetilde{\mathcal{C}}_j|}, \quad \forall i \neq j \in [k],$$

and let

$$\widehat{a} = n \min_{i\in[k]} \widehat{B}_{ii} \quad \text{and} \quad \widehat{b} = n \max_{i\neq j\in[k]} \widehat{B}_{ij}.$$

Penalized neighbor voting:

For

$$t = \frac{1}{2}\log\frac{\widehat{a}(1-\widehat{b}/n)}{\widehat{b}(1-\widehat{a}/n)},$$

define

$$\rho = -\frac{1}{2t}\log\left(\frac{\frac{\widehat{a}}{n}e^{-t}+1-\frac{\widehat{a}}{n}}{\frac{\widehat{b}}{n}e^{t}+1-\frac{\widehat{b}}{n}}\right),$$

For each $u \in [n]$, set

$$\widehat{\sigma}(u) = \operatorname*{argmax}_{l\in[k]} \sum_{\sigma^0(v)=l} A_{uv} - \rho \sum_{v\in[n]} \mathbf{1}_{\{\sigma^0(v)=l\}}.$$

where

$$\begin{aligned} I' &= -\log\left((1-p)(1-q)+pq+\left(e^{t_u-t^*}+e^{t^*-t_u}\right)\sqrt{(1-p)(1-q)pq}\right) \\ &\geq \left(1-C_\epsilon\frac{3\epsilon_0}{5}\right)I^*. \end{aligned} \tag{73}$$

To see this, we first note that for any $x, y \in (0,1)$ and sufficient small constant $c_0 > 0$, if $y \geq x \geq (1-c_0)y$ and $\frac{y-x}{1-y} \leq 1$, then

$$-\log(1-x) = -\log(1-y) - \log\left(1+\frac{y-x}{1-y}\right) \geq -\log(1-y) - 2\frac{y-x}{1-y} \geq -\left(1-C'_y c_0\right)\log(1-y),$$

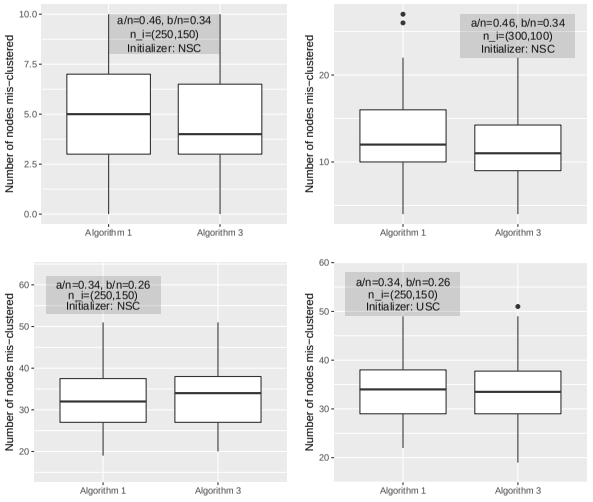


Figure 5: Comparison of Algorithm 1 and its simplified version (Algorithm 3) under various parameters for two-community networks. The parameters are displayed inside each boxplot. Here n_i gives the community sizes and B matrix satisfies $B_{ii} = a/n, \forall i = 1,2$ and $B_{ij} = b/n, \forall i \neq j$. Each pair of boxplots are based on 100 runs.

where $C'_y = \frac{2y}{-(1-y)\log(1-y)}$. When $\frac{a}{b} > (\frac{2}{\epsilon_0})^2$ and $t_u = \log\frac{2}{\epsilon_0}$, we have $I' = -\log(1-x)$ for

$$x = p+q-2pq-(e^{t_u-t^*}+e^{t^*-t_u})\sqrt{(1-p)(1-q)pq} \geq p-2pq-qe^{t_u}-pe^{-t_u} \geq p(1-\epsilon_0-\frac{\epsilon_0^2}{2}),$$

while $I^* = -\log(1-y)$ for

$$y = p+q-2pq-2\sqrt{(1-p)(1-q)pq} \leq p+q \leq p(1+(\frac{\epsilon_0}{2})^2).$$

Thus, for any $\epsilon_0 \in (0, c_\epsilon)$, $1-\frac{\epsilon}{2} \geq y \geq x \geq (1-2\epsilon_0)y$ and $\frac{y-x}{1-y} \leq 1$, and we apply the inequality in the third last display to obtain (73).

Thus, the term in (49) is upper bounded by

$$\exp\left(-\left(1-C_\epsilon\frac{3\epsilon_0}{5}\right)\frac{n_1+n_l}{2}I^*\right).$$

On the other hand, since $|e^{-t_u}-1| \leq 1$, $|e^{t_u}-1|$ is bounded and $\frac{p-q}{p} \asymp 1$, the term in (48) continues to be bounded by

$$\exp\left(-o(1)\frac{n_1+n_l}{2}I^*\right).$$

Moreover, by the same argument as in Lemma 17, (51) continues to hold. Thus, we can replace (52) as

$$p_l \le \exp\left(-\big(1-C_\epsilon\frac{2\epsilon_0}{3}\big)\frac{n_1+n_l}{2}I^*\right),$$

and so when $k\ge 3$,

$$\mathbb{P}\left\{\widehat{\sigma}_u(u)\neq\pi_u(\sigma(u))\right\}\le(k-1)\exp\left\{-\left(1-C_\epsilon\frac{2\epsilon_0}{3}\right)\frac{nI^*}{\beta k}\right\}+Cn^{-(1+\delta)} \tag{74}$$

and when $k=2$, we can replace β by 1 in the last display.

When $k\ge 3$, given the last display and (58), we have

$$\begin{aligned}\mathbb{P}\left\{\widehat{\sigma}(u)\neq\sigma(u)\right\}&=\mathbb{P}\left\{\xi_u(\widehat{\sigma}_u(u))\neq\sigma(u)\right\}\\&\le\mathbb{P}\left\{\xi_u(\widehat{\sigma}_u(u))\neq\sigma(u),\,\xi_u=\pi_u^{-1}\right\}+\mathbb{P}\left\{\xi_u\neq\pi_u^{-1}\right\}\\&\le\mathbb{P}\left\{\widehat{\sigma}_u(u)\neq\pi_u(\sigma(u))\right\}+\mathbb{P}\left\{\xi_u\neq\pi_u^{-1}\right\}\\&\le Cn^{-(1+\delta)}+(k-1)\exp\left\{-\left(1-C_\epsilon\frac{2\epsilon_0}{3}\right)\frac{nI^*}{\beta k}\right\}.\end{aligned} \tag{75}$$

Thus, the assumption that $\frac{(a-b)^2}{ak\log k}\to\infty$ and Markov's inequality leads to

$$\begin{aligned}&\mathbb{P}\left\{\ell_0(\sigma,\widehat{\sigma})>\exp\left\{-(1-C_\epsilon\epsilon_0)\frac{nI^*}{\beta k}\right\}\right\}\\&\quad\le\mathbb{P}\left\{\ell_0(\sigma,\widehat{\sigma})>(k-1)\exp\left\{-(1-C_\epsilon\frac{5\epsilon_0}{6})\frac{nI^*}{\beta k}\right\}\right\}\\&\quad\le\frac{1}{(k-1)\exp\left\{-(1-C_\epsilon\frac{5\epsilon_0}{6})\frac{nI^*}{\beta k}\right\}}\frac{1}{n}\sum_{u=1}^n\mathbb{P}\left\{\widehat{\sigma}(u)\neq\sigma(u)\right\}\\&\quad\le\exp\left\{-\frac{C_\epsilon\epsilon_0}{6}\frac{nI^*}{\beta k}\right\}+\frac{Cn^{-(1+\delta)}}{(k-1)\exp\left\{-(1-C_\epsilon\frac{5\epsilon_0}{6})\frac{nI^*}{\beta k}\right\}}.\end{aligned} \tag{76}$$

If $(k-1)\exp\left\{-(1-C_\epsilon\frac{5\epsilon_0}{6})\frac{nI^*}{\beta k}\right\}\ge n^{-(1+\delta/2)}$, then

$$\mathbb{P}\left\{\ell_0(\sigma,\widehat{\sigma})>\exp\left\{-(1-C_\epsilon\epsilon_0)\frac{nI^*}{\beta k}\right\}\right\}\le\exp\left\{-\frac{C_\epsilon\epsilon_0}{6}\frac{nI^*}{\beta k}\right\}+Cn^{-\delta/2}=o(1). \tag{77}$$

If $(k-1)\exp\left\{-(1-C_\epsilon\frac{5\epsilon_0}{6})\frac{nI^*}{\beta k}\right\}< n^{-(1+\delta/2)}$, then

$$\begin{aligned}\mathbb{P}\left\{\ell_0(\sigma,\widehat{\sigma})>\exp\left\{-(1-C_\epsilon\epsilon_0)\frac{nI^*}{\beta k}\right\}\right\}&\le\mathbb{P}\left\{\ell_0(\sigma,\widehat{\sigma})>0\right\}\le\sum_{u=1}^n\mathbb{P}\left\{\widehat{\sigma}(u)\neq\sigma(u)\right\}\\&\le n(k-1)\exp\left\{-(1-C_\epsilon\frac{2\epsilon_0}{3})\frac{nI^*}{\beta k}\right\}+Cn^{-\delta}\le Cn^{-\delta/2}=o(1).\end{aligned} \tag{78}$$

Here, the second last inequality holds since $(k-1)\exp\left\{-(1-C_\epsilon\frac{2\epsilon_0}{3})\frac{nI^*}{\beta k}\right\}<\exp\left\{-(1-C_\epsilon\frac{5\epsilon_0}{6})\frac{nI^*}{\beta k}\right\}<n^{-(1+\delta/2)}$. We complete the proof for the case of $k\ge 3$ by noting that no constant or sequence in the foregoing arguments involves B,σ or u. When $k=2$, we run the foregoing arguments with β replaced by 1 to obtain the desired claim. ■

Appendix C. Proofs of Theorems 13, 14 and 15

Proposition 24 *For SBM in the space $\Theta_0(n,k,a,b,\beta)$ satisfying $n\ge 2\beta k$, we have $\lambda_k\ge\frac{a-b}{\beta k}$.*

Proof Since the eigenvalues of P are invariant with respect to permutation of the community labels, we consider the case where $\sigma(u)=i$ for $u\in\left\{\sum_{j=1}^{i-1}n_j-1,\sum_{j=1}^i n_j\right\}$ without loss of generality, where $\sum_{j=1}^0 n_j=0$. Let us use the notation $\mathbf{1}_d\in\mathbb{R}^d$ and $\mathbf{0}_d\in\mathbb{R}^d$ to denote the vectors with all entries being 1 and 0 respectively. Then, it is easy to check that

$$P-\frac{b}{n}\mathbf{1}_n\mathbf{1}_n^T=\frac{a-b}{n}\sum_{i=1}^k v_iv_i^T,$$

where $v_1=(\mathbf{1}_{n_1}^T,\mathbf{0}_{n_2}^T,...,\mathbf{0}_{n_k}^T)^T$, $v_1=(\mathbf{0}_{n_1}^T,\mathbf{1}_{n_2}^T,\mathbf{0}_{n_3}^T,...,\mathbf{0}_{n_k}^T)^T$,..., $v_k=(\mathbf{0}_{n_1}^T,...,\mathbf{0}_{n_{k-1}}^T,\mathbf{1}_{n_k}^T)^T$. Note that $\{v_i\}_{i=1}^k$ are orthogonal to each other, and therefore

$$\lambda_k\left(\sum_{i=1}^k v_iv_i^T\right)\ge\min_{i\in[k]}n_i\ge\frac{n}{\beta k}-1\ge\frac{n}{2\beta k}.$$

By Weyl's inequality (Theorem 4.3.1 of Horn and Johnson (2012)),

$$\lambda_k(P)\ge\frac{a-b}{n}\lambda_k\left(\sum_{i=1}^k v_iv_i^T\right)+\lambda_n\left(\frac{b}{n}\mathbf{1}_n\mathbf{1}_n^T\right)\ge\frac{a-b}{2\beta k}.$$

This completes the proof. ■

Proof [Proof of Theorem 13] Let us first consider $\Theta_0(n,k,a,b,\beta)$. By Theorem 6 and Proposition 24, the misclassification proportion is bounded by $C\frac{k^2a}{(a-b)^2}$ under the condition $\frac{k^3a}{(a-b)^2}\le c$ for some small c. Thus, Condition 1 holds when $\frac{k^3a}{(a-b)^2}=o(1)$, which leads to the desired conclusion in view of Theorem 4 and Theorem 12. The proof of the space $\Theta(n,k,a,b,\lambda,\beta;\alpha)$ follows the same argument. ■

Proof [Proof of Theorem 14] The proof is the same as that of Theorem 13. ■

Proof [Proof of Theorem 15] When the parameters a and b are known, we can use $\tau=Ca$ for some sufficiently large $C>0$ for both USC(τ) and NSC(τ). Then, the results of Theorem 6 and Theorem 10 hold without assuming $a\le C_1b$ or fixed k. Moreover, $\widehat{a}_u$ and $\widehat{b}_u$ in

(11) and (22) can be replaced by a and b. Then, the conditions (16) and (18) in Theorem 4 and Theorem 12 can be weakened as $\gamma = o(k^{-1})$ because the we do not need to establish Lemma 16 anymore. Combining Theorem 4, Theorem 6, Theorem 10 and Theorem 12, we obtain the desired results. ■

Appendix D. Proofs of Lemma 20 and Lemma 22

The following lemma is Corollary A.1.10 in Alon and Spencer (2004).

Lemma 25 *For independent Bernoulli random variables $X_u \sim Bern(p_u)$ and $p = \frac{1}{n}\sum_{u\in[n]} p_u$, we have*

$$\mathbb{P}\left(\sum_{u\in[n]}(X_u - p_u) \geq t\right) \leq \exp\left(t - (pn+t)\log\left(1+\frac{t}{pn}\right)\right),$$

for any $t \geq 0$.

The following result is Lemma 3.5 in Chin et al. (2015).

Lemma 26 *Consider any adjacency matrix $A \in \{0,1\}^{n\times n}$ for an undirected graph. Suppose $\max_{u\in[n]}\sum_{v\in[n]} A_{uv} \leq \gamma$ and for any $S, T \subset [n]$, one of the following statements holds with some constant $C > 0$:*

1. $\frac{e(S,T)}{|S||T|\frac{\gamma}{n}} \leq C$,
2. $e(S,T)\log\left(\frac{e(S,T)}{|S||T|\frac{\gamma}{n}}\right) \leq C|T|\log\frac{n}{|T|}$,

where $e(S,T)$ is the number of edges connecting S and T. Then, $\sum_{(u,v)\in H} x_u A_{uv} y_v \leq C'\sqrt{\gamma}$ uniformly over all unit vectors x, y, where $H = \{(u,v) : |x_u y_v| \geq \sqrt{\gamma}/n\}$ and $C' > 0$ is some constant.

The following lemma is critical for proving both theorems.

Lemma 27 *For any $\tau > C(1 + np_{\max})$ with some sufficiently large $C > 0$, we have*

$$|\{u \in [n] : d_u \geq \tau\}| \leq \frac{n}{\tau}$$

with probability at least $1 - e^{-C'n}$ for some constant $C' > 0$.

Proof Let us consider any fixed subset of nodes $S \subset [n]$ such that it has degree at least τ and $|S| = l$ for some $l \in [n]$. Let $e(S)$ be the number of edges in the subgraph S and $e(S, S^c)$ be the number of edges connecting S and S^c. By the requirement on S, either $e(S) \geq C_1 l\tau$ or $e(S, S^c) \geq C_1 l\tau$ for some universal constant $C_1 > 0$. We are going to show that both $\mathbb{P}(e(S) \geq C_1 l\tau)$ and $\mathbb{P}(e(S, S^c) \geq C_1 l\tau)$ are small. Note that $\mathbb{E}e(S) \leq C_2 l^2 p_{\max}$

and $\mathbb{E}e(S, S^c) \leq C_2 lnp_{\max}$ for some universal $C_2 > 0$. Then, when $\tau > C(np_{\max} + 1)$ for some sufficiently large $C > 0$, Lemma 25 implies

$$\mathbb{P}(e(S) \geq C_1 l\tau) \leq \exp\left(-\frac{1}{4}C_1 l\tau\log\left(1+\frac{C_1\tau}{2C_2 lp_{\max}}\right)\right),$$

and

$$\mathbb{P}(e(S, S^c) \geq C_1 l\tau) \leq \exp\left(-\frac{1}{4}C_1 l\tau\log\left(1+\frac{C_1\tau}{2C_2 np_{\max}}\right)\right).$$

Applying union bound, the probability that the number of nodes with degree at least τ is greater than ξn is

$$\begin{aligned}
& \mathbb{P}\Big(|\{u \in [n] : d_u \geq \tau\}| > \xi n\Big) \\
\leq & \sum_{l>\xi n}\mathbb{P}\Big(|\{u \in [n] : d_u \geq \tau\}| = l\Big) \\
\leq & \sum_{l>\xi n}\sum_{|S|=l}\left(\mathbb{P}(e(S) \geq C_1 l\tau) + \mathbb{P}(e(S, S^c) \geq C_1 l\tau)\right) \\
\leq & \sum_{l>\xi n}\exp\left(l\log\frac{en}{l}\right)\left(\exp\left(-\frac{1}{4}C_1 l\tau\log\left(1+\frac{C_1\tau}{2C_2 lp_{\max}}\right)\right)\right. \\
& \left.+\exp\left(-\frac{1}{4}C_1 l\tau\log\left(1+\frac{C_1\tau}{2C_2 np_{\max}}\right)\right)\right) \\
\leq & \sum_{l>\xi n} 2\exp\left(l\log\frac{en}{l} - \frac{1}{4}C_1 l\tau\log\left(1+\frac{C_1\tau}{2C_2 np_{\max}}\right)\right) \\
\leq & \exp(-C'n),
\end{aligned}$$

where the last inequality is by choosing $\xi = \tau^{-1}$. Therefore, with probability at least $1 - e^{-C'n}$, the number of nodes with degree at least τ is bounded by $\tau^{-1}n$. ■

Lemma 28 *Given $\tau > 0$, define the subset $J = \{u \in [n] : d_u \leq \tau\}$. Then for any $C' > 0$, there is some $C > 0$ such that*

$$\|A_{JJ} - P_{JJ}\|_{\text{op}} \leq C\left(\sqrt{np_{\max}} + \sqrt{\tau} + \frac{np_{\max}}{\sqrt{\tau} + \sqrt{np_{\max}}}\right),$$

with probability at least $1 - n^{-C'}$.

Proof The idea of the proof follows the argument in Friedman et al. (1989); Feige and Ofek (2005). By definition,

$$\|A_{JJ} - P_{JJ}\|_{\text{op}} = \sup_{x,y\in S^{n-1}}\sum_{(u,v)\in J\times J} x_u(A_{uv} - P_{uv})y_v.$$

Define $L = \{(u,v) : |x_u y_v| \le (\sqrt{\tau} + \sqrt{p_{\max} n})/n\}$ and $H = \{(u,v) : |x_u y_v| \ge (\sqrt{\tau} + \sqrt{p_{\max} n})/n\}$, then we have

$$\|A_{JJ} - P_{JJ}\|_{\mathrm{op}} \le \sup_{x,y \in S^{n-1}} \sum_{(u,v) \in L \cap J \times J} x_u (A_{uv} - P_{uv}) y_v + \sup_{x,y \in S^{n-1}} \sum_{(u,v) \in H \cap J \times J} x_u (A_{uv} - P_{uv}) y_v.$$

A discretization argument in Chin et al. (2015) implies that

$$\begin{aligned} \sup_{x,y \in S^{n-1}} \sum_{(u,v) \in L \cap J \times J} x_u (A_{uv} - P_{uv}) y_v \lesssim & \max_{x,y \in \mathcal{N}} \max_{S \subset [n]} \sum_{(u,v) \in L \cap S \times S} x_u (A_{uv} - \mathbb{E} A_{uv}) y_v \\ & + \max_{x,y \in \mathcal{N}} \max_{S \subset [n]} \sum_{(u,v) \in L \cap S \times S} x_u (\mathbb{E} A_{uv} - P_{uv}) y_v, \end{aligned}$$

where $\mathcal{N} \subset S^{n-1}$ and $|\mathcal{N}| \le 5^n$. Then, Bernstein's inequality and union bound imply that $\max_{x,y \in \mathcal{N}} \max_{S \subset [n]} \sum_{(u,v) \in L \cap S \times S} x_u (A_{uv} - \mathbb{E} A_{uv}) y_v \le C(\sqrt{\tau} + \sqrt{n p_{\max}})$ with probability at least $1 - e^{-C' n}$. We also have $\max_{x,y \in \mathcal{N}} \max_{S \subset [n]} \sum_{(u,v) \in L \cap S \times S} x_u (\mathbb{E} A_{uv} - P_{uv}) y_v \le \|\mathbb{E} A - P\|_{\mathrm{op}} \le 1$. This completes the first part.

To bound the second part $\sup_{x,y \in S^{n-1}} \sum_{(u,v) \in H \cap J \times J} x_u (A_{uv} - P_{uv}) y_v$, we are going to bound $\sup_{x,y \in S^{n-1}} \sum_{(u,v) \in H \cap J \times J} x_u A_{uv} y_v$ and $\sup_{x,y \in S^{n-1}} \sum_{(u,v) \in H \cap J \times J} x_u P_{uv} y_v$ separately. By the definition of H,

$$\sup_{x,y \in S^{n-1}} \sum_{(u,v) \in H \cap J \times J} x_u P_{uv} y_v = \sup_{x,y \in S^{n-1}} \sum_{(u,v) \in H \cap J \times J} \frac{x_u^2 y_v^2}{|x_u y_v|} P_{uv} \le \frac{n p_{\max}}{\sqrt{\tau} + \sqrt{p_{\max} n}}.$$

To bound $\sup_{x,y \in S^{n-1}} \sum_{(u,v) \in H \cap J \times J} x_u A_{uv} y_v$, it is sufficient to check the conditions of Lemma 26 for the graph A_{JJ}. By definition, its degree is bounded by τ. Following the argument of Lei and Rinaldo (2014), the two conditions of Lemma 26 hold with $\gamma = \tau + n p_{\max}$ with probability at least $1 - n^{-C'}$. Thus, $\sup_{x,y \in S^{n-1}} \sum_{(u,v) \in H \cap J \times J} x_u A_{uv} y_v \le C(\sqrt{\tau} + \sqrt{n p_{\max}})$ with probability at least $1 - n^{-C'}$. Hence, the proof is complete. ■

Proof [Proof of Lemma 20] By triangle inequality,

$$\|T_\tau(A) - P\|_{\mathrm{op}} \le \|T_\tau(A) - T_\tau(P)\|_{\mathrm{op}} + \|T_\tau(P) - P\|_{\mathrm{op}},$$

where $T_\tau(P)$ is the matrix obtained by zeroing out the u^{th} row and column of P with $d_u \ge \tau$. Let $J = \{u \in [n] : d_u \le \tau\}$, and then $\|T_\tau(A) - T_\tau(P)\|_{\mathrm{op}} = \|A_{JJ} - P_{JJ}\|_{\mathrm{op}}$, whose bound has been established in Lemma 28. By Lemma 27, $|J^c| \le n/\tau$ with high probability. This implies $\|T_\tau(P) - P\|_{\mathrm{op}} \le \|T_\tau(P) - P\|_{\mathrm{F}} \le \sqrt{2n|J^c| p_{\max}^2} \le \frac{\sqrt{2} n p_{\max}}{\sqrt{\tau}}$. Taking $\tau \in [C_1(1 + n p_{\max}), C_2(1 + n p_{\max})]$, the proof is complete. ■

Now let us prove Lemma 22. The following lemma, which controls the degree, is Lemma 7.1 in Le et al. (2015).

Lemma 29 *For any $C' > 0$, there exists some $C > 0$ such that with probability at least $1 - n^{-C'}$, there exists a subset $J \subset [n]$ satisfying $n - |J| \le \frac{n}{2e(np_{\max}+1)}$ and*

$$|d_v - \mathbb{E} d_v| \le C\sqrt{(np_{\max} + 1)\log(e(np_{\max} + 1))}, \quad \text{for all } v \in J,$$

where $d_v = \sum_{u \in [n]} A_{uv}$.

Using this lemma, together with Lemma 27 and Lemma 28, we are able to prove the following result, which improves the bound in Theorem 7.2 of Le et al. (2015).

Lemma 30 *For any $C' > 0$, there exists some $C > 0$ such that with probability at least $1 - n^{-C'}$, there exists a subset $J \subset [n]$ satisfying $n - |J| \le n/d$ and*

$$\|(L(A_\tau) - L(P_\tau))_{J \times J}\|_{\mathrm{op}} \le C\left(\frac{\sqrt{d \log d}(d + \tau)}{\tau^2} + \frac{\sqrt{d}}{\tau}\right),$$

where $d = e(np_{\max} + 1)$.

Proof Let us use the notation $d_v = \sum_{u \in [n]} A_{uv}$ in the proof. Define the set $J_1 = \{v \in [n] : d_v \le C_1 d\}$ for some sufficiently large constant $C_1 > 0$. Using Lemma 27 and Lemma 28, with probability at least $1 - n^{-C'}$, we have

$$n - |J_1| \le \frac{n}{2d}, \tag{79}$$

and

$$\|(A - P)_{J_1 J_1}\|_{\mathrm{op}} \le C\sqrt{d}. \tag{80}$$

Let J_2 be the subset in Lemma 29, and then with probability at least $1 - n^{-C'}$, J_2 satisfies

$$n - |J_2| \le \frac{n}{2d}, \tag{81}$$

and

$$|d_v - \mathbb{E} d_v| \le C\sqrt{d \log d}, \quad \text{for all } v \in J_2. \tag{82}$$

Define $J = J_1 \cap J_2$. By (79) and (81), we have

$$n - |J| = |(J_1 \cap J_2)^c| \le |J_1^c| + |J_2^c| = n - |J_1| + n - |J_2| \le \frac{n}{d}, \tag{83}$$

and

$$\|(A - P)_{JJ}\|_{\mathrm{op}} \le \|(A - P)_{J_1 J_1}\|_{\mathrm{op}} \le C\sqrt{d}. \tag{84}$$

Moreover, (82) implies

$$\max_{v \in J} |d_v - \mathbb{E} d_v| \le C\sqrt{d \log d}.$$

Define $\bar{d}_v = \sum_{u \in [n]} P_{uv}$. Then,

$$\max_{v \in J} |d_v - \bar{d}_v| \le \max_{v \in J} |d_v - \mathbb{E} d_v| + 1 \le C\sqrt{d \log d}. \tag{85}$$

Define $D_\tau = \mathrm{diag}(d_1+\tau, ..., d_n+\tau)$ and $\bar{D}_\tau = \mathrm{diag}(\bar{d}_1+\tau, ..., \bar{d}_n+\tau)$. We introduce the notation

$$R = (A_\tau)_{JJ}, \quad B = (D_\tau)_{JJ}^{-1/2}, \quad \bar{R} = (P_\tau)_{JJ}, \quad \bar{B} = (\bar{D}_\tau)_{JJ}^{-1/2}.$$

Using (85), we have

$$\|B - \bar{B}\|_{\mathrm{op}} \le \max_{v\in[n]} \left| \frac{1}{\sqrt{d_v+\tau}} - \frac{1}{\sqrt{\bar{d}_v+\tau}} \right| \le C\frac{\sqrt{d\log d}}{\tau^{3/2}},$$

for some constant $C > 0$. The definitions of B and $\bar{B}$ implies $\|B\|_{\mathrm{op}} \vee \|\bar{B}\|_{\mathrm{op}} \le \frac{1}{\sqrt{\tau}}$. We rewrite the bound (84) as $\|R-\bar{R}\|_{\mathrm{op}} \le C\sqrt{d}$. Since all entries of $\mathbb{E}A_\tau$ is bounded by $(\tau+d)/n$, we have $\|\bar{R}\|_{\mathrm{op}} \le \|\mathbb{E}A_\tau\|_{\mathrm{op}} \le d+\tau$. Therefore, $\|R\|_{\mathrm{op}} \le \|\bar{R}\|_{\mathrm{op}} + \|R-\bar{R}\|_{\mathrm{op}} \le C(d+\tau)$. Finally,

$$\begin{aligned} & \|(L(A_\tau) - L(P_\tau))_{J\times J}\|_{\mathrm{op}} \\ \le\ & \|B\|_{\mathrm{op}}\|R\|_{\mathrm{op}}\|B-\bar{B}\|_{\mathrm{op}} + \|B\|_{\mathrm{op}}\|R-\bar{R}\|_{\mathrm{op}}\|\bar{B}\|_{\mathrm{op}} + \|B-\bar{B}\|_{\mathrm{op}}\|\bar{R}\|_{\mathrm{op}}\|\bar{B}\|_{\mathrm{op}} \\ \le\ & C\left(\frac{\sqrt{d\log d}(d+\tau)}{\tau^2} + \frac{\sqrt{d}}{\tau}\right). \end{aligned}$$

The proof is complete. ■

Proof [Proof of Lemma 22] Recall that $d = np_{\max}+1$. Following the proof of Theorem 8.4 in Le et al. (2015), it can be shown that with probability at least $1-n^{-C'}$, for any $J \subset [n]$ such that $n - |J| \le n/d$,

$$\|L(A_\tau) - L(P_\tau)\|_{\mathrm{op}} \le \|(L(A_\tau) - L(P_\tau))_{JJ}\|_{\mathrm{op}} + C\left(\frac{1}{\sqrt{d}} + \sqrt{\frac{\log d}{\tau}}\right),$$

where the first term on the right side of the inequality above is bounded in Lemma 30 by choosing an appropriate J. Hence, with probability at least $1-2n^{-C'}$,

$$\|L(A_\tau) - L(P_\tau)\|_{\mathrm{op}} \le C\left(\frac{\sqrt{d\log d}(d+\tau)}{\tau^2} + \frac{\sqrt{d}}{\tau}\right) + C\left(\frac{1}{\sqrt{d}} + \sqrt{\frac{\log d}{\tau}}\right).$$

Choosing $\tau \in [C_1(1+np_{\max}), C_2(1+np_{\max})]$, the proof is complete. ■

Acknowledgments

The research of AY. Z. and HH. Z. was supported in part by the NSF grant DMS-1507511. The research of Z.M. was supported in part by the NSF CAREER grant DMS-1352060 and a Sloan Research Fellowship.

References

Emmanuel Abbe and Colin Sandon. Community detection in general stochastic block models: fundamental limits and efficient recovery algorithms. *arXiv preprint arXiv:1503.00609*, 2015a.

Emmanuel Abbe and Colin Sandon. Detection in the stochastic block model with multiple clusters: proof of the achievability conjectures, acyclic bp, and the information-computation gap. *arXiv preprint arXiv:1512.09080*, 2015b.

Emmanuel Abbe and Colin Sandon. Recovering communities in the general stochastic block model without knowing the parameters. In *Advances in Neural Information Processing Systems*, pages 676–684, 2015c.

Emmanuel Abbe, Afonso S Bandeira, and Georgina Hall. Exact recovery in the stochastic block model. *arXiv preprint arXiv:1405.3267*, 2014.

Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. NP-hardness of Euclidean sum-of-squares clustering. *Machine Learning*, 75(2):245–248, 2009.

Noga Alon and Joel H Spencer. *The Probabilistic Method*. John Wiley & Sons, 2004.

Arash A Amini and Elizaveta Levina. On semidefinite relaxations for the block model. *arXiv preprint arXiv:1406.5647*, 2014.

Arash A Amini, Aiyou Chen, Peter J Bickel, and Elizaveta Levina. Pseudo-likelihood methods for community detection in large sparse networks. *The Annals of Statistics*, 41 (4):2097–2122, 2013.

Peter J Bickel and Aiyou Chen. A nonparametric view of network models and Newman–Girvan and other modularities. *Proceedings of the National Academy of Sciences*, 106 (50):21068–21073, 2009.

T Tony Cai and Xiaodong Li. Robust and computationally feasible community detection in the presence of arbitrary outlier nodes. *arXiv preprint arXiv:1404.6000*, 2014.

Yudong Chen and Jiaming Xu. Statistical-computational tradeoffs in planted problems and submatrix localization with a growing number of clusters and submatrices. *arXiv preprint arXiv:1402.1267*, 2014.

Peter Chin, Anup Rao, and Van Vu. Stochastic block model and community detection in the sparse graphs: A spectral algorithm with optimal rate of recovery. *arXiv preprint arXiv:1501.05021*, 2015.

David S Choi, Patrick J Wolfe, and Edoardo M Airoldi. Stochastic blockmodels with a growing number of classes. *Biometrika*, 99(2):273–284, 2012.

Amin Coja-Oghlan. Graph partitioning via adaptive spectral techniques. *Combinatorics, Probability and Computing*, 19(02):227–284, 2010.

Chandler Davis and William Morton Kahan. The rotation of eigenvectors by a perturbation. III. *SIAM Journal on Numerical Analysis*, 7(1):1–46, 1970.

Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106, 2011.

Yash Deshpande, Emmanuel Abbe, and Andrea Montanari. Asymptotic mutual information for the two-groups stochastic block model. *arXiv preprint arXiv:1507.08685*, 2015.

Uriel Feige and Eran Ofek. Spectral techniques applied to sparse random graphs. *Random Structures & Algorithms*, 27(2):251–275, 2005.

Donniell E Fishkind, Daniel L Sussman, Minh Tang, Joshua T Vogelstein, and Carey E Priebe. Consistent adjacency-spectral partitioning for the stochastic block model when the model parameters are unknown. *SIAM Journal on Matrix Analysis and Applications*, 34(1):23–39, 2013.

Joel Friedman, Jeff Kahn, and Endre Szemeredi. On the second eigenvalue of random regular graphs. In *Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing*, pages 587–598. ACM, 1989.

Chao Gao, Zongming Ma, Anderson Y Zhang, and Harrison H Zhou. Community detection in degree-corrected block models. *arXiv preprint arXiv:1607.06993*, 2016.

Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

Anna Goldenberg, Alice X Zheng, Stephen E Fienberg, and Edoardo M Airoldi. A survey of statistical network models. *Foundations and Trends in Machine Learning*, 2(2):129–233, 2010.

Lars Hagen and Andrew B Kahng. New spectral methods for ratio cut partitioning and clustering. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 11(9):1074–1085, 1992.

Bruce Hajek, Yihong Wu, and Jiaming Xu. Achieving exact cluster recovery threshold via semidefinite programming. *arXiv preprint arXiv:1412.6156*, 2014.

Bruce Hajek, Yihong Wu, and Jiaming Xu. Achieving exact cluster recovery threshold via semidefinite programming: Extensions. *arXiv preprint arXiv:1502.07738*, 2015.

Mark S Handcock, Adrian E Raftery, and Jeremy M Tantrum. Model-based clustering for social networks. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 170(2):301–354, 2007.

Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

Roger A Horn and Charles R Johnson. *Matrix Analysis*. Cambridge University Press, 2012.

Jiashun Jin. Fast community detection by score. *The Annals of Statistics*, 43(1):57–89, 2015.

Varun Jog and Po-Ling Loh. Information-theoretic bounds for exact recovery in weighted stochastic block models using the renyi divergence. *arXiv preprint arXiv:1509.06418*, 2015.

Antony Joseph and Bin Yu. Impact of regularization on spectral clustering. *arXiv preprint arXiv:1312.1733*, 2013.

Brian Karrer and Mark EJ Newman. Stochastic blockmodels and community structure in networks. *Physical Review E*, 83(1):016107, 2011.

Amit Kumar, Yogish Sabharwal, and Sandeep Sen. A simple linear time $(1+\varepsilon)$-approximation algorithm for geometric k-means clustering in any dimensions. In *Proceedings-Annual Symposium on Foundations of Computer Science*, pages 454–462. IEEE, 2004.

Can M Le, Elizaveta Levina, and Roman Vershynin. Sparse random graphs: Regularization and concentration of the Laplacian. *arXiv preprint arXiv:1502.03049*, 2015.

Jing Lei and Alessandro Rinaldo. Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43(1):215–237, 2014.

Jing Lei and Lingxue Zhu. A generic sample splitting approach for refined community recovery in stochastic block models. *arXiv preprint arXiv:1411.1469*, 2014.

Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is NP-hard. In *WALCOM: Algorithms and Computation*, pages 274–285. Springer, 2009.

Laurent Massoulié. Community detection thresholds and the weak ramanujan property. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 694–703. ACM, 2014.

Frank McSherry. Spectral partitioning of random graphs. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 529–537. IEEE, 2001.

Elchanan Mossel and Jiaming Xu. Density evolution in the degree-correlated stochastic block model. *arXiv preprint arXiv:1509.03281*, 7, 2015.

Elchanan Mossel, Joe Neeman, and Allan Sly. Stochastic block models and reconstruction. *arXiv preprint arXiv:1202.1499*, 2012.

Elchanan Mossel, Joe Neeman, and Allan Sly. Belief propagation, robust reconstruction, and optimal recovery of block models. *arXiv preprint arXiv:1309.1380*, 2013a.

Elchanan Mossel, Joe Neeman, and Allan Sly. A proof of the block model threshold conjecture. *arXiv preprint arXiv:1311.4115*, 2013b.

Elchanan Mossel, Joe Neeman, and Allan Sly. Consistency thresholds for binary symmetric block models. *arXiv preprint arXiv:1407.1591*, 2014.

Mark EJ Newman and Elizabeth A Leicht. Mixture models and exploratory analysis in networks. *Proceedings of the National Academy of Sciences*, 104(23):9564–9569, 2007.

Tai Qin and Karl Rohe. Regularized spectral clustering under the degree-corrected stochastic blockmodel. In *Advances in Neural Information Processing Systems*, pages 3120–3128, 2013.

Alfred Rényi. On measures of entropy and information. In *Fourth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 547–561, 1961.

Karl Rohe, Sourav Chatterjee, and Bin Yu. Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, 39(4):1878–1915, 2011.

Purnamrita Sarkar and Peter J Bickel. Role of normalization in spectral clustering for stochastic blockmodels. *arXiv preprint arXiv:1310.1495*, 2013.

Daniel L Sussman, Minh Tang, Donniell E Fishkind, and Carey E Priebe. A consistent adjacency spectral embedding for stochastic blockmodel graphs. *Journal of the American Statistical Association*, 107(499):1119–1128, 2012.

Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4): 395–416, 2007.

Ulrike von Luxburg, Mikhail Belkin, and Olivier Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, pages 555–586, 2008.

Van Vu. A simple SVD algorithm for finding hidden partitions. *arXiv preprint arXiv:1404.3918*, 2014.

Stanley Wasserman. *Social Network Analysis: Methods and Applications*, volume 8. Cambridge University Press, 1994.

Se-Young Yun and Alexandre Proutiere. Accurate community detection in the stochastic block model via spectral algorithms. *arXiv preprint arXiv:1412.7335*, 2014a.

Se-Young Yun and Alexandre Proutiere. Community detection via random and adaptive sampling. In *COLT*, pages 138–175, 2014b.

Se-Young Yun and Alexandre Proutiere. Optimal cluster recovery in the labeled stochastic block model. *ArXiv e-prints*, 5, 2015.

Anderson Y Zhang and Harrison H Zhou. Minimax rates of community detection in stochastic block models. *arXiv preprint arXiv:1507.05313*, 2015.

Yunpeng Zhao, Elizaveta Levina, and Ji Zhu. Consistency of community detection in networks under degree-corrected stochastic block models. *The Annals of Statistics*, 40 (4):2266–2292, 2012.

Journal of Machine Learning Research 18 (2017) 1-33 Submitted 10/16; Revised 5/17; Published 7/17

On the Propagation of Low-Rate Measurement Error to Subgraph Counts in Large Networks

Prakash Balachandran PRAKASHB@BU.EDU
Eric D. Kolaczyk KOLACZYK@BU.EDU
Department of Mathematics & Statistics
Boston University
Boston, MA 02215 USA

Weston D. Viles WESTON.D.VILES@DARTMOUTH.EDU
Department of Biomedical Data Science
Dartmouth College
Hanover, NH 03755 USA

Editor: Network Analysis Special Issue

Abstract

Our work in this paper is inspired by a statistical observation that is both elementary and broadly relevant to network analysis in practice—that the uncertainty in approximating some true graph $G = (V, E)$ by some estimated graph $\hat{G} = (V, \hat{E})$ manifests as errors in our knowledge of the presence/absence of edges between vertex pairs, which must necessarily propagate to any estimates of network summaries $\eta(G)$ we seek. Motivated by the common practice of using plug-in estimates $\eta(\hat{G})$ as proxies for $\eta(G)$, our focus is on the problem of characterizing the distribution of the discrepancy $D = \eta(\hat{G}) - \eta(G)$, in the case where $\eta(\cdot)$ is a subgraph count. Specifically, we study the fundamental case where the statistic of interest is $|E|$, the number of edges in G. Our primary contribution in this paper is to show that in the empirically relevant setting of large graphs with low-rate measurement errors, the distribution of $D_E = |\hat{E}| - |E|$ is well-characterized by a Skellam distribution, when the errors are independent or weakly dependent. Under an assumption of independent errors, we are able to further show conditions under which this characterization is strictly better than that of an appropriate normal distribution. These results derive from our formulation of a general result, quantifying the accuracy with which the difference of two sums of dependent Bernoulli random variables may be approximated by the difference of two independent Poisson random variables, i.e., by a Skellam distribution. This general result is developed through the use of Stein's method, and may be of some general interest. We finish with a discussion of possible extension of our work to subgraph counts $\eta(G)$ of higher order.

Keywords: Limit distribution, network analysis, Skellam distribution, Stein's method.

1. Introduction

The analysis of network data is widespread across the scientific disciplines. Technological and infrastructure, social, biological, and information networks are a few of the major network classes in which such analyses have been employed. However, despite the already substantial body of work in network analysis generally (e.g., see Jackson, 2010; Kolaczyk, 2009; Newman, 2010), with contributions from a variety of different fields of study, much work still remains to more fully develop the theory and methods of statistical analysis of network data, particularly for certain classes of problems of a fairly fundamental nature. Here in this paper we pose and address a version of one such fundamental problem, that regarding the propagation of error through the process of network construction and summary.

In applied network analysis, a common *modus operandi* is to (i) gather basic measurements relevant to the interactions among elements in a system of interest, (ii) construct a graph-based representation of that system, with nodes serving as elements and links indicating interactions between pairs of elements, and (iii) summarize the structure of the resulting graph using a variety of numerical and visual tools. See Kolaczyk (2009, Chs 3 & 4) for background and several case studies illustrating this process. Key here is the point that the process of network analysis usually rests upon some collection of measurements of a more basic nature. For example, online social networks (e.g., Facebook) are based on the extraction and merging of lists of 'friends' from millions of individual accounts (Hansen et al., 2010, Ch 11.8). Similarly, biological networks (e.g., of gene regulatory relationships) are often based on notions of association (e.g., correlation, partial correlation, etc.) among experimental measurements of gene activity levels (Fogelberg and Palade, 2009). Finally, maps of the logical Internet traditionally have been synthesized from the results of surveys in which paths along which information flows are learned through a large set of packet probes (e.g., Cheswick et al., 2000).

Importantly, while it is widely recognized that there is measurement error associated with these and other common types of network constructions, most applied network analyses in practice effectively proceed as if there were in fact no error. There are at least two possible reasons for this current state of affairs: (1) there is comparatively little in the way of formal probabilistic analyses characterizing the propagation of such error and of statistical methods accounting for such propagation, and (2) in many settings (arguably due at least in part to (1)), much attention is given at the stages of measurement and network construction to trying to keep the rate of error 'low' in declaring the presence and absence of links between nodes.

Here we offer a formal treatment of the problem of propagation of error. We provide a general framework within which it is possible to characterize the manner in which errors made in assigning links between nodes accumulate in the reporting of certain functions of the network as a whole. Our treatment is probabilistic, wherein our goal is to understand the nature of the distribution induced on the graph functions by that of the errors in the graph construction. We anticipate that this contribution will provide part of a critical foundation upon which, in turn, it should be possible to develop a statistical treatment, e.g., the construction of appropriate confidence intervals for graph-based quantities of interest.

More formally, we consider a setting wherein an underlying (undirected) network-graph G possesses a network characteristic $\eta(G)$ of interest. While there are many types of functions $\eta(\cdot)$ used in practice to characterize networks (e.g., centralities, path-based metrics, output from methods of community detection, etc.) we restrict our attention here to the canonical problem of subgraph counting. That is, we are interested in the class of functions η of the form

$$\eta_H(G) = \frac{1}{|Iso(H)|} \sum_{H' \subseteq K_{n_v}, H' \cong H} 1\{H' \subseteq G\} \ , \tag{1}$$

where $n_v = |V(G)|$ is the number of vertices in G, K_{n_v} is the complete graph on n_v vertices, H is a graph of interest (i.e., copies of which we desire to count), and $H \subseteq G$ indicates that H is a subgraph of G (i.e., $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$). The value $|Iso(H)|$ is a normalization

©2017 Prakash Balachandran, Eric D. Kolaczyk, and Weston D. Viles.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-497.html.

factor for the number of isomorphisms of H. We will concentrate primarily on the fundamental case where $\eta(G) = |E|$, i.e., the number of edges in G.

If $\hat{G}$ is a network-graph resulting from an attempt to construct G from some collection of basic measurements, then the common practice of reporting the analogous characteristics of $\hat{G}$ is equivalent to the use of $\eta\left(\hat{G}\right)$, i.e., effectively a plug in estimator. Let the discrepancy between these two quantities be defined as $D = \eta\left(\hat{G}\right) - \eta\left(G\right)$, which in the case of counting edges reduces to $D_E = |\hat{E}| - |E|$. Our goal is to make precise probabilistic statements about the behavior of D under certain conditions.

Importantly, in the case where η is defined as a subgraph count, as in (1), D may be expressed as the difference of (i) the number of times the subgraph H arises somewhere in $\hat{G}$ but does *not* in fact exist in the same manner in G, and (ii) vice versa. Hence, D may be understood in this context to be the difference in total number of Type I and Type II errors, respectively. Intuitively, in the cases where a sufficiently low rate of error occurs on a large graph G, each of these two sums should have a Poisson-like behavior. This observation suggests that the propagation of low-rate measurement error to subgraph counts should behave, under appropriate conditions, like the difference of two Poisson random variables, i.e., a so-called Skellam distribution (Skellam, 1946).

Our contribution in this paper is to provide an initial set of results on the accuracy with which the Skellam distribution may be used in approximating the distribution of D, under the setting where the graph G is large and the rate of error is low. We consider the cases of both sparse and dense networks. The primary technical device used here is Stein's method (e.g, Barbour and Chen, 2005). Specifically, we present a Stein operator for the Skellam probability distribution and, in a manner consistent with the Stein methodology, we derive several bounds on the discrepancy between the distribution of the difference of two arbitrary sums of binary random variables to an appropriately parameterized Skellam distribution. The latter in turn is then used to establish in particular the rate of weak convergence of D_E to an appropriate Skellam random variable, under either independent or weakly dependent measurement errors.

As remarked above, there appears to be little in the way of a formal and general treatment of the error propagation problem we consider here. However, there are, of course, several areas in which the probabilistic or statistical treatment of uncertainty enters prominently in network analysis. The closest area to what we present here is the extensive literature on distributions of subgraph counts in random graphs. See Janson et al. (2011), for example, for comprehensive coverage. Importantly, there the graph G is assumed to emerge according to a (classical) random graph and uncertainty typically is large enough that normal limit theorems are the norm (although Poisson limit theorems also have been established). In contrast, in our setting we assume that G is a fixed, true underlying graph, and then study the implications of observing a 'noisy' version $\hat{G}$ of that graph, under various assumptions on the nature of the noise, which involves two specific types of error (i.e., Type I and II errors), the contributions of which are informed in part by the topology of G itself. An area related to this work in random graphs is the work in statistical models for networks, such as exponential random graph models (ERGMs). See Lusher et al. (2012) for a recent treatment. Here, while these models are inherently statistical in nature, the randomness due to generation of the graph G and due to observation of G—resulting in what we call $\hat{G}$—usually are combined into a single realization from the underlying distribution. And while subgraph counts do play a key role in traditional ERGMs, they typically enter as covariates in these (auto)regressive models. In a somewhat different direction, uncertainty in network construction due to sampling has also been

studied in some depth. See, for example, Kolaczyk (2009, Ch 5) or Ahmed et al. (2014) for surveys of this area. However, in this setting, the uncertainty arises only from sampling—the subset of vertices and edges obtained through sampling are typically assumed to be observed without error. Finally, we note that there just recently have started to emerge in the statistics literature formal treatments of the same type of graph observation error model that we propose here. There the emphasis is on producing estimators of network model parameters and/or classifiers (e.g., Priebe et al., 2015), for example, rather than on the type of basic network summary statistics that are the focus here.

The organization of this paper is as follows. In Section 2 we provide necessary background. In Section 3 we then provide a general set of results useful for our general problem. Specifically, we establish a bound for the Kolmogorov-Smirnov distance between the distribution of the difference of two arbitrary sums of binary random variables from a certain Skellam. This work is based on the application of Stein's method to the Skellam distribution, a first of its kind to the best of our knowledge, and the results therefore are of some independent interest as well. In Section 4 we then illustrate the way in which these general results may be used to understand the propagation of error in networks for counting edges. In doing so, several other general results are established. Some implications of these results on the problem of counting subgraphs of higher order are noted in Section 5, along with other related discussion. Proofs of our key results may be found in the appendices.

2. Background

In this section, we provide essential notation and background for our forthcoming general results.

2.1 Notation and Assumptions

By $G = (V, E)$ we will mean an undirected graph, with vertex set V of cardinality $|V|$ and edge set E of cardinality $|E|$. Vertex pairs in the complement of E, i.e., in E^c, will be referred to as non-edges. Much of the results that follow will be stated as a function of the number of vertices which, for notational convenience, we denote $n_v = |V|$. Let $\mu = \mu(G) = 2|E|/n_v$ correspond to the average degree of a vertex in G. We assume the vertex set V is known but that the edge set E is unknown. However, we assume there is information by which to construct an approximation to E or, more formally, by which to infer E, as a set $\hat{E}$, yielding an inferred graph $\hat{G} = (V, \hat{E})$.

While there are many ways in practice by which the set $\hat{E}$ is obtained, one principled way of viewing the task is as one of performing $\binom{n_v}{2}$ hypothesis tests, using the data underlying the graph construction process as input, one for each of the vertex pairs $\{i, j\}$ in the graph G. In some contexts, $\hat{G}$ is literally obtained through hypothesis testing procedures; for instance, in constructing some gene regulatory networks from microarray expression data. See Kolaczyk (2009, Ch 7), for example. Formally, in such cases we can think of $\hat{G}$ as resulting from a collection of testing problems

$$H_0 : \{i, j\} \notin E \text{ versus } H_1 : \{i, j\} \in E \ ,$$

for $\{i, j\} \in V^{(2)}$, where

$$V^{(2)} = \{\{i, j\} : i, j \in V; i < j\} \ .$$

These tests amount to a collection of $\binom{n_v}{2}$ binary random variables $\left\{Y_{ij} : \{i,j\} \in V^{(2)}\right\}$, where

$$Y_{ij} = \begin{cases} 1 & \text{if } H_0 \text{ is rejected} \\ 0 & \text{if } H_0 \text{ is not rejected.} \end{cases}$$

Note that the random variables Y_{ij} need not be independent and, in fact, in many contexts will most likely be dependent. Gene regulatory networks inferred by correlating gene expression values at each vertex i with that of all other vertices $j \in V \setminus \{i\}$ and maps of the logical Internet obtained through merging paths learned by sending traffic probes between many sources and destinations are just two examples where dependency can be expected.

Whether obtained informally or formally, we can define the inferred edge set $\hat{E}$ as

$$\hat{E} = \left\{\{i,j\} \in V^{(2)} : Y_{ij} = 1\right\} \ .$$

It is useful to think of the collection of random variables $\left\{Y_{ij} : \{i,j\} \in V^{(2)}\right\}$ as being associated with two types of errors. That is, we express the marginal distributions of the Y_{ij} in the form

$$Y_{ij} \sim \begin{cases} \text{Bernoulli}\left(\alpha_{ij}\right), \text{if } \{i,j\} \in E^c, \\ \text{Bernoulli}\left(1-\beta_{ij}\right), \text{if } \{i,j\} \in E, \end{cases} \tag{2}$$

where $E^c = V^{(2)} \setminus E$. Again pursuing the example of network construction based on hypothesis testing, α_{ij} can be interpreted as the probability of Type-I error for the test on vertex pair $\{i,j\} \in E^c$, while β_{ij} is interpreted as the probability of Type-II error for the test on vertex pair $\{i,j\} \in E$. For example, α_{ij} might be the probability of incorrectly inferring a regulatory relationship between two genes i and j, based on thresholding the empirical correlation between their expression values, as quantified through deep sequencing measurements over a series of experiments. Similarly, β_{ij} might be the probability of missing a logical link in the Internet topology between two routers i and j, due to artifacts in measurement technologies (e.g., traceroute).

Our interest in this paper is in characterizing the manner in which the uncertainty in the Y_{ij} propagates to subgraph counts on $\hat{G}$. More specifically, we seek to characterize the distribution of the difference

$$D = \frac{1}{|Iso(H)|} \sum_{H' \subseteq K_{n_v}, H' \cong H} \left[1\{H' \subseteq \hat{G}\} - 1\{H' \subseteq G\}\right] \ , \tag{3}$$

for a given choice of subgraph H. Naturally, this distribution will depend in no small part on context. Here we focus on a general formulation of the problem in which we make the following three assumptions.

(A1) *Large Graphs:* $n_v \to \infty$.

(A2) *Edge Unbiasedness:* $\sum_{\{i,j\} \in E^c} \alpha_{ij} = \sum_{\{i,j\} \in E} \beta_{ij} \ (\equiv \lambda)$.

(A3) *Low Error Rate:* $\lambda = \Theta\left(\mu\right)$.

Assumption (A1) reflects both the fact that the study of large graphs is a hallmark of modern applied work in complex networks and, accordingly, our desire to make statements that are asymptotic in n_v.

Our use of assumption (A2) reflects the understanding that a 'good' approximation $\hat{G}$ to the graph G should at the very least have roughly the right number of edges. The difference of the two sums defined in (A2) is in fact the expectation of the statistic D in (3) for the case where the subgraph being counted is just a single edge, i.e., it is the expected discrepancy between the number of observed edges $|\hat{E}|$ and the actual number of edges $|E|$. So (A2) states that this particular choice of D has expectation zero. Alternatively, (A2) may be interpreted as saying that the total numbers of Type I and Type II errors should be equal to a common value λ.

Finally, we capture the notion of a 'low' rate of error in $\hat{G}$ through assumption (A3). Specifically, we assume that the number of Type I or Type II errors in edge status that we expect throughout the network is roughly on par with the average number of edges incident to any given vertex in the network. This condition can be re-expressed in a useful manner with respect to n_v if, as is common in the literature, we distinguish between sparse and dense graphs. By the term *sparse* we will mean a graph for which $|E| = \Theta\left(n_v \log n_v\right)$, and by *dense*, $|E| = \Theta\left(n_v^2\right)$. Hence, assumption (A3) reduces to $\lambda = \Theta\left(\log n_v\right)$ in the case of sparse graphs, and to $\lambda = \Theta(n_v)$, in the case of dense graphs.

In addition, for convenience, we add to the core assumptions (A1)-(A3) a fourth assumption, upon which we will call periodically throughout the paper when desiring to simplify some of our expressions.

(A4) *Homogeneity:* $\alpha_{ij} \equiv \alpha$ and $\beta_{ij} \equiv \beta$, for $\alpha, \beta \in (0,1)$.

In other words, we assume that the probability of making a Type I or II error (as the case may be) does not depend upon the specific (non)edge in question.

Lastly, for completeness, we recall the definition of the Skellam distribution. A random variable W defined on the integers is said to have a Skellam distribution, with parameters $\lambda_1, \lambda_2 > 0$, i.e., $W \sim \text{Skellam}\left(\lambda_1, \lambda_2\right)$, if

$$\mathbb{P}\left(W = k\right) = e^{-(\lambda_1+\lambda_2)} \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k\left(2\sqrt{\lambda_1\lambda_2}\right) \text{ for } k \in \mathbb{Z}, \tag{4}$$

where $I_k\left(2\sqrt{\lambda_1\lambda_2}\right)$ is the modified Bessel function of the first kind with index k and argument $2\sqrt{\lambda_1\lambda_2}$. The Skellam distribution may be constructed by defining W through the difference of two independent Poisson random variables, with means λ_1 and λ_2, respectively. The mean and variance of this distribution are given by $\mathbb{E}[W] = \lambda_1 - \lambda_2$ and $\text{Var}(W) = \lambda_1 + \lambda_2$. The distribution of W is in general asymmetric, with symmetry holding if and only if $\lambda_1 = \lambda_2$.

The main results we provide in this paper are in the form of bounds on the extent to which the distribution of random variables like the discrepancy D in (3) may be well-approximated by an appropriate Skellam distribution. For this purpose, we adopt the Kolmogorov-Smirnov distance to quantify the distance between distributions of two random variables, say, X_1 and X_2, i.e.,

$$ds_{KS}(X_1, X_2) \equiv \sup_{x \in \mathbb{R}} \left|\mathbb{P}\left(X_1 \leq x\right) - \mathbb{P}\left(X_2 \leq x\right)\right| .$$

2.2 Counting Edges

Generic subgraph counts, and the corresponding noise in obtaining them, can be quite varied in real applications. Accordingly, most of our specific results pertain to the fundamental case of counting

edges. That is, where the choice of subgraph H is simply a single edge, and therefore the function $\eta(G)$ in (1) is just the total number of edges in G, i.e., $\eta(G) = |E|$. We will consider two scenarios for this case, wherein the random variables Y_{ij} are independent or weakly dependent.

In the case where the edge noise is independent, the discrepancy

$$\begin{aligned} D_E &= |\hat{E}| - |E| \\ &= \sum_{\{i,j\} \in E^c} Y_{ij} - \sum_{\{i,j\} \in E} (1 - Y_{ij}) \end{aligned}$$

has expectation $\mathbb{E}[D_E] = \alpha|E^c| - \beta|E| = \lambda - \lambda = 0$ and variance $\sigma^2 = \alpha(1-\alpha)|E^c| + \beta(1-\beta)|E|$, and its behavior can be established using existing methods from the literature (i.e., essentially, Chen-Stein methods). However, we include it as an important base case, comparing results obtainable by our methods to those obtainable by more traditional techniques, in Section 4.1.

Alternatively, suppose that the variables Y_{ij} are *dependent*. The random variable D_E again has expectation zero, although its variance—and hence its asymptotic behavior—will differ from the independent case, in a manner dictated by the nature of the underlying dependency in the noise. It often can be expected in practice that the error associated with construction of the empirical graph $\hat{G}$ will involve dependency across (non)edges. For example, relations in gene regulatory networks are often declared based on sufficiently strong levels of association between gene-specific measurements (e.g., measures of gene expression). The comparison of the measurements for each gene with those of all of the others necessarily induces potential dependencies among the random variables Y_{ij}. However, a precise characterization of such dependency is typically problem-specific and, more often than not, nontrivial in nature. In Section 4.2 we will assume general dependency conditions in the spirit of traditional monotone coupling arguments, which will allow for further analysis and interpretation.

3. General Results on Approximation by Skellam

Recall the general form of our statistic of interest D in (3), as the difference of two sums of binary random variables. Under appropriate conditions it seems reasonable to expect that the distribution of D be well-approximated by a Skellam distribution. And for the simplest case, in which we are counting edges under independent noise, it is possible to show that this is in fact the case, through manipulation of existing results for approximating sums by Poisson distributions. Without independence, however, it is necessary to approach the problem directly, by explicitly using the Skellam distribution. In this section, we therefore provide the results of such an approach. This is a completely general treatment—devoid of the motivating context of counting subgraphs—and therefore also likely of some independent interest. In Section 4 we return to the problem of counting subgraphs under low-rate error and illustrate the use of the results presented here in this section through application to the case of counting edges.

Our approach in this section is through Stein's method. This choice is reminiscent, naturally, of the Chen-Stein treatment for Poisson approximations. However, the task is technically more involved at several points, as it requires handling a Stein function that is defined through a second-order difference, rather than the first-order difference encountered in the Poisson problem. Moreover, the kernel of the Skellam distribution includes a modified Bessel function of the first kind, which emerges in ways necessitating a somewhat delicate treatment.

3.1 A Stein Bound for the Skellam Distribution

Let U be a random variable defined as

$$U = \sum_{k=1}^{n} L_k - \sum_{k=1}^{m} M_k \ , \tag{5}$$

where $\{\{L_k\}_{k=1}^n, \{M_k\}_{k=1}^m\}$ is a collection of two sets of indicator random variables with $\mathbb{E}[L_k] = p_k$ for $k = 1, \ldots, n$ and $\mathbb{E}[M_k] = q_k$ for $k = 1, \ldots, m$. In the case of our subgraph counting problem, $U = D$, where D is defined in (3), although for the remainder of this current section U is defined generally.

Recall the definition of a Skellam random variable W in (4). We desire a bound on

$$d_{KS}(U, W) := \sup_{x \in \mathbb{R}} |\mathbb{P}(U \leq x) - \mathbb{P}(W \leq x)| \ , \tag{6}$$

quantifying how close the distribution of U is to that of W. In pursuing the standard paradigm for Stein's method, we first determine an operator $\mathcal{A}[f(k)]$ such that

$$\mathbb{E}\mathcal{A}[f(W)] = 0 \quad \text{if and only if} \quad W \sim \text{Skellam}(\lambda_1, \lambda_2)$$

for any bounded function $f : \mathbb{Z} \mapsto \mathbb{R}$. This operator need not be unique, but the theory only requires one. This is accomplished through the following result, the proof of which uses several properties of the modified Bessel function of the first kind, as detailed in the appendix, in Section A.1.

Theorem 1 *A Stein operator $\mathcal{A}$ for the Skellam (λ_1, λ_2) distribution is*

$$\mathcal{A}[f(k)] = \lambda_1 f(k+1) - k f(k) - \lambda_2 f(k-1) .$$

With this operator in hand, and again following the usual paradigm under Stein's method, we set

$$\mathcal{A}[f(k)] = g(k)$$

for a class of test functions $g(k)$, and allow that to implicitly define the function f. The choice of the test functions g is guided by the choice of the metric used to measure the distance between U and W. Since the metric we choose to measure the distance between U and W is given by $d_{KS}(U, W)$ in (6), we choose the test function $g := g_x$ given by

$$g_x(k) = 1\{k \leq x\} - \mathbb{P}(W \leq x) \tag{7}$$

for any $x \in \mathbb{R}$.

At this point it is common to exhibit a solution f defined by our choice of g. Instead, we forestall that step until later in this section, choosing rather to state a general result that will allow us to more quickly gain insight into the nature of the bounds we are able to obtain. Our result employs a minor variant of the notion of coupling that is common to the literature on Chen-Stein approximations.

Theorem 2 *Let U be as in (5) and let $\mathcal{L}(U)$ denote the law of U. Let*

$$\mathcal{L}\left(U_k^{(L)} + 1\right) = \mathcal{L}(U | L_k = 1) \quad \text{for} \quad k = 1, \ldots, n$$

and

$$\mathcal{L}\left(U_k^{(M)} - 1\right) = \mathcal{L}\left(U | M_k = 1\right), \quad \text{for} \quad k = 1 \ldots, m$$

be a collection of random variables all defined on a common probability space. If $\lambda_1 = \sum_{k=1}^n p_k$ *and* $\lambda_2 = \sum_{k=1}^m q_k$*, and* $W \sim Skellam(\lambda_1, \lambda_2)$*, then*

$$d_{KS}(U, W) \leq ||\Delta f|| \left\{ \sum_{k=1}^n p_k \mathbb{E}\left|U - U_k^{(L)}\right| + \sum_{k=1}^m q_k \mathbb{E}\left|U - U_k^{(M)}\right| \right\} \quad , \tag{8}$$

where

$$||\Delta f|| = \sup_{x \in \mathbb{R}} \sup_{j \in \mathbb{Z}} \left| f_x\left(j+1\right) - f_x\left(j\right) \right|$$

and f_x *is a solution to* $\mathcal{A}\left[f_x(k)\right] = g_x(k)$ *for* $k \in \mathbb{Z}$*.*

The proof of this result relies on elementary considerations of the equation $\mathcal{A}[f_x(k)] = g_x(k)$ and may be found in the appendix in Section A.2. The extent to which it allows one to obtain error estimates of practical interest in a particular setting will depend on the extent to which both the main expression within brackets in (8) and the preceding constant $||\Delta f||$ can be further controlled. While control of the former is problem dependent, control of the latter is not, and may be dealt with separately, as we do next. Afterwards, in Section 4, we illustrate the control of the bracketed expression in (8), in the context of the problem of counting edges introduced in Section 2.2.

3.2 Controlling the Constant Term

Controlling $||\Delta f||$ in (8) first requires understanding the solution $f_x(k)$. We provide a family of closed-form expressions for this solution in the following.

Theorem 3 *Let* g_k *be defined as in equation (7). If* f_x *is a bounded solution to the difference equation*

$$\lambda_1 f_x\left(k+1\right) - k f_x\left(k\right) - \lambda_2 f_x\left(k-1\right) = g_x\left(k\right)$$

for $k \in \mathbb{Z}$*, then* f_x *is given by*

$$f_x\left(m\right) = \begin{cases} (-1)^m \left(\sqrt{\frac{\lambda_2}{\lambda_1}}\right)^m I_m\left[(-1)^c \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^c \frac{1}{I_c} f\left(c\right) \right. \\ \left. + \frac{e^{\lambda_1+\lambda_2}}{\sqrt{\lambda_1\lambda_2}} \sum_{n=c}^{m-1} \frac{(-1)^{n+1}}{I_n I_{n+1}} \mathbb{P}\left(W \leq \min\left\{n, x\right\}\right) \mathbb{P}\left(W > \max\left\{n, x\right\}\right) \right] & \text{if } m > c \\ (-1)^m \left(\sqrt{\frac{\lambda_2}{\lambda_1}}\right)^m I_m\left[(-1)^c \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^c \frac{1}{I_c} f\left(c\right) \right. \\ \left. - \frac{e^{\lambda_1+\lambda_2}}{\sqrt{\lambda_1\lambda_2}} \sum_{n=m}^{c-1} \frac{(-1)^{n+1}}{I_n I_{n+1}} \mathbb{P}\left(W \leq \min\left\{n, x\right\}\right) \mathbb{P}\left(W > \max\left\{n, x\right\}\right) \right] & \text{if } m < c. \end{cases}$$

for any initial condition $(c, f_x\left(c\right))$ *with* $c \in \mathbb{Z}$ *and* $f_x\left(c\right) \in \mathbb{R}$*. Here* I_m *is used as short-hand for* $I_m(2\sqrt{\lambda_1\lambda_2})$*.*

The proof of this theorem is similar to that of solving a second order linear differential equation. An integrating factor is found, integration is performed with a boundary condition at $-\infty$, and then a second integration is performed with the initial condition. Details are provided in the appendix in Section A.3.

Leveraging our insight into f_x to control $||\Delta f||$ means producing a bound on the absolute differences $|\Delta f_x(j)| = |f_x(j+1) - f_x(j)|$ independent of $x \in \mathbb{R}$ and $j \in \mathbb{Z}$. Consider first the special case where λ_1 and λ_2 are equal, for which we are able to offer the following result.

Theorem 4 *Suppose that* $\lambda_1 = \lambda_2 \equiv \lambda$*, so that* $\mathbb{E}[U] = 0$ *and* W *is a Skellam*(λ, λ) *distribution, symmetric about zero. Assume* $\lambda \geq 1$*. Then*

$$||\Delta f|| \leq \frac{160}{2\lambda}.$$

The proof of this theorem is highly technical in nature, and relies on a concentration inequality for the Skellam(λ, λ) distribution (Balachandran et al., 2013b) with several other technical arguments. A sketch of the proof may be found in the appendix in Section A.4, while a detailed presentation is available in the online appendix.

Note that the bound in Theorem 4 is essentially the analogue of the classical result for Poisson approximation, in which, for sufficiently large λ, the term $1/\lambda$ is the standard factor. In both cases, therefore, the corresponding term $||\Delta f||$ is bounded by the inverse of the expected total number of counts, where here that is $\mathbb{E}[T_1 + T_2] = 2\lambda$.

The above result is of immediate relevance to the problem of counting edges, given assumption (A2), whether under the assumption that the edge noise is independent or dependent. We will make use of this result in the next section. For applications involving higher-order subgraphs, we can expect to need an extension of Theorem 4 to the general case of $\lambda_1 \neq \lambda_2$. For arbitrary $\lambda_1, \lambda_2 > 0$, we are unable to produce a satisfactory bound. However, supported by preliminary numerical work, we have the following conjecture.

Conjecture 5 *In general, for* λ_1 *and* λ_2 *sufficiently close and large,*

$$||\Delta f|| \leq \frac{C}{\lambda_1 + \lambda_2},$$

for some constant $C > 0$ *independent of* λ_1, λ_2*.*

4. Application of General Results to Counting Edges

We now illustrate the use of our general results for the problem of characterizing the propagation of low-rate measurement error to subgraph counts in large graphs, for the specific case of counting edges.

4.1 Edge Counts Under Independent Edge Noise

Recall the problem wherein the function of interest (1) counts the number of edges in G, i.e., $\eta(G) = |E|$, and the variables Y_{ij} in (2) are independent. In light of Theorems 2 and 4, we have the following result characterizing the behavior of the discrepancy D in (3), which here is simply $D_E = |\hat{E}| - |E|$.

Theorem 6 *Under assumptions (A1)-(A4) and independence among errors in declaring (non)edge status (i.e., among the* Y_{ij}*),*

$$d_{KS}\left(\, D_E \,,\, Skellam(\lambda, \lambda)\,\right) \leq O\left(n_v^{-1}\right) \quad , \tag{9}$$

for both sparse and dense graphs G*.*

Proof of this result may be found in the appendix, in Section A.5. The theorem establishes a rate at which—in large networks, whether sparse or dense, with independent and homogeneous low-rate errors—the distribution of the discrepancy D_E tends to that of an appropriate Skellam distribution, i.e., symmetric and centered on zero, with variance 2λ. The same rate may be established using more standard arguments from Chen-Stein theory, the proof of which we also include in the appendix, for completeness. These latter arguments, of course, only hold in the case of independence assumed here, and do not extend generally to the case of dependence in the edge noise.

To put the rate established in the above theorem in better context, it is interesting to compare to the case where a normal distribution is used instead to approximate that of the discrepancy D_E. The following theorem, proof of which also may be found in Section A.5, provides both upper and lower bounds.

Theorem 7 *Let $\sigma^2 = Var(D_E)$. Under the same conditions as Theorem 6 , in the case of sparse graphs*

$$\Omega\left(\log^{-1} n_v\right) \leq d_{KS}\left(\, D_E/\sigma \,,\, N(0,1)\,\right)\right) \leq O\left(\log^{-1/2} n_v\right) \quad , \tag{10}$$

while in the case of dense graphs,

$$\Omega\left(n_v^{-1}\right) \leq d_{KS}\left(\, D_E/\sigma \,,\, N(0,1)\,\right)\right) \leq O\left(n_v^{-1/2}\right) \quad , \tag{11}$$

where $N(0,1)$ refers to a standard normal random variable.

These two theorems together indicate that in this context a Skellam approximation is clearly superior to a normal for sparse graphs, and they suggest that it can be better as well for dense graphs. These statements are supported by the results of a small simulation study, shown in Figure 1. There we compared the two approximations as n_v ranges from 100 to 1000 to 10,000, for error rates λ defined to be constant, logarithmic, square root, or linear functions of n_v. For the sparse and dense cases, we let $|E|$ equal $n_v \log n_v$ and $n_v(n_v-1)/4$, respectively. Looking at the sparse case, for when $\lambda = \log n_v$, the Skellam approximation clearly dominates the normal. However, interestingly, this dominance continues even when the error rate is set equal to $n_v^{1/2}$. Only once the error rate is n_v do we see the normal approximation begin to overtake the Skellam approximation. Note that by this stage, $\beta = O(1)$, and so essentially there is no 'signal' standing out from the 'noise'. Similarly, looking at the dense case, we see that the Skellam approximation is better than the normal approximation at all error rates, including, in particular, when the error rate equals n_v, the case addressed by the above two theorems.

In summary, in the independent case, the Skellam distribution dominates the normal as an approximation when there can be expected to be a clear graph 'signal' standing out against the 'noise' induced by underlying low-rate measurement errors.

4.2 Edge Counts Under Dependent Edge Noise

Again, as just above, consider the context wherein counting edges is of interest, so that $\eta(G) = |E|$ and our goal is to characterize the accuracy with which $D_E = |\hat{E}| - |E|$ is approximated by a Skellam(λ, λ) random variable. Now, however, we assume that the error associated with construction of the empirical graph $\hat{G}$ will involve dependency across (non)edges. That is, the random variables Y_{ij} are now dependent. A precise characterization of such dependency is typically problem-specific and, more often than not, nontrivial in nature. Here, for the purposes of illustration, we instead provide certain results of a general nature, working from the bound (8) of Theorem 2.

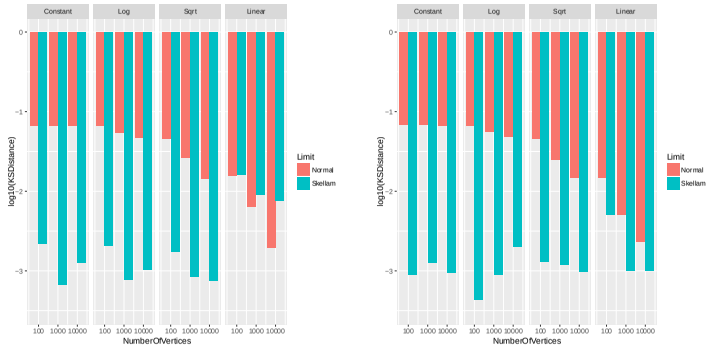


Figure 1: (Log)Kolmogorov-Smirnov distance between Skellam and standard normal approximations to the distribution of discrepancy D_E in edge counts under independent errors. *Left:* Sparse case. *Right:* Dense case.

Of the two terms in (8), the first term $||\Delta f||$ is again known to behave as $O(\lambda^{-1})$, by Theorem 4. On the other hand, control of the second term, in brackets, requires some care. For example, naive inter-change of absolute values and summations with expectation yields that

$$p_k \mathbb{E}\left|U - U_k^{(L)}\right| \leq p_k^2 + \sum_{k \neq j} p_k p_j + E[L_k L_j] + \sum_{\ell=1}^{m} p_k q_\ell + E[L_k M_\ell] \quad ,$$

and similarly for $q_k \mathbb{E}\left|U - U_k^{(M)}\right|$. Unfortunately, it is straightforward to show that for the dependent error version of the problem considered in the previous subsection (i.e., involving independent and homogeneous low-rate errors on large-spare networks) the bound we obtain for $d_{KS}\left(\, D_E \,,\, \text{Skellam}(\lambda,\lambda)\,\right)$ will be no better than $O(\lambda)$—regardless of the nature of the dependency among the Y_{ij}.

One possible approach to a more subtle handling of these terms is motivated by considerations of hypothesis testing. Suppose that the L_k correspond to indicators of Type I error for n tests under their corresponding null hypotheses, and the M_k, to indicators of Type II error for m tests under their corresponding alternative hypotheses. Furthermore, suppose that the corresponding test statistics are all defined on the same scale and compared to the same threshold. Moreover, for simplicity, we assume these statistics all have non-negative values and that their distribution under the null sits to the left of that under the alternative, so that more extreme positive values tend to support the alternative. In this setting, if we know, for example, that $L_1 = 1$, we know that at least one rejection of a null hypothesis has occurred, indicating that the threshold sits to the left of the right-most extreme of the empirical null distribution. Accordingly, we are inclined to expect that there may be other such rejections of the null, i.e., other Type I errors. At the same time, we would expect fewer Type II errors, i.e., fewer M that equal 1. Conversely, if we see a Type II error, say $M_1 = 1$, it can be argued that we would be inclined to expect more Type II errors and, at

the same time, fewer Type I errors. These arguments are heuristic, of course, although intuitively appealing. From a practical perspective, their validity would need to be examined within specific applied contexts. However, that work lies beyond the scope of the present manuscript.

Together these high-level arguments suggest that a reasonable generic model for these errors is one in which there is positive correlation *within* the L's and M's, respectively, but negative correlation *between*. The conditions of the following theorem capture this notion, which in turn allow us to produce a sensible bound, improving on that of Theorem 8.

Theorem 8 *Let $\tilde{L}_j^{L_k}$ and $\tilde{M}_\ell^{L_k}$ be random variables distributed as L_j and M_ℓ respectively, conditional on $L_k = 1$. Similarly, let $\tilde{L}_j^{M_k}$ and $\tilde{M}_\ell^{M_k}$ be distributed as L_j and M_ℓ, conditional on $M_k = 1$. Suppose that*

i. $\tilde{L}_j^{L_k} \geq L_j$ and $\tilde{M}_\ell^{L_k} \leq M_\ell$, for $j \neq k$ and $\ell = 1, \ldots, m$, while

ii. $\tilde{L}_j^{M_k} \leq L_j$ and $\tilde{M}_\ell^{M_k} \geq M_\ell$, $j = 1 \ldots, n$ and $\ell \neq k$.

Then

$$d_{KS}(U, W) \leq ||\Delta f|| \left\{ Var(U) - (\lambda_1 + \lambda_2) \right\} \quad , \tag{12}$$

where $W \sim$ Skellam(λ_1, λ_2), with λ_1, λ_2 defined as in Theorem 2.

The proof of this theorem is given in the appendix, in Section A.6. We note that for a collection of binary random variables to satisfy conditions (i) and (ii) in the above theorem, it is sufficient, for example, to generate a vector of positively associated random variables $(L_1, \ldots, L_n, 1 - M_1, \ldots, 1 - M_m)$. The L's and the M's will then be positively associated within, but negatively associated between, which in turn implies the conditions (i.e., analogous to positive and negative relatedness, respectively). See Barbour and Chen (2005, p. 78), for example, for a brief summary of these latter notions and their relationships.

With this theorem, the following then holds for large networks with dependent and homogeneous errors, when the dependency is of the nature just defined.

Corollary 9 *Suppose that the collections of edge indicator random variables $\{Y_{ij}\}_{\{i,j\} \in E^c}$ and $\{Y_{ij}\}_{\{i,j\} \in E}$ satisfy conditions (i) and (ii) of Theorem 8 , playing the roles of the L's and M's, respectively. Then under assumptions (A1)-(A2),*

$$d_{KS}\left(D_E \,,\, \text{Skellam}(\lambda, \lambda) \right) = O\left(\frac{Var(D_E) - 2\lambda}{2\lambda} \right) \quad .$$

This result can be compared to that of Theorem 6 , where the edge noise was independent and the error in approximating by a Skellam decayed like n_v^{-1}. By way of contrast, Corollary 9 tells us that in order to achieve a decay in approximation error like $O(f(n_v))$, we must have $\text{Var}(D_E) = 2\lambda\,(1 + O(f(n_v)))$.

More generally, the quality of the approximation of D_E by a Skellam will be influenced by the nature of the dependency in the errors, as the latter manifests itself through the overall variance $\text{Var}(D_E)$. The nature of that dependency is highly problem-specific, and a detailed investigation of possible cases is well beyond the scope of the present paper, although we provide some initial insight later as it arises in the context of counting higher-order subgraphs, through Theorem 10 in Section 5.

However, nontrivial insight also can be gained into the influence of the level of dependency on accuracy through numerical work under the following simple model.

For a vector of binary random variables $(L_1, \ldots, L_n, 1 - M_1, \ldots, 1 - M_m)$, let $S = D + m$, where $D = \sum_{i=1}^n L_i - \sum_{i=1}^m M_i$. We equip S with a distribution of the form

$$\mathbb{P}(S = k) \propto \binom{n+m}{k}^{\nu - 1} \mathbb{P}(U + V = k) \quad , \tag{13}$$

for ν a real number, where U and V are binomial, with parameters (n, p) and (m, q), respectively. This distribution is a rescaling of that of the sum of two independent binomials, in a spirit analogous to the Conway-Maxwell binomial (COMB) distribution introduced recently by Kadane (2014). The COMB distribution is a simple extension of the binomial distribution that introduces dependency among the corresponding Bernoulli random variables. Our proposed distribution for S in (13) involves two binomial random variables, for which the corresponding Bernoulli random variables are dependent both within and between the two. Accordingly, we call this a COMB2 distribution.

Now impose assumption (A2) on this model. Since the assumption implies that $\mathbb{E}[D] = 0$, it follows that necessarily we must have $\mathbb{E}[S] = m$. Furthermore, the limiting Skellam distribution in Corollary 9 will be symmetric under this assumption. Symmetry can be imposed here on the distribution of S, and hence D, by taking $n = m$ and $q = 1 - p$. Therefore, we let $|E| = \binom{n_v}{2}/2 = n_v(n_v - 1)/4$ and $\alpha = 1 - \beta$. Note that this choice of $|E|$ means that our numerical work pertains to the case of dense graphs. (We are unable to exhibit a sparse variant of the COMB2 model with the necessary characteristics above.)

Note that when $\nu = 1$, the binary random variables $(L_1, \ldots, L_n, 1 - M_1, \ldots, 1 - M_m)$ are independent. On the other hand, proceeding along lines of reasoning similar to those in Kadane (2014), it can be argued that the COMB2 distribution, with the parameter constraints just described, renders the $(L_1, \ldots, L_n, 1 - M_1, \ldots, 1 - M_m)$ positively associated when $\nu < 1$, with the mass being transferred increasingly to the endpoints of the support of the distribution of D as $\nu \to -\infty$. As a result, per the discussion immediately following Theorem 8, the particular COMB2 distribution we have defined can be used to simulate network edge data in a way that satisfies the conditions of Corollary 9.

In Figure 2 are shown the results of numerical work calculating the Kolmogorov-Smirnov distance between the Skellam and standard normal approximations to the distribution of the discrepancy D_E in edge counts under the COMB2 distribution, for $\nu = 0, 0.5$, and 1.0. The noise levels used here are the same as used earlier in producing Figure 1. We see that the accuracy of the Skellam distribution decays slightly with increasing dependency in the errors, and with increasing noise levels.

5. Discussion

The propagation of uncertainty in network analysis is a topic that currently lags the field in development. Despite almost 15 years of work in the modern 'network science' era, on a vast array of topics, from researchers in many different disciplines, there remains a sizable gap in our understanding of how 'low level' errors (i.e., at the level of declaration of edge / non-edge status between vertex pairs) propagate to 'high level' summaries (e.g., subgraph counts, centralities, etc.). As a result, in most practical work, network summary statistics are cited without any indication of likely error.

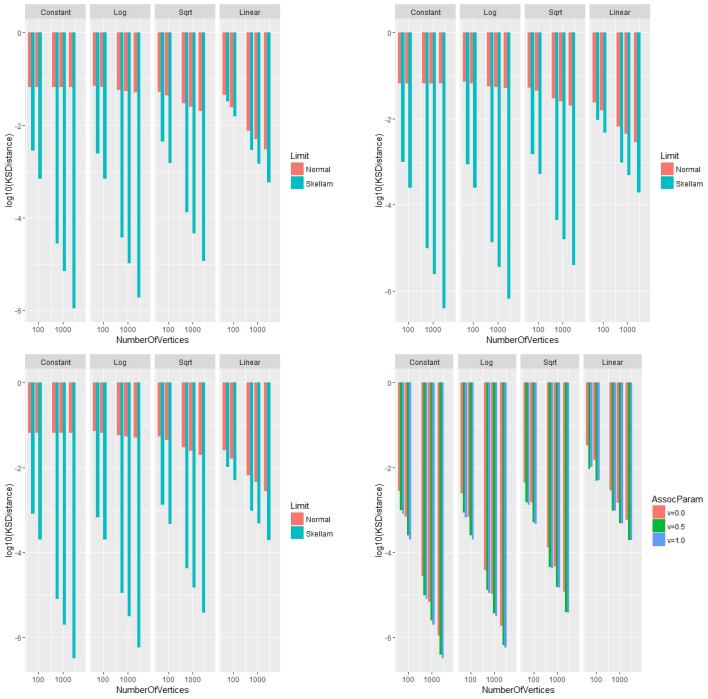


Figure 2: (Log)Kolmogorov-Smirnov distances to the distribution of the discrepancy D_E in edge counts under the COMB2 distribution, with $\nu = 0$ (top left), $\nu = 0.5$ (top right), and $\nu = 1.0$ (bottom left), for the Skellam and standard normal approximations. Also shown is a comparison of Skellam approximations as a function of ν (bottom right).

Our contributions in this paper are aimed at helping to begin laying a foundation for work in this area, with a focus on establishing an initial understanding of the distributional behavior of certain simple network summary statistics. Our choice to work with subgraph counts is both natural and motivated by convenience, whereas our emphasis on the specific case of large networks with low-rate measurement error is intended to capture a sizable fraction of what arguably is encountered in practice.

Ultimately, however, this initial work leaves us still well short of natural practical endpoints in applied statistics and machine learning for network science—for example, even to be able to do hypothesis testing or construct confidence intervals for graph-based parameters of interest (i.e., for $\eta(G)$). To close this gap is well beyond the scope of this paper. Indeed, we anticipate that it

will require the contributions of many papers in this emerging topic area, from researchers across a combination of probability, statistics and machine learning, and domain areas. Accordingly, we devote the remainder of this paper to a brief discussion of what we view as key challenges in terms of applied probability, statistical and machine learning methodology, and practical applications.

5.1 Challenges in Applied Probability

From the perspective of applied probability—including random graph theory—our problem formulation appears to be new. Our formulation is reminiscent of the type of 'signal plus noise' model commonly used in nonparametric function estimation and digital signal processing. Notably, in our formulation the true underlying graph G is fixed. This necessitates a different treatment than, say, traditional analysis of subgraph counts in classical Erdős-Rényi random graphs (e.g., Janson et al., 2011, Chs 3 & 6). In the special case where an Erdős-Rényi model is assumed, as well as assuming independence among the measurement errors, and the analysis is done without conditioning on G, then the problem could be viewed as involving a classical random graph wherein the (marginal) probability of an edge is a function not only of an initial edge frequency but also the Type I and Type II error rates. In general, however, either when G is fixed, as assumed in this paper, or from some other class of random graph models (e.g., various models with heterogeneous degree distributions), or when the measurement errors are dependent, the problem is more involved. By conditioning on G, our formulation allows us to focus our analysis firstly on a high-level notion of Type I and II errors among (non)edges, and then secondly on the manner in which the structure of the underlying graph G may interact with those errors.

We view our work in part as laying a key initial piece of the foundation on an important new class of problems in applied probability. However, we have provided a detailed analysis only for the most fundamental of subgraph count statistics, i.e., the number of edges in a network. Our initial work on extension to counts for subgraphs of higher order suggests that the problem becomes increasingly nontrivial. Specifically, the interaction of noise level, graph topology, and choice of subgraph would appear to need to be studied with care.

The following general theorem should be useful in exploring further in this direction.

Theorem 10 *Let H be a given subgraph of interest. Re-express the difference D in subgraph counts defined in equation (3) as*

$$D_H = \sum_{H' \notin \mathcal{C}_H} L_{H'} - \sum_{H' \in \mathcal{C}_H} M_{H'} \ ,$$

for $\mathcal{C}_H = \{H' \subseteq K_{n_v} : H' \cong H, H' \subseteq G\}$, where $L_{H'}$ and $M_{H'}$ are indicator variables of Type I and Type II error, respectively, for a subgraph H'. Under the assumption of independent edge noise,

$$d_{KS}(D_H, W) \leq ||\Delta f|| \{Var(D_H) - (\lambda_1 + \lambda_2)\} \ , \tag{14}$$

where $W \sim$ Skellam(λ_1, λ_2), with $\lambda_1 = \sum_{H' \notin \mathcal{C}_H} p_{H'}$ and $\lambda_2 = \sum_{H' \in \mathcal{C}_H} q_{H'}$, for $p_{H'} = \mathbb{E}[L_{H'}]$ and $q_{H'} = \mathbb{E}[M_{H'}]$.

This result follows directly from application of Theorem 8 and the comment immediately following that theorem. In particular, each of the indicator random variables $L_{H'}$ and $1 - M_{H'}$ may be expressed as a product of $n_{v(H)}$ choose two binary random variables, where $n_{v(H)}$ is the order of the subgraph H. Since these products are non-decreasing functions of their arguments, and their

arguments are independent, it follows that the collection of random variables defined by the union of the $L_{H'}$ and the $1 - M_{H'}$ are positively associated (e.g., Esary et al., 1967).

Application of this theorem to specific choices of subgraphs H requires calculation or bounding of the two key elements within brackets in (14), i.e., $\text{Var}(D_H)$ and $\lambda_1 + \lambda_2$, as well as control of the constant $||\Delta f||$. For the case of independent edge noise (which, nevertheless, yields *dependent* indicator variables $L_{H'}$ and $M_{H'}$), there is some hope that the former two quantities may be bounded through straightforward calculations, at least for low-order subgraphs. As for the latter quantity, under Conjecture 5 this term is controlled by a term of order $(\lambda_1 + \lambda_2)^{-1}$, but this conjecture, while supported by numerical work, remains to be proven.

By way of illustration, consider the problem of counting two-stars on the one-dimensional toroidal lattice of degree $2r$, $r \geq 1$. That is, the graph G is composed of a ring, with each vertex connected to its r nearest neighbors in each direction. The subgraph H is simply a chain of length two, i.e., a subgraph on three vertices $\{i, j, k\}$ with edges $\{i, j\}$ and $\{i, k\}$. In this case, the total number of two-stars is $\eta(G) = n_v \binom{2r}{2}$. Under homogeneous Type I and Type II error rates α and β, the parameters λ_1 and λ_2 in Theorem 10 take the values

$$\lambda_1 = 2rn_v \left(n_v - 2r - 1\right) \alpha \left(1 - \beta\right) + n_v \binom{n_v - 2r - 1}{2} \alpha^2$$

and

$$\lambda_2 = n_v \binom{2r}{2} \left[1 - (1 - \beta)^2\right] \ .$$

Here the two terms defining λ_1 derive from counting the number of vertex triples in G that possess only one edge or no edges among them, respectively. Under assumptions (A1) - (A4), and supposing that G is sparse, λ_1 and λ_2, are both $O(\log^2 n_v)$, so that neither of the sums in D_H dominates the other. However, λ_1 is in general not equal to λ_2, and so Conjecture 5 is pertinent here. At the same time, calculation of the variance in (14) is decidedly more involved, as it requires evaluation of the covariances $\text{Cov}\left(L_{H'}, L_{H''}\right)$, $\text{Cov}\left(M_{H'}, M_{H''}\right)$, and $\text{Cov}\left(L_{H'}, M_{H''}\right)$ over all pairs of vertex-triples, corresponding to putative subgraphs H' and H''. Under independent edge noise, the calculations simplify to some extent, in that these covariances are non-zero only in the case where the relevant pair of vertex triples is defined in terms of a total of only either three or four unique vertices. (When five or six unique vertices define the two triples, the indicators L and/or M are functions of two independent sets of edge indicators Y_{ij}.) Nevertheless, there still remain several subcases to enumerate for each of the three forms of covariance above in order to complete the calculation, rendering the exercise arguably feasible but tedious. Ultimately, however, since the denominator in (14) is $O\left(\log^2 n_v\right)$ under Conjecture 5, we see that under this particular noise level, with this particular graph topology, the convergence to Skellam is unlikely to be fast. Numerical experiments confirm this expectation.

This small illustration is sufficient to illustrate the point we assert above, i.e., that the interaction of noise level, graph topology, and choice of subgraph are likely to be nontrivial in the class of problems introduced here in this paper, and will likely require substantial and careful study in applied probability. Of course, looking even more broadly, by restricting our attention to subgraph counts we have only considered a portion of the full universe of network summary statistics. Clearly there are other classes of summaries of interest to be explored, and which can be expected to potentially display different behavior. Summaries relying on shortest path behavior (e.g., the diameter of a graph, between centrality of nodes, etc.) are a natural example.

5.2 Challenges in Statistics and Machine Learning Methodology

From the perspective of statistics and machine learning, the work in this paper is largely a necessary means to an end. In the context of network-based propagation of error, a central goal of interest in these fields would be, for example, to develop methods of hypothesis testing or confidence interval construction for graph-based parameters of interest (i.e., for $\eta(G)$) based on an observed graph $\hat{G}$. Alternatively, one might wish to use such network summaries (or perhaps even the networks themselves) as input into a regression or classifier. Here, again, much work remains to be done.

For parameter estimation and testing in the spirit of classical statistics (e.g., Statistics 101), it is typical to first understand the probability distribution underlying a quantity of interest. That is the goal of this manuscript and, for an important subclass of problems in this space, our results offer some initial insight. But knowledge of this distribution does not necessarily yield a tool for statistical inference. For example, to apply the Skellam results here in practice would require knowledge of the Type I and II error rates. If that knowledge is lacking (as generally is the case), then they must be estimated. How best to perform such estimation will presumably depend on context. To the best of our knowledge, this question has not been explored broadly in the literature. See Section 5.3 for additional comments in this direction.

In our work here, we have purposely taken an agnostic stance on the nature of G. However, it can be expected that additional leverage on the propagation of error problem—and, in particular, related problems of statistical inference—can be gained by incorporating additional model structure into the problem. For example, within the statistics and machine learning literatures, arguably the canonical methodology for producing an estimate $\hat{G}$ of a graph G from data is the neighborhood selection approach of Meinshausen and Bühlmann (2006) and the many derivatives thereof. In this approach, we assume we have available n independent and identically distributed observations of a n_v-dimensional random vector $\mathbf{X}$ that follows a multivariate normal distribution, with mean 0 and covariance matrix Σ. The graph G in this setting is taken to be the so-called concentration graph, defined so as to have an edge between a given vertex pair i and j if and only if the i, j-th entry of the concentration matrix Σ^{-1} is non-zero. Since it is typical to have $n \ll n_v$ in practice, methods of inference for this problem generally make use of complexity-penalized optimization. The original proposal in Meinshausen and Bühlmann (2006) makes use of ℓ_1-penalized regression. In what appears to be a first, the combination of Gaussian graphical models and hypothesis testing for network structure has been explored by Neykov et al. (2016) in recent work. There they characterize the complexity of what they call combinatorial inference problems through the use of minimax lower bounds. Algorithms are provided for tests that achieve those lower bounds.

Examples of other related work in this space include methodology for predicting network topology or attributes with models that explicitly include a component for network noise (e.g., Jiang et al., 2011; Jiang and Kolaczyk, 2012), the 'denoising' of noisy networks (e.g., Chatterjee, 2015), and the adaptation of methods for vertex classification using networks observed with errors Priebe et al. (2015). Also related is recent work by Cai et al. (2017) on the detection of ring lattice structure in the presence of random connections (i.e., small worlds). While there are undoubtedly other examples in this space that we have not cited here, it is nevertheless the case that we are currently still only at the start of methodological development of this type.

Finally, we note that network summary statistics have also long been used in practice as input to regressions and classifiers. A canonical example in the field of marketing is Hill et al. (2006), where the focus is on the identification of early adopters through the use of consumer networks,

with neighborhood summaries (e.g., proportion of neighbors adopting a product) being used as predictors. Similar use has been made in the social sciences. For example, Hahm et al. (2012) use vertex centrality measures from a friendship network as input in an effort to predict adolescents at risk of binge drinking. In these and other cases, it is certainly conceivable that the network summary measures are observed with error. In that case, incorporation of ideas like errors-in-variables regression might be appropriate (Fuller, 2009). Characterization of distributional properties for network summary measures, as in this paper, can be an important step towards understanding an appropriate error model for the predictors.

5.3 Challenges for Practical Applications

When all is said and done, probabilistic and statistical / machine learning development in the area of network-based propagation of error will only have significant practical impact when paired with an appropriate degree of context-specific understanding of the nature and magnitude of errors that arise in network construction. Certainly in most applied settings it is widely recognized by practitioners that there is measurement error associated with common types of network constructions. And in many settings the general issue has received nontrivial attention, such as, for example, in the context of protein-protein interaction networks (e.g., Hart et al., 2006) or social networks (e.g, Almquist, 2012).

In order to interface in a productive manner with theoretical and methodological development, such as offered in this paper, it will be important to be able to quantify the noise levels in applications. The difficulty of this task, of course, can vary greatly by context, as well as can the magnitude of the noise. In the case of protein-protein interaction networks, for example, it is understood that Type I and Type II error rates associated with traditional affinity binding experiments can be alarmingly high. Hart et al. (2006) summarized the fairly substantial literature on quantifying these rates, finding that values of $\alpha \approx 0.35$ and $\beta \approx 0.40$ were not atypical. In this case, such rates derive largely from experimental evidence. Alternatively, Balachandran et al. (2013a) employ empirical null principles (Efron, 2012) for correlation-based inference of a small gene regulatory network in yeast to estimate error rates $\alpha \approx 0.004$ and $\beta \approx 0.009$. But these later should be taken with a grain of salt. Cosgrove et al. (2010) show that such nominal estimates of error rates can be decidedly off from actual error rates when the underlying model assumptions are violated in ways that can often be expected in practice (e.g., due to experimental batch effects).

Finally, it is important to note that in some areas of network science the notion of 'ground truth'—in the sense of what constitutes a true (non)edge—can be a slippery concept. For example, there continues to be much discussion and debate in computational neuroscience as to what functional (or, alternatively, effective) connectivity represents, as inferred from data (e.g., from neuroimaging measurements), in contrast to anatomical (or, alternatively, physical or structural) connectivity. See, for instance, the review by Friston (2011). Such concerns further complicate the task of defining and quantifying appropriately the notions of Types I and II errors used in this paper.

Acknowledgments

We thank Kostas Spiliopoulos and Tasso Kaper for helpful discussions. This work was supported in part by AFOSR award 12RSL042 and NSF grant CNS-0905565.

Appendix A.

A.1 Proof of Theorem 1

Proof We begin with the operator,

$$\mathcal{A}[f(k)] = \lambda_1 f(k+1) - k f(k) - \lambda_2 f(k-1)$$

with the intent of showing that the random variable $W \sim \text{Skellam}(\lambda_1, \lambda_2)$ if and only if $\mathbb{E}\mathcal{A}[f(W)] = 0$ for any bounded function $f : \mathbb{Z} \mapsto \mathbb{R}$.

We begin with the necessity direction and the computation of

$$\begin{aligned}\mathbb{E}\mathcal{A}[f(W)] &= \mathbb{E}[\lambda_1 f(W+1) - W f(W) - \lambda_2 f(W-1)] \\ &\propto \sum_{k=-\infty}^{\infty} [\lambda_1 f(k+1) - k f(k) - \lambda_2 f(k-1)] \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k\end{aligned}$$

where $\propto$ is to be read as "proportional to," and as shorthand, we write I_k for $I_k(2\sqrt{\lambda_1\lambda_2})$. By standard properties of I_k (e.g., Abramowitz and Stegun, 1972) we have that

$$I_{k-1} - I_{k+1} = \frac{k}{\sqrt{\lambda_1\lambda_2}} I_k$$

or, in other words,

$$\sqrt{\lambda_1\lambda_2}\frac{I_{k-1}}{I_k} - \sqrt{\lambda_1\lambda_2}\frac{I_{k+1}}{I_k} = k. \tag{15}$$

This means that

$$\begin{aligned}\mathbb{E}\mathcal{A}[f(W)\,1\{W \le n\}] &\propto \sum_{k=-\infty}^{n} \left[\sqrt{\frac{\lambda_1}{\lambda_2}} f(k+1) - \frac{I_{k-1}}{I_k} f(k) + \right. \\ &\qquad \left. \frac{I_{k+1}}{I_k} f(k) - \sqrt{\frac{\lambda_2}{\lambda_1}} f(k-1)\right] \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k \\ &= \sum_{k=-\infty}^{n} \left[\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k+1} I_k f(k+1) - \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k} I_{k-1} f(k)\right] \\ &\qquad + \sum_{k=-\infty}^{n} \left[\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k} I_{k+1} f(k) - \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k-1} I_k f(k-1)\right] \\ &= \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{n+1} I_n f(n+1) + \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{n} I_{n+1} f(n).\end{aligned}$$

Now, since f is bounded,

$$\lim_{n\to\infty} \left\{ \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{n+1} I_n f(n+1) + \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{n} I_{n+1} f(n) \right\} = 0$$

so that by monotone convergence,

$$\begin{aligned}\mathbb{E}\left[\mathcal{A}\left[f(W)\right]\right] &= \lim_{n\to\infty} \mathbb{E}\mathcal{A}\left[f\left(W\right)1\left\{W\leq n\right\}\right] \\ &= \lim_{n\to\infty}\left\{\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{n+1} I_n f\left(n+1\right)+\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{n} I_{n+1} f\left(n\right)\right\} \\ &= 0\end{aligned}$$

which proves the claim.

To prove sufficiency, we begin with $\mathbb{E}\mathcal{A}\left[f\left(W\right)\right]=0$ and suppose that $f_k\left(j\right)=1\left\{j=k\right\}$ for some $j\in\mathbb{Z}$ in which case

$$\lambda_1 p\left(k-1\right)-kp\left(k\right)-\lambda_2 p\left(k+1\right)=0$$

where $p\left(k\right)=\mathbb{P}\left(W=k\right)$. An *ansatz* of

$$S\left(k\right)=\left(\sqrt{\tfrac{\lambda_1}{\lambda_2}}\right)^k I_k\left(2\sqrt{\lambda_1\lambda_2}\right) \quad \text{and} \quad T\left(k\right)=\left(\sqrt{\tfrac{\lambda_1}{\lambda_2}}\right)^k K_k\left(2\sqrt{\lambda_1\lambda_2}\right)$$

shows that S and T form two linearly independent solutions to this second order linear difference equation, where $I_k(x)$ and $K_k(x)$ are the modified Bessel functions of the first and second kinds. Thus, we know that the general solution is given by,

$$p\left(k\right)=C_1 S\left(k\right)+C_2 T\left(k\right)$$

for some constants $C_1, C_2\in\mathbb{R}$.

Now, to determine the constants C_1 and C_2 we appeal to the fact that $\sum_{k=-\infty}^{\infty} p\left(k\right)=1$. Since $I_k, K_k>0$ for all $k\in\mathbb{Z}$ and $\sum_{k=-\infty}^{\infty} K_k=\infty$ it must be that $C_2=0$. Now, consider the generating function

$$e^{\frac{z}{2}(t+1/t)}=\sum_{k=-\infty}^{\infty} t^k I_k\left(z\right)$$

which means that

$$\begin{aligned}C_1 &= \frac{1}{\sum_{k=-\infty}^{\infty}\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k\left(2\sqrt{\lambda_1\lambda_2}\right)} \\ &= \frac{1}{e^{\sqrt{\lambda_1\lambda_2}\left(\sqrt{\frac{\lambda_1}{\lambda_2}}+\sqrt{\frac{\lambda_2}{\lambda_1}}\right)}} \\ &= e^{-(\lambda_1+\lambda_2)}\end{aligned}$$

so that

$$p\left(k\right)=e^{-(\lambda_1+\lambda_2)}\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k\left(2\sqrt{\lambda_1\lambda_2}\right)$$

so that $W\sim\text{Skellam}\left(\lambda_1,\lambda_2\right)$. ■

A.2 Proof of Theorem 2

Proof Given that f_x is a solution to $\mathcal{A}[f_x(k)]=g_x(k)$, we have

$$\lambda_1 f_x(k+1)-kf_x(k)-\lambda_2 f_x(k-1)=1(k\leq x)-\mathbb{P}\left[W\leq x\right].$$

Substituting $k=U$ and taking expected values, we obtain,

$$\left|\mathbb{P}\left[U\leq x\right]-\mathbb{P}\left[W\leq x\right]\right|=\left|\mathbb{E}\left[\lambda_1 f_x(U+1)-Uf_x(U)-\lambda_2 f_x(U-1)\right]\right| \quad . \tag{16}$$

Next, recall from (5) that $U=\sum_{k=1}^n L_k-\sum_{k=1}^m M_k$. Since $\lambda_1=\sum_{k=1}^n p_k$ and $\lambda_2=\sum_{k=1}^m q_k$, we have after conditioning on L_k and M_k,

$$\begin{aligned}&\left|\mathbb{E}\left[\lambda_1 f_x(U+1)-Uf_x(U)-\lambda_2 f_x(U-1)\right]\right| \\ &= \left|\sum_{k=1}^n \mathbb{E}\left[p_k f_x(U+1)-L_k f_x(U)\right]+\sum_{k=1}^m \mathbb{E}\left[M_k f_x(U)-q_k f_x(U-1)\right]\right| \\ &= \left|\sum_{k=1}^n p_k\left(\mathbb{E}\left[f_x(U+1)\right]-\mathbb{E}\left[f_x(U)|L_k=1\right]\right)+\sum_{k=1}^m q_k\left(\mathbb{E}\left[f_x(U)|M_k=1\right]-\mathbb{E}\left[f_x(U-1)\right]\right)\right| \\ &= \left|\sum_{k=1}^n p_k\left(\mathbb{E}\left[f_x(U+1)-f_x\left(U_k^{(L)}+1\right)\right]\right)+\sum_{k=1}^m q_k\left(\mathbb{E}\left[f_x\left(U_k^{(M)}-1\right)-f_x(U-1)\right]\right)\right| \\ &\leq \sum_{k=1}^n p_k||\Delta f||\mathbb{E}\left|U-U_k^{(L)}\right|+\sum_{k=1}^m q_k||\Delta f||\mathbb{E}\left|U-U_k^{(M)}\right| \\ &= ||\Delta f||\left[\sum_{k=1}^n p_k\mathbb{E}\left|U-U_k^{(L)}\right|+\sum_{k=1}^m q_k\mathbb{E}\left|U-U_k^{(M)}\right|\right].\end{aligned}$$

Combining this with (16) yields the result. ■

A.3 Proof of Theorem 3

Proof First, consider the solution to

$$\lambda_1 f\left(k+1\right)-kf\left(k\right)-\lambda_2 f\left(k-1\right)=g\left(k\right), \tag{17}$$

for some bounded function $g:\mathbb{Z}\mapsto\mathbb{R}$, with the boundary condition

$$\lim_{k\to-\infty}\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k f\left(k\right)=0. \tag{18}$$

We use (15) to substitute for k in (17). Then, multiplying both sides of (17) by $\left(\sqrt{\lambda_1/\lambda_2}\right)^k I_k$, we obtain,

$$\begin{aligned}
&\lambda_1\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k f(k+1) - \lambda_1\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k-1} I_{k-1} f(k) \\
&+ \lambda_2\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k+1} I_{k+1} f(k) - \lambda_2\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k} I_k f(k-1) \\
&= \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k g(k)
\end{aligned}$$

which is the same as,

$$\begin{aligned}
&\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k+1} I_k f(k+1) - \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k} I_{k-1} f(k) \\
&+ \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k} I_{k+1} f(k) - \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{k-1} I_k f(k-1) \\
&= \frac{1}{\sqrt{\lambda_1\lambda_2}}\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k g(k).
\end{aligned}$$

Notice that we have grouped terms together so that summing over k yields a telescoping sum. So, summing over $k \in \{-\infty, \ldots, n\}$ and using the boundary condition (18),

$$\left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{n+1} I_n f(n+1) + \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^{n} I_{n+1} f(n) = \frac{1}{\sqrt{\lambda_1\lambda_2}} \sum_{k=-\infty}^{n} \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k g(k).$$

Now, multiplying both sides by $(-1)^{n+1}/(I_n I_{n+1})$ and summing over $n \in \{c, c+1, \ldots, m\}$ for $m > c$ and over $n \in \{m, m+1, \ldots, c-1\}$, for some initial condition $c \in \mathbb{Z}$ and $f(c) \in \mathbb{R}$, we obtain

$$f(m) = \begin{cases} (-1)^m \left(\sqrt{\frac{\lambda_2}{\lambda_1}}\right)^m I_m \left[(-1)^c \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^c \frac{1}{I_c} f(c) \right. \\ \left. + \frac{1}{\sqrt{\lambda_1\lambda_2}} \sum_{n=c}^{m-1} \frac{(-1)^{n+1}}{I_n I_{n+1}} \sum_{k=-\infty}^{n} \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k g(k)\right] & \text{if } m > c \\ (-1)^m \left(\sqrt{\frac{\lambda_2}{\lambda_1}}\right)^m I_m \left[(-1)^c \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^c \frac{1}{I_c} f(c) \right. \\ \left. - \frac{1}{\sqrt{\lambda_1\lambda_2}} \sum_{n=m}^{c-1} \frac{(-1)^{n+1}}{I_n I_{n+1}} \sum_{k=-\infty}^{n} \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k g(k)\right] & \text{if } m < c. \end{cases}$$

Note that if

$$g(k) = g_x(k) = 1\{k \le x\} - \mathbb{P}(W \le x)$$

then

$$\sum_{k=-\infty}^{n} \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k g(k) = \begin{cases} e^{\lambda_1+\lambda_2}\mathbb{P}(W \le n)\,\mathbb{P}(W > x) & \text{if } n \le x \\ e^{\lambda_1+\lambda_2}\mathbb{P}(W \le x)\,\mathbb{P}(W > n) & \text{if } n \ge x \end{cases}$$

since, for example if $n \le x$

$$\begin{aligned}
\sum_{k=-\infty}^{n} \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^k I_k g(k) &= e^{\lambda_1+\lambda_2} \sum_{k=-\infty}^{n} \mathbb{P}(W=k)\, g(k) \\
&= e^{\lambda_1+\lambda_2} \sum_{k=-\infty}^{n} \mathbb{P}(W=k)\,[1\{k \le x\} - \mathbb{P}(W \le x)] \\
&= e^{\lambda_1+\lambda_2}\,[\mathbb{P}(W \le \min\{n,x\}) - \mathbb{P}(W \le x)\,\mathbb{P}(W \le n)] \\
&= e^{\lambda_1+\lambda_2}\mathbb{P}(W \le n)\,[1 - \mathbb{P}(W \le x)] \\
&= e^{\lambda_1+\lambda_2}\mathbb{P}(W \le n)\,\mathbb{P}(W > x).
\end{aligned}$$

The case that $n \ge x$ is similar. This means that

$$f_x(m) = \begin{cases} (-1)^m \left(\sqrt{\frac{\lambda_2}{\lambda_1}}\right)^m I_m \left[(-1)^c \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^c \frac{1}{I_c} f(c) \right. \\ \left. + \frac{e^{\lambda_1+\lambda_2}}{\sqrt{\lambda_1\lambda_2}} \sum_{n=c}^{m-1} \frac{(-1)^{n+1}}{I_n I_{n+1}} \mathbb{P}(W \le \min\{n,x\})\,\mathbb{P}(W > \max\{n,x\})\right] & \text{if } m > c \\ (-1)^m \left(\sqrt{\frac{\lambda_2}{\lambda_1}}\right)^m I_m \left[(-1)^c \left(\sqrt{\frac{\lambda_1}{\lambda_2}}\right)^c \frac{1}{I_c} f(c) \right. \\ \left. - \frac{e^{\lambda_1+\lambda_2}}{\sqrt{\lambda_1\lambda_2}} \sum_{n=m}^{c-1} \frac{(-1)^{n+1}}{I_n I_{n+1}} \mathbb{P}(W \le \min\{n,x\})\,\mathbb{P}(W > \max\{n,x\})\right] & \text{if } m < c. \end{cases}$$

■

A.4 Proof of Theorem 4

Our proof of Theorem 4 is highly involved, from an analysis perspective, but the overall program can be stated in a relatively succinct manner. Accordingly, we sketch here the overall program behind our proof and refer the interested reader to the online appendix for a detailed account.

Proof Recall that we are trying to obtain a bound on $|\Delta f_x(j)| = |f_x(j+1) - f_x(j)|$ independent of $x \in \mathbb{R}$ and $j \in \mathbb{Z}$. From Theorem 3, we have the solution to the Stein equation, however to use it to bound $|\Delta f_x(j)|$, we need to simplify it further. For ease of notation, we simply refer to f instead of f_x and g instead of g_x.

First, note that we have the freedom to choose the initial condition $(c, f(c))$. Making the choice that $c = \lambda_2 - \lambda_1$, and hence that $c = 0$ under the assumption that $\lambda_1 = \lambda_2$, we are able to simplify our expression for f in Theorem 3 to read, in the case that $m > 0$, as

$$\begin{aligned}
f(m+1) = (-1)^{m+1} I_{m+1} \Bigg[&\frac{1}{I_0} f(0) \\
&- \frac{e^{2\lambda}}{\lambda} \frac{1}{I_0 I_1} \mathbb{P}(W \le \min\{0,x\})\,\mathbb{P}(W > \max\{0,x\}) \\
&- \frac{e^{2\lambda}}{\lambda} \sum_{n=0}^{m-1} \frac{(-1)^{n+1}}{I_{n+1} I_{n+2}} \mathbb{P}(W \le \min\{n+1,x\})\,\mathbb{P}(W > \max\{n+1,x\}) \Bigg]
\end{aligned}$$

and, in the case that if $m < 0$, as

$$f(m-1) = (-1)^{m-1} I_{m-1} \left[\frac{1}{I_0} f(0) - \frac{e^{2\lambda}}{\lambda} \frac{1}{I_0 I_1} \mathbb{P}(W \leq \min\{-1, x\}) \mathbb{P}(W > \max\{-1, x\}) + \frac{e^{2\lambda}}{\lambda} \sum_{n=m}^{-1} \frac{(-1)^{n+1}}{I_{n-1} I_n} \mathbb{P}(W \leq \min\{n-1, x\}) \mathbb{P}(W > \max\{n-1, x\}) \right].$$

Finally, for the case $m = 0$, we have

$$\begin{aligned} f(0) &= \frac{e^{2\lambda}}{2\lambda} \frac{1}{I_0 + I_1} [\mathbb{P}(W \leq \min\{0, x\}) \mathbb{P}(W > \max\{0, x\}) \\ &\quad + \mathbb{P}(W \leq \min\{-1, x\}) \mathbb{P}(W > \max\{-1, x\})]. \end{aligned}$$

Next, through manipulation of the arguments in the sums defining the above expressions for f, exploiting properties of the modified Bessel functions I_k, and applying the triangle inequality, we are able to produce bounds on the differences $|f(m+1) - f(m)|$ of the form

$$\begin{aligned} |f(m+1) - f(m)| &\leq \frac{\mathbb{P}(W \leq x)}{\lambda} \left\{ \left| \sum_{n=1,3,\ldots}^{m-1} \frac{I_{m+1}}{I_{n+2}} - \frac{I_m}{I_{n-1}} \right| \right. \\ &\quad + \sum_{n=1,3,\ldots}^{m-1} H(n) \left| \frac{I_{m+1} I_n}{I_{n+1} I_{n+2}} - \frac{I_m}{I_{n+1}} - \frac{I_{m+1}}{I_{n+1}} + \frac{I_m}{I_{n-1}} \right| \Bigg\} \\ &\quad + \left| I_m \frac{1}{I_0} f(0) + I_{m+1} \left\{ \frac{1}{I_0} f(0) \right. \right. \\ &\quad \left. \left. - \frac{e^{2\lambda}}{\lambda} \frac{1}{I_0 I_1} \mathbb{P}(W \leq \min\{0, x\}) \mathbb{P}(W > \max\{0, x\}) \right\} \right| , \end{aligned} \tag{19}$$

if m is even, and

$$\begin{aligned} |f(m+1) - f(m)| &\leq \frac{\mathbb{P}(W \leq x)}{\lambda} \left\{ \left| \sum_{n=1,3,\ldots}^{m-1} \frac{I_{m+1}}{I_{n+2}} - \frac{I_m}{I_{n-1}} \right| \right. \\ &\quad + \sum_{n=1,3,\ldots}^{m-1} H(n) \left| \frac{I_{m+1} I_n}{I_{n+1} I_{n+2}} - \frac{I_m}{I_{n+1}} - \frac{I_{m+1}}{I_{n+1}} + \frac{I_m}{I_{n-1}} \right| \\ &\quad + |H(m+1) - H(m)| \Bigg\} \\ &\quad + \left| I_m \frac{1}{I_0} f(0) + I_{m+1} \left\{ \frac{1}{I_0} f(0) \right. \right. \\ &\quad \left. \left. - \frac{e^{2\lambda}}{\lambda} \frac{1}{I_0 I_1} \mathbb{P}(W \leq \min\{0, x\}) \mathbb{P}(W > \max\{0, x\}) \right\} \right| , \end{aligned} \tag{20}$$

if m is odd. Here $H(n) = \mathbb{P}(W > n) / \mathbb{P}(W = n)$ is the inverse of the hazard function of the Skellam distribution (and is not to be confused with our use of H in the main body of the paper as a subgraph of the graph G).

Note that (19) is defined by three key terms, while (20) has the same three, augmented by the addition of a fourth, i.e., $|H(m+1) - H(m)|$. Through a series of arguments (the result for each of which is presented as a separate proposition in the online appendix), we are able to control each of these terms as follows. First, we show that

$$\sup_{m \in \mathbb{N}^+} \left| \sum_{n=1,3,\ldots}^{m-1} \frac{I_{m+1}}{I_{n+2}} - \frac{I_m}{I_{n-1}} \right| \leq 5 \ .$$

Next we show that

$$\sum_{n=1,3,\ldots}^{m-1} H(n) \left| \frac{I_{m+1} I_n}{I_{n+1} I_{n+2}} - \frac{I_m}{I_{n+1}} - \frac{I_{m+1}}{I_{n+1}} + \frac{I_m}{I_{n-1}} \right| \leq 73 \ ,$$

for $\lambda \geq 1$. And furthermore, we show that

$$\begin{aligned} &\left| I_m \frac{1}{I_0} f(0) + I_{m+1} \left\{ \frac{1}{I_0} f(0) - \frac{e^{2\lambda}}{\lambda} \frac{1}{I_0 I_1} \mathbb{P}(W \leq \min\{0, x\}) \mathbb{P}(W > \max\{0, x\}) \right\} \right| \\ \leq \ &\frac{\mathbb{P}(W \leq x)}{\lambda} \ . \end{aligned}$$

Finally, it is clear that

$$\begin{aligned} &H(m) - H(m+1) \\ = \ &\frac{\mathbb{P}(W > m)}{\mathbb{P}(W = m)} - \frac{\mathbb{P}(W > m+1)}{\mathbb{P}(W = m+1)} \\ = \ &\frac{1}{\mathbb{P}(W = m)} [\mathbb{P}(W > m) - \mathbb{P}(W > m+1)] \\ = \ &\frac{\mathbb{P}(W = m+1)}{\mathbb{P}(W = m)} \\ \leq \ &1 \end{aligned}$$

and, at the same time $H(m) - H(m+1) \geq 0$ so, we have that we may bound the magnitude of this final term by 1.

As a result of all of the above, we may conclude that

$$|f(m+1) - f(m)| \leq \frac{80}{\lambda}$$

for $m > 0$. Or, equivalently, we may express the right-hand side above as $160/2\lambda$.

The argument for the case of $m < 0$ involves similar reasoning, as described in the online appendix. ■

A.5 Proof of Theorems 6 and 7

A.5.1 PROOF OF THEOREM 6

Proof The terms $\mathbb{E}\left|U - U_k^{(L)}\right|$ and $\mathbb{E}\left|U - U_k^{(M)}\right|$ in (8) measure the dependence of U on the events $L_k = 1$ and $M_k = 1$, respectively. In the context of the empirical graph $\hat{G}$, the random variables L are equal to Y_{ij}, for $\{i,j\} \in E^c$, while the random variables M are equal to Y_{ij}, for $\{i,j\} \in E$. With the Y_{ij} assumed independent, $U_k^{(L)}$ and $U_k^{(M)}$ are independent of their respective events, and so we obtain

$$d_{KS}(U,W) \leq ||\Delta f|| \left[\sum_{k=1}^{n} p_k^2 + \sum_{k=1}^{m} q_k^2\right] \ . \tag{21}$$

Accordingly, and drawing on definitions and the result of Theorem 4,

$$\begin{aligned} d_{KS}\left(D_E, \text{Skellam}(\lambda,\lambda)\right) \leq \frac{1}{\lambda}\left[\sum_{(i,j)\in E^c} \alpha^2 + \sum_{(i,j)\in E} \beta^2\right] &= \frac{|E^c|\alpha^2 + |E|\beta^2}{|E^c|\alpha} \\ &= \frac{|E^c|\alpha^2 + |E|\left(\frac{|E^c|}{|E|}\right)^2\alpha^2}{|E^c|\alpha} \\ &= \alpha + \frac{|E^c|}{|E|}\alpha \\ &= \alpha + \frac{\binom{n_v}{2} - |E|}{|E|}\alpha \\ &= \frac{\binom{n_v}{2}}{|E|}\alpha \ . \end{aligned}$$

Noting that $\alpha = \lambda/|E^c|$, and recalling that $|E^c| = \Theta\left(n_v^2\right)$ under both sparse and dense graphs G, the last quantity above is seen to behave like $\lambda/|E|$ which, under assumption (A3) and our definition of sparse and dense in Section 2 , reduces to $O\left(n_v^{-1}\right)$. So the bound in (9) is established.

Note that the right-hand side of (21) is analogous to the classical form of the bound for individual sums of independent indicator random variables (e.g., Barbour and Chen, 2005). As remarked in the main text, for this particular case of independent Y_{ij}, those more classical techniques could also be used to produce the result of Theorem 6. Specifically, Let $T_1, T_2, \tilde{T}_1$, and $\tilde{T}_2$ be independent random variables supported on the integers. Denote by $d_{TV}(X_1, X_2)$ the total-variation distance between two random variables X_1 and X_2. Then

$$\begin{aligned} d_{KS}\left(T_1 - T_2, \tilde{T}_1 - \tilde{T}_2\right) &\leq d_{TV}\left(T_1 - T_2, \tilde{T}_1 - \tilde{T}_2\right) \\ &\leq d_{TV}\left((T_1, T_2), (\tilde{T}_1, \tilde{T}_2)\right) \\ &\leq d_{TV}\left(T_1, \tilde{T}_1\right) + d_{TV}\left(T_2, \tilde{T}_2\right) \ , \end{aligned}$$

where the first inequality exploits the fact that total-variation distance provides an upper bound on Kolmogorov-Smirnov distance, and the second and third inequalities follow from Lemmas 3.6.3

and 3.6.2 of Durrett (2010), respectively. Now define

$$T_1 = \sum_{\{i,j\}\in E^c} Y_{ij} \quad \text{and} \quad T_2 = \sum_{\{i,j\}\in E} (1 - Y_{ij}) \ ,$$

and let $\tilde{T}_1$ and $\tilde{T}_2$ be independent Poisson random variables with common mean λ. Setting $\lambda = |E^c|\alpha = |E|\beta$, and applying to each of $d_{TV}\left(T_1, \tilde{T}_1\right)$ and $d_{TV}\left(T_2, \tilde{T}_2\right)$ the standard Stein bounds for Poisson approximation to sums of independent indicators (e.g., Barbour and Chen, 2005, Eqn 2.6), we again obtain that

$$d_{KS}\left(D_E, \text{Skellam}(\lambda,\lambda)\right) \leq \frac{1}{\lambda}\left[\sum_{(i,j)\in E^c} \alpha^2 + \sum_{(i,j)\in E} \beta^2\right] = \frac{|E^c|\alpha^2 + |E|\beta^2}{|E^c|\alpha} \ ,$$

and the rest follows. ■

A.5.2 PROOF OF THEOREM 7

To establish the bounds in (10) and (11), we use the following result from Stein's method for the normal distribution (e.g., Barbour and Chen, 2005).

Theorem 11 *Let $\xi_1, \ldots, \xi_n$ be independent random variables which have zero means and finite variances $\mathbb{E}\left[\xi_i^2\right] = \sigma_i^2$, $1 \leq i \leq n$, and satisfy $\sum_{i=1}^n \sigma_i^2 = 1$. If $F_n(x)$ is the cumulative distribution function of $\sum_{i=1}^n \xi_i$, then, for every $\epsilon > 0$,*

$$\frac{1 - e^{-\frac{\epsilon^2}{4}}}{40}\sum_{i=1}^{n} \mathbb{E}\left[\xi_i^2 I_{\{|\xi_i|>\epsilon\}}\right] - \sum_{i=1}^{n}\sigma_i^4 \leq \sup_{x\in\mathbb{R}}|F_n(x) - \Phi(x)| \leq 7\sum_{i=1}^{n}\mathbb{E}\left[|\xi_i|^3\right].$$

Proof of Theorem 7 We apply Theorem 11, with $\xi_i = X_i/\sigma$ where X_i is a term in one of the sums of D_E, to establish each of our upper and lower bounds in turn.

Upper Bounds in (10) and (11): First, note that since

$$\sum_{i=1}^{n}\mathbb{E}\left[|\xi_i|^3\right] = \frac{\sum_{i=1}^n \mathbb{E}\left[|X_i|^3\right]}{\sigma^3} \ ,$$

and

$$\mathbb{E}\left[|X_i|^3\right] = \alpha(1-\alpha)\left[(1-\alpha)^2 + \alpha^2\right] \quad \text{or} \quad \beta(1-\beta)\left[(1-\beta)^2 + \beta^2\right] \ ,$$

with n understood to be either $|E^c|$ or $|E|$, it follows that

$$\begin{aligned} \sum_{i=1}^{n}\mathbb{E}\left[|\xi_i|^3\right] &= \frac{\alpha(1-\alpha)\left[(1-\alpha)^2+\alpha^2\right]|E^c| + \beta(1-\beta)\left[(1-\beta)^2+\beta^2\right]|E|}{\left(\alpha(1-\alpha)|E^c| + \beta(1-\beta)|E|\right)^{\frac{3}{2}}} \\ &\leq \max\{(1-\alpha)^2+\alpha^2, (1-\beta)^2+\beta^2\}\frac{\alpha(1-\alpha)|E^c| + \beta(1-\beta)|E|}{\left(\alpha(1-\alpha)|E^c| + \beta(1-\beta)|E|\right)^{\frac{3}{2}}} \\ &= \frac{\max\{(1-\alpha)^2+\alpha^2, (1-\beta)^2+\beta^2\}}{\left(\alpha(1-\alpha)|E^c| + \beta(1-\beta)|E|\right)^{\frac{1}{2}}} \\ &= \frac{\max\{(1-\alpha)^2+\alpha^2, (1-\beta)^2+\beta^2\}}{\sqrt{2-(\alpha+\beta)}} \cdot \frac{1}{\sqrt{\alpha|E^c|}} \ , \end{aligned}$$

where in the last equality we have used the fact that $\beta = (|E^c|/|E|)\,\alpha$ follows from (A2). Finally, note that

$$(1-\alpha)^2 + \alpha^2 = 1 - 2\alpha + 2\alpha^2 = 1 - 2\alpha(1-\alpha) \leq 1$$

and the same holds for $(1-\beta)^2 + \beta^2$, since $0 \leq \alpha, \beta \leq 1$, so that

$$\sum_{i=1}^{n} \mathbb{E}\left[|\xi_i|^3\right] \leq \frac{1}{\sqrt{2-(\alpha+\beta)}} \cdot \frac{1}{\sqrt{\alpha|E^c|}} \ .$$

This immediately implies, after another application of $\beta = (|E^c|/|E|)\alpha$,

$$\sup_{x\in\mathbb{R}} |F_n(x) - \Phi(x)| \leq \frac{7}{\sqrt{2-(\alpha+\frac{|E^c|}{|E|}\alpha)}} \cdot \frac{1}{\sqrt{\alpha|E^c|}} \ .$$

Using $\alpha = \lambda/|E^c|$, and invoking the assumption of low-rate measurement error in (A3) and the definitions of sparse and dense graphs in Section 2, the upper bounds in (10) and (11) follow.

Lower bound in (10) and (11): First, note that since $\xi_i = X_i/\sigma$, $\sigma_i^2 = \alpha(1-\alpha)/\sigma^2$ or $\sigma_i^2 = \beta(1-\beta)/\sigma^2$. Thus,

$$\begin{aligned}
\sum_{i=1}^{n} \sigma_i^4 &= \frac{(\alpha(1-\alpha))^2|E^c| + (\beta(1-\beta))^2|E|}{(\alpha(1-\alpha)|E^c| + \beta(1-\beta)|E|)^2} \\
&= \frac{(\alpha(1-\alpha))^2|E^c| + \left(\frac{|E^c|}{|E|}\alpha(1-\beta)\right)^2|E|}{(2-(\alpha+\beta))^2(\alpha|E^c|)^2} \\
&= \frac{1}{|E^c|} \cdot \frac{1}{(2-(\alpha+\beta))^2} \cdot \left[(1-\alpha)^2 + \frac{|E^c|}{|E|}(1-\beta)^2\right] \\
&= \frac{1}{(2-(\alpha+\beta))^2}\left[\frac{(1-\alpha)^2}{|E^c|} + \frac{(1-\beta)^2}{|E|}\right]
\end{aligned} \ ,$$

where in the second equality, we have used $\beta = (|E^c|/|E|)\alpha$.

Next, choose $\epsilon = 1/(2\sigma)$. Note that this is the midpoint of the intervals

$$\left(\frac{\alpha}{\sigma}, \frac{1-\alpha}{\sigma}\right), \quad \text{and} \quad \left(\frac{\beta}{\sigma}, \frac{1-\beta}{\sigma}\right)$$

if $\alpha, \beta < 1/2$ and of the intervals

$$\left(\frac{1-\alpha}{\sigma}, \frac{\alpha}{\sigma}\right), \quad \text{and} \quad \left(\frac{1-\beta}{\sigma}, \frac{\beta}{\sigma}\right) \ .$$

if $\alpha, \beta \geq 1/2$. In either case, these are the endpoints of the interval formed by the values of $|\xi_i| = |X_i|/\sigma$.

Due to the symmetry in these intervals about $\frac{1}{2}$ we may, without loss of generality, assume $\alpha, \beta < 1/2$. In doing so, and using $\beta = (|E^c|/|E|)\alpha$,

$$\begin{aligned}
&\frac{1-e^{-\frac{\epsilon^2}{4}}}{40} \sum_{i=1}^{n} \mathbb{E}\left[\xi_i^2 I_{\{|\xi_i|>\epsilon\}}\right] \\
&= \frac{1-e^{-\frac{\epsilon^2}{4}}}{40} \cdot \frac{(1-\alpha)^2\alpha|E^c| + (1-\beta)^2\beta|E|}{\alpha(1-\alpha)|E^c| + \beta(1-\beta)|E|} \\
&= \frac{1-e^{-\frac{\epsilon^2}{4}}}{40} \cdot \frac{(1-\alpha)^2 + (1-\beta)^2}{2-(\alpha+\beta)} \\
&= e^{-\frac{\epsilon^2}{4}} \frac{e^{\frac{\epsilon^2}{4}}-1}{40} \cdot \frac{(1-\alpha)^2 + (1-\beta)^2}{2-(\alpha+\beta)} \\
&\geq e^{-\frac{\epsilon^2}{4}} \frac{\epsilon^2}{160} \cdot \frac{(1-\alpha)^2 + (1-\beta)^2}{2-(\alpha+\beta)} \\
&= e^{-\frac{\epsilon^2}{4}} \frac{1}{640} \cdot \frac{1}{\alpha(1-\alpha)|E^c| + \beta(1-\beta)|E|} \cdot \frac{(1-\alpha)^2 + (1-\beta)^2}{2-(\alpha+\beta)} \\
&= e^{-\frac{\epsilon^2}{4}} \frac{1}{640} \cdot \frac{1}{\alpha|E^c|} \cdot \frac{(1-\alpha)^2 + (1-\beta)^2}{(2-(\alpha+\beta))^2} \ .
\end{aligned}$$

Combining the two sets of expressions above, the lower bound becomes

$$\begin{aligned}
&\frac{1-e^{-\frac{\epsilon^2}{4}}}{40} \sum_{i=1}^{n} \mathbb{E}\left[\xi_i^2 I_{\{|\xi_i|>\epsilon\}}\right] - \sum_{i=1}^{n} \sigma_i^4 \\
\geq\ & e^{-\frac{\epsilon^2}{4}} \frac{1}{640} \cdot \frac{1}{\alpha|E^c|} \cdot \frac{(1-\alpha)^2 + (1-\beta)^2}{(2-(\alpha+\beta))^2} - \frac{1}{(2-(\alpha+\beta))^2}\left[\frac{(1-\alpha)^2}{|E^c|} + \frac{(1-\beta)^2}{|E|}\right] \\
=\ & \frac{1}{(2-(\alpha+\beta))^2}\left[(1-\alpha)^2\left(\frac{\exp\left(-\frac{1}{16}\cdot\frac{1}{\alpha|E^c|}\cdot\frac{1}{2-(\alpha+\beta)}\right)}{640} \cdot \frac{1}{\alpha|E^c|} - \frac{1}{|E^c|}\right)\right. \\
&\left. +(1-\beta)^2\left(\frac{\exp\left(-\frac{1}{16}\cdot\frac{1}{\alpha|E^c|}\cdot\frac{1}{2-(\alpha+\beta)}\right)}{640} \cdot \frac{1}{\alpha|E^c|} - \frac{1}{|E|}\right)\right] \ . \qquad (22)
\end{aligned}$$

But for sufficiently large n_v, the exponential term in (22) behaves like $\exp[-1/(16\lambda)] \approx 1 - (1/16\lambda)$. Substituting accordingly and simplifying to ignore the various terms tending to a constant in large n_v, the expression in (22) can be seen to behave asymptotically like

$$\frac{1}{4}\left[\left(\frac{1}{640\lambda} - \frac{1}{|E^c|}\right) + \left(\frac{1}{640\lambda} - \frac{1}{|E|}\right)\right] \ .$$

Again, by the assumption of low-rate measurement error in (A3) and the definitions of sparse and dense graphs given in Section 2, appropriate substitution of the values for λ, $|E|$, and $|E^c|$ yield the lower bounds in (10) and (11). This completes the proof of Theorem 7. ■

A.6 Proof of Theorem 8

Proof The proof follows by rewriting each of the two sums bracketed in (8), and then aggregating terms. Under condition (i) of the theorem,

$$\begin{aligned}\left|U - U_k^{(L)}\right| &= \left|L_k + \sum_{j\neq k} L_j - \sum_{\ell} M_\ell - \left(\sum_{j\neq k} \tilde{L}_j^{L_k} - \sum_{\ell} \tilde{M}_\ell^{L_k}\right)\right| \\ &= \sum_{j\neq k} \tilde{L}_j^{L_k} - \sum_{\ell} \tilde{M}_\ell^{L_k} - \left(\sum_{j\neq k} L_j - \sum_{\ell} M_\ell\right) - L_k \\ &= U_k^{(L)} - U \ .\end{aligned}$$

Similarly, under condition (ii) of the theorem, $\left|U - U_k^{(M)}\right| = U - U_k^{(M)}$.

In the absence of having to deal directly with the absolute values, we find that

$$\sum_{k=1}^{n} p_k E\left|U - U_k^{(L)}\right| = \sum_{k=1}^{n}\sum_{k\neq j} E\left[L_k L_j\right] - \sum_{k,\ell} E\left[L_k M_\ell\right] - E\left[U\right]\lambda_1$$

and

$$\sum_{k=1}^{m} q_k E\left|U - U_k^{(M)}\right| = \sum_{k=1}^{m}\sum_{\ell\neq k} E\left[M_k M_\ell\right] - \sum_{k,\ell} E\left[L_k M_\ell\right] + E\left[U\right]\lambda_2 \ .$$

As a result, the bracketed term in (8) takes the form

$$\begin{aligned}&\sum_{k=1}^{n} p_k E\left|U - U_k^{(L)}\right| + \sum_{k=1}^{m} q_k E\left|U - U_k^{(M)}\right| \\ =\ & \sum_{k=1}^{n}\sum_{k\neq j} E\left[L_k L_j\right] + \sum_{k=1}^{m}\sum_{\ell\neq k} E\left[M_k M_\ell\right] - 2\sum_{k,\ell} E\left[L_k M_\ell\right] - \left(E\left[U\right]\right)^2 \\ =\ & E\left[U^2\right] - \left(E\left[U\right]\right)^2 - \sum_{k=1}^{n} E\left[L_k^2\right] - \sum_{\ell=1}^{m} E\left[M_\ell^2\right] \\ =\ & \mathrm{Var}(U) - \sum_{k=1}^{n} E\left[L_k\right] - \sum_{\ell=1}^{m} E\left[M_\ell\right] \\ =\ & \mathrm{Var}(U) - (\lambda_1 + \lambda_2) \\ =\ & \mathrm{Var}(U) - \mathrm{Var}(W) \ .\end{aligned}$$

■

References

Milton Abramowitz and Irene A Stegun. *Handbook of Mathematical Functions: with Formulas, Graphs, and Mathematical Tables*. Number 55. Courier Dover Publications, 1972.

Nesreen K Ahmed, Jennifer Neville, and Ramana Kompella. Network sampling: from static to streaming graphs. *ACM Transactions on Knowledge Discovery from Data*, 8:1–56, 2014.

Zack W. Almquist. Random errors in egocentric networks. *Social Networks*, 34(4):493–505, 2012.

Prakash Balachandran, Edoardo Airoldi, and Eric Kolaczyk. Inference of network summary statistics through network denoising. *arXiv preprint arXiv:1310.0423*, 2013a.

Prakash Balachandran, Weston D. Viles, and Eric D Kolaczyk. Exponential-type inequalities involving ratios of the modified Bessel function of the first kind and their applications. arxiv:1311.1450, 2013b.

Andrew D Barbour and Louis Hsiao Yun Chen. *An Introduction to Stein's Method*, volume 4. World Scientific, 2005.

T. Tony Cai, Tengyuan Liang, and Alexander Rakhlin. On detection and structural reconstruction of small-world random networks. *IEEE Transactions on Network Science and Engineering*, 2017.

Sourav Chatterjee. Matrix estimation by universal singular value thresholding. *Annals of Statistics*, 43(1):177–214, 2015.

Bill Cheswick, Hal Burch, and Steve Branigan. Mapping and visualizing the internet. In *USENIX Annual Technical Conference, General Track*, pages 1–12. Citeseer, 2000.

Elissa J. Cosgrove, Timothy S. Gardner, and Eric D. Kolaczyk. On the choice and number of microarrays for transcriptional regulatory network inference. *BMC bioinformatics*, 11(1):454, 2010.

Rick Durrett. *Probability: Theory and Examples*. Cambridge University Press, 2010.

Bradley Efron. *Large-scale Inference: Empirical Bayes Methods for Estimation, Testing, and Prediction*, volume 1. Cambridge University Press, 2012.

James D Esary, Frank Proschan, David W Walkup, et al. Association of random variables, with applications. *The Annals of Mathematical Statistics*, 38(5):1466–1474, 1967.

Christopher Fogelberg and Vasile Palade. Machine learning and genetic regulatory networks: a review and a roadmap. In *Foundations of Computational, Intelligence Volume 1*, pages 3–34. Springer, 2009.

Karl J Friston. Functional and effective connectivity: a review. *Brain Connectivity*, 1(1):13–36, 2011.

Wayne A Fuller. *Measurement Error Models*, volume 305. John Wiley & Sons, 2009.

Hyeouk Chris Hahm, Eric Kolaczyk, Jisun Jang, Theadora Swenson, and Asma Moiz Bhindarwala. Binge drinking trajectories from adolescence to young adulthood: the effects of peer social network. *Substance use & misuse*, 47(6):745–756, 2012.

Derek Hansen, Ben Shneiderman, and Marc A Smith. *Analyzing Social Media Networks with NodeXL: Insights from a Connected World*. Morgan Kaufmann, 2010.

G. Traver Hart, Arun K. Ramani, and Edward M. Marcotte. How complete are current yeast and human protein-interaction networks? *Genome Biology*, 7(11):1, 2006.

Shawndra Hill, Foster Provost, and Chris Volinsky. Network-based marketing: Identifying likely adopters via consumer networks. *Statistical Science*, pages 256–276, 2006.

Matthew O Jackson. *Social and Economic Networks*. Princeton University Press, 2010.

Svante Janson, Tomasz Luczak, and Andrzej Rucinski. *Random Graphs*, volume 45. John Wiley & Sons, 2011.

Xiaoyu Jiang and Eric D. Kolaczyk. A latent eigenprobit model with link uncertainty for prediction of protein–protein interactions. *Statistics in Biosciences*, 4(1):84–104, 2012.

Xiaoyu Jiang, David Gold, and Eric D. Kolaczyk. Network-based auto-probit modeling for protein function prediction. *Biometrics*, 67(3):958–966, 2011.

Joseph B Kadane. Sums of possibly associated Bernoulli variables: The Conway-Maxwell-binomial distribution. *arXiv preprint arXiv:1404.1856*, 2014.

Eric D Kolaczyk. *Statistical Analysis of Network Data*. Springer, 2009.

Dean Lusher, Johan Koskinen, and Garry Robins. *Exponential Random Graph Models for Social Networks: Theory, Methods, and Applications*. Cambridge University Press, 2012.

Nicolai Meinshausen and Peter Bühlmann. High-dimensional graphs and variable selection with the lasso. *The Annals of Statistics*, pages 1436–1462, 2006.

Mark Newman. *Networks: an Introduction*. Oxford University Press, 2010.

Matey Neykov, Junwei Lu, and Han Liu. Combinatorial inference for graphical models. *arxiv preprint arXiv:1608.03045*, 2016.

Carey E. Priebe, Daniel L. Sussman, Minh Tang, and Joshua T. Vogelstein. Statistical inference on errorfully observed graphs. *Journal of Computational and Graphical Statistics*, 24(4):930–953, 2015.

John G Skellam. The frequency distribution of the difference between two Poisson variates belonging to different populations. *Journal of the Royal Statistical Society. Series A (General)*, 109(Pt 3):296, 1946.

Journal of Machine Learning Research 18 (2017) 1-58 Submitted 10/16; Revised 4/17; Published 6/17

Dense Distributions from Sparse Samples: Improved Gibbs Sampling Parameter Estimators for LDA

Yannis Papanikolaou YPAPANIK@CSD.AUTH.GR
Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki, Greece

James R. Foulds JFOULDS@UCSD.EDU
California Institute for Telecommunications and Information Technology
University of California
San Diego, CA, USA

Timothy N. Rubin TIM.RUBIN@GMAIL.COM
SurveyMonkey
San Mateo, CA, USA

Grigorios Tsoumakas GREG@CSD.AUTH.GR
Department of Informatics
Aristotle University of Thessaloniki
Thessaloniki, Greece

Editor: David Blei

Abstract

We introduce a novel approach for estimating Latent Dirichlet Allocation (LDA) parameters from collapsed Gibbs samples (CGS), by leveraging the full conditional distributions over the latent variable assignments to efficiently average over multiple samples, for little more computational cost than drawing a single additional collapsed Gibbs sample. Our approach can be understood as adapting the soft clustering methodology of Collapsed Variational Bayes (CVB0) to CGS parameter estimation, in order to get the best of both techniques. Our estimators can straightforwardly be applied to the output of any existing implementation of CGS, including modern accelerated variants. We perform extensive empirical comparisons of our estimators with those of standard collapsed inference algorithms on real-world data for both unsupervised LDA and Prior-LDA, a supervised variant of LDA for multi-label classification. Our results show a consistent advantage of our approach over traditional CGS under all experimental conditions, and over CVB0 inference in the majority of conditions. More broadly, our results highlight the importance of averaging over multiple samples in LDA parameter estimation, and the use of efficient computational techniques to do so.

Keywords: Latent Dirichlet Allocation, topic models, unsupervised learning, multi-label classification, text mining, collapsed Gibbs sampling, CVB0, Bayesian inference

©2017 Papanikolaou, Foulds, Rubin and Tsoumakas.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-526.html.

1. Introduction

Latent Dirichlet Allocation (LDA) (Blei et al., 2003) is a probabilistic model which organizes and summarizes large corpora of text documents by automatically discovering the semantic themes, or *topics*, hidden within the data. Since the model was introduced by Blei et al. (2003), LDA and its extensions have been successfully applied to many other data types and application domains, including bioinformatics (Zheng et al., 2006), computer vision (Cao and Fei-Fei, 2007), and social network analysis (Zhang et al., 2007), in addition to text mining and analytics (AlSumait et al., 2008). It has also been the subject of numerous adaptations and improvements, for example to deal with supervised learning tasks (Blei and McAuliffe, 2007; Ramage et al., 2009; Zhu et al., 2009), time and memory efficiency issues (Newman et al., 2007; Porteous et al., 2008; Yao et al., 2009) and large or streaming data settings (Gohr et al., 2009; Hoffman et al., 2010b; Rubin et al., 2012; Foulds et al., 2013).

When training an LDA model on a collection of documents, two sets of parameters are primarily estimated: the *distributions over word types* ϕ for topics, and the *distributions over topics* θ for documents. After more than a decade of research on LDA training algorithms, the Collapsed Gibbs Sampler (CGS) (Griffiths and Steyvers, 2004) remains a popular choice for topic model estimation, and is arguably the de facto industry standard technique for commercial and industrial applications. Its success is due in part to its simplicity, along with the availability of efficient implementations (McCallum, 2002a; Smola and Narayanamurthy, 2010) leveraging sparsity (Yao et al., 2009; Li et al., 2014) and parallel architectures (Newman et al., 2007). It exhibits much faster convergence than the naive uncollapsed Gibbs sampler of Pritchard et al. (2000), as well as the partially collapsed Gibbs sampling scheme (Asuncion, 2011). The CGS algorithm marginalizes out parameters ϕ and θ, necessitating a procedure to recover them from samples of the model's latent variables $\mathbf{z}$. Griffiths and Steyvers (2004) proposed such a procedure, which, while very successful, has not been revisited in the last decade to our knowledge. Our goal is to improve upon this procedure.

In this paper, we propose a new method for estimating topic model parameters ϕ and θ from collapsed Gibbs samples. Our approach approximates the posterior mean of the parameters, which leads to increased stability and statistical efficiency relative to the standard estimator. Crucially, our approach leverages the *distributional information given by the form of the Gibbs update equation* to implicitly average over multiple Markov chain Monte Carlo (MCMC) samples, with little more computational cost than that which is required to compute a single sample. As such, our work is focused on the realistic practical scenario where we can afford a moderate burn-in period, but cannot afford to compute and average over enough post burn-in MCMC samples to accurately estimate the parameters' posterior mean. In our experiments, this requires several orders of magnitude further sampling iterations beyond burn-in.

Our use of the full conditional distribution of each topic assignment is reminiscent of the CVB0 algorithm for collapsed variational inference (Asuncion et al., 2009). The update equations of CVB0 bear resemblance to those of CGS, except that they involve deterministic updates of dense probability distributions, while CGS draws sparse samples from similar distributions, with corresponding trade-offs in execution time and convergence behavior, as we study in Section 5.5. Our approach can be understood as adapting this dense *soft clustering* methodology of the CVB0 algorithm to CGS parameter estimation, leveraging

	training phase	parameter recovery	update rule	inference algorithm	memory per token
CVB0	dense	dense	deterministic	variational Bayes	$O(K)$
CGS	sparse	sparse	random	MCMC	$O(1)$
CGS_p	sparse	dense	random	MCMC	$O(1)$

Table 1: Properties of CVB0, CGS, and our proposed CGS_p method.

the corresponding uncertainty information, as in CVB0, but within a Markov chain Monte Carlo framework, in order to draw on the benefits of both techniques. Our approach does not incur the memory overhead of CVB0, relative to CGS. The properties of CVB0, CGS, and our proposed estimators are summarized in Table 1.

It is important to note that we do not modify the CGS algorithm, and thus do not affect its runtime or memory requirements. Rather, we modify the *procedure for recovering parameter estimates from collapsed Gibbs samples*. After running the standard CGS algorithm (or a modern accelerated approximate implementation) for a sufficient number of iterations, we obtain the standard document-topic and topic-word count matrices. At this point, instead of using the standard CGS parameter estimation equations to calculate θ and ϕ, we employ our proposed estimators. As a result, our estimators can be plugged in to the output of any CGS-based inference method, including modern variants of the algorithm such as Sparse LDA (Yao et al., 2009), Metropolis-Hastings-Walker (Li et al., 2014), LightLDA (Yuan et al., 2015), or WarpLDA (Chen et al., 2016), allowing for easy and wide adoption of our approach by the research community and in industry applications. Our algorithm is very simple to implement, requiring only slight modifications to existing CGS code. Popular LDA software packages such as MALLET (McCallum, 2002a) and Yahoo_LDA (Smola and Narayanamurthy, 2010) could straightforwardly be extended to incorporate our methods, leading to improved parameter estimation in numerous practical topic modeling applications. In Section 5.7, we provide an application of our estimators to both MALLET's Sparse LDA implementation, and to WarpLDA. When paired with a sparse CGS implementation, our approach gives the best of two worlds: making use of the *full uncertainty information encoded in the dense Gibbs sampling transition probabilities* in the final parameter recovery step, while still leveraging *sparsity to accelerate the CGS training process*.

In extensive experiments, we show that our approach leads to improved performance over standard CGS parameter estimators in both unsupervised and supervised LDA models. Our approach also outperforms the CVB0 algorithm under the majority of experimental conditions. The computational cost of our method is comparable that of a single (dense) CGS iteration, and we show that even when a state-of-the-art sparse implementation of the CGS training algorithm is used, this is a price well worth paying. Our results further illustrate the benefits of averaging over multiple samples for LDA parameter estimation. The contributions of our work can be summarized as follows:

- We propose a novel, theoretically motivated method for improved estimation of the topic-level and document-level parameters of an LDA model based on collapsed Gibbs samples.

- We present an extensive empirical comparison of our proposed estimators against those of standard CGS as well as the CVB0 algorithm (Asuncion et al., 2009) —a variant of the Collapsed Variational Bayesian inference (CVB) method—in both unsupervised and supervised learning settings, demonstrating the benefits of our approach.

- We provide an additional experimental comparison of the CGS and CVB0 algorithms regarding their convergence behavior, to further contextualize our empirical results.

Our theoretical and experimental results lead to three primary conclusions of interest to topic modeling practitioners:

- Averaging over multiple CGS samples to construct a point estimate for LDA parameters is beneficial to performance, in both the unsupervised and supervised settings, in cases where identifiability issues can be safely resolved.

- Using CVB0-style soft clustering to construct these point estimates is both valid and useful in an MCMC setting, and corresponds to implicitly averaging over many samples, thereby improving performance with little extra computational cost. This ports some of the benefits of CVB0's use of dense uncertainty information to CGS, while retaining the sparsity advantages of that algorithm during training.

- Increasing the number of topics increases the benefit of our soft clustering/averaging approach over traditional CGS. While CVB0 outperforms CGS with few topics and a moderate number of iterations, CGS-based methods otherwise outperform CVB0, especially when using our improved estimators.

As this paper deals with LDA in both the unsupervised and the supervised setting, we will treat the following terms as equivalent when describing the procedure of learning an LDA model from training data: 'estimation', 'fitting', and 'training'. Since we are in a Bayesian setting the word 'inference' will refer to the recovery of parameters, which are understood to be random variables, and/or the latent variables, and to the computation of the posterior distributions of parameters and latent variables. The term 'prediction' will be applied to the case of predicting on test data (i.e., data that was unobserved during training). Additionally, since Prior-LDA is essentially a special case of unsupervised LDA, we will focus our algorithm descriptions on the case of standard unsupervised LDA and specify the exceptions as they apply to Prior-LDA. Furthermore, we note here that we will be using the CGS algorithm described by Griffiths and Steyvers (2004), which approximates the LDA parameters of interest through iterative sampling in a Markov chain Monte Carlo (MCMC) procedure, unless otherwise specified.

The rest of this paper is organized as follows: Section 2 describes backround information on the LDA model, Bayesian inference, and the CGS and the CVB0 inference methods, as well as the Prior-LDA model, providing an overview of the related literature. Sections 3 and 4 introduce our estimators for the relevant LDA parameters. Sections 5 and 6 describe our experiments comparing the estimators of our method to those of CGS and CVB0 in unsupervised and supervised learning settings, respectively. Finally, Section 7 summarizes the main contributions and conclusions of our work.

2. Background and Related Work

In this section we provide the background necessary to describe our methods, and also establish the relevant notation.

2.1 Latent Dirichlet Allocation

LDA is a hierarchical Bayesian model that describes how a corpus of documents is generated via a set of unobserved *topics*. Each topic is parameterized by a discrete probability distribution over word types, which captures some coherent semantic theme within the corpus[1]. Here, we assume that this set of word types is the same for all topics (i.e., we have a single, shared vocabulary). Furthermore, LDA assumes that each document can be described as a discrete distribution over these topics. The process for generating each document then involves first sampling a topic from the document's distribution over topics, and then sampling a word token from the corresponding topic's distribution over word types.

Formally, let us denote as V the size of the vocabulary (the set of unique word types) and as $D_{Train} = |\mathcal{D}_{Train}|$ and $D_{Test} = |\mathcal{D}_{Test}|$ the number of training and testing documents respectively. We will use K to denote the number of topics. Similarly, v denotes a word type, w_i a word token, k a topic (or in the case of Prior-LDA, a label), and z_i a topic assignment. The parameter ϕ_{kv}, $k \in \{1...K\}$, $v \in \{1...V\}$ represents the probability of word type v under topic (or label) k, and θ_{dk} represents the probability of topic (or label) k for document d. Additionally α_k will denote the parameter of the Dirichlet prior on θ for topic k and β_v the parameter of the Dirichlet prior on ϕ for word type v. Unless otherwise noted we assume for simplicity that the two priors are symmetric, so all the elements of the parameter vector $\boldsymbol{\alpha}$ will have the same value (the same holds for the parameter vector $\boldsymbol{\beta}$). The term N_d denotes the number of word tokens in a given document d. Lastly, in the multi-label setting, $\mathcal{K}_d$ stands for the set of labels that are associated with d. This notation is summarized in Table 2.

LDA assumes that a given corpus of documents has been generated as follows:

- For each topic $k \in \{1 \dots K\}$, sample a discrete distribution ϕ_k over the word types of the corpus from a Dirichlet($\boldsymbol{\beta}$).
- For each document d,
  - Sample a discrete distribution θ_d over topics from a Dirichlet($\boldsymbol{\alpha}$).
  - Sample a document length N_d from a Poisson(ξ), with $\xi \in \mathbb{R}_{>0}$
  - For each word token w_i in document d, $i \in \{1...N_d\}$:
    - Sample a topic $z_i = k$ from discrete(θ_d).
    - Sample a word type $w_i = v$ from discrete(ϕ_{z_i}).

The goal of inference is to estimate the aforementioned θ and ϕ parameters—the discrete distributions for documents over topics, and topics over word types, respectively. In doing so, we learn a lower-dimensional representation of the structure of all documents in the corpus in terms of the topics. In the unsupervised learning context, this lower-dimensional

1. A discrete distribution a multinomial distribution for which a single value is drawn, i.e., where $N = 1$.

V	the size of the vocabulary
K	the number of topics (or labels)
D_{Train}	the number of training documents (similarly D_{Test})
$\mathcal{D}_{Train}$	the set of training documents (similarly $\mathcal{D}_{Test}$)
d	a document
v	a word type
w_i	a single word token, i.e., an instance of a word type
$\mathbf{w}_d$	the vector of word assignments in document d
k	a topic (label)
z_i	the topic assignment to a word token w_i
$\mathbf{z}_d$	the vector of topic assignments to all word tokens of a document
z_{djk}	binary indicator variable, equals 1 IFF the jth word in document d is assigned to topic k
N_d	number of word tokens in d
$\mathcal{K}_d$	set of topics (or labels) in d
n_{kv}	number of times that word type v has been assigned to topic k across the corpus
n_{dk}	number of word tokens in d that have been assigned to topic k
ϕ_k	word type distribution for topic k
θ_d	topic distribution for document d
ϕ_{kv}	topic-word type probability
θ_{dk}	document-topic probability
α_k	Dirichlet prior hyperparameter on θ for topic k
β_v	Dirichlet prior hyperparameter on ϕ for word type v
γ_{dik}	CVB0 variational probability of word token w_i in document d being assigned to topic k

Table 2: Notation used throughout the article.

representation of documents is useful for both summarizing documents and for generating predictions about future, unobserved documents. In the supervised, multi-label learning context, extensions of the basic LDA model—such as Prior-LDA —put topics into one-to-one correspondence with labels, and the model is used for assigning labels to test documents.

2.2 Bayesian Inference, Prediction, and Parameter Estimation

LDA models are typically trained using Bayesian inference techniques. Given the observed training documents $\mathcal{D}_{Train}$, the goal of Bayesian inference is to "invert" the assumed generative process of the model, going "backwards" from the data to recover the parameters, by inferring the posterior distribution over them. In the case of LDA, the posterior distribution over parameters and latent variables is given by Bayes' rule as

$$p(\phi, \theta, \mathbf{z}|\mathcal{D}_{Train}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{p(\mathcal{D}_{Train}|\phi, \theta, \mathbf{z}) p(\phi, \theta, \mathbf{z}|\boldsymbol{\alpha}, \boldsymbol{\beta})}{p(\mathcal{D}_{Train}|\boldsymbol{\alpha}, \boldsymbol{\beta})} \ .$$

Being able to make use of the uncertainty encoded in the posterior distribution is a key benefit of the Bayesian approach. In the ideal procedure, having computed the posterior it can be used to make predictions on held-out data via the posterior predictive distribution, which computes the probability of new data by averaging over the parameters, weighted by their posterior probabilities. For a test document $\mathbf{w}^{(new)}$, under the LDA model we have

$$p(\mathbf{w}^{(new)}|\mathcal{D}_{Train}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \int\int p(\mathbf{w}^{(new)}|\phi, \theta^{(new)}) p(\theta^{(new)}|\boldsymbol{\alpha}) d\theta^{(new)} p(\phi|\mathcal{D}_{Train}, \boldsymbol{\alpha}, \boldsymbol{\beta}) d\phi \ .$$

Similarly, for supervised variants of LDA (discussed below), the ideal procedure is to average over all possible parameter values when predicting labels of new documents. In practice, however, it is intractable to compute or marginalize over the posterior, necessitating approximate inference techniques such as MCMC methods, which sample from the posterior using an appropriate Markov chain, and hence approximate it with the resulting set of samples. In particular, Nguyen et al. (2014) advocate using multiple samples to approximate the posterior, and averaging over the corresponding approximation to the posterior predictive distribution in order to make predictions for new data.

However, in the LDA literature it is much more common practice to simply approximate the posterior (and hence posterior predictive) distribution based on a single estimated value of the parameters (a *point estimate*). Although not ideal from a Bayesian perspective, a point estimate $\hat{\phi}$, $\hat{\theta}$ of the parameters is convenient to work with as it is much easier for a human analyst to interpret, and is computationally much cheaper to use at test time when making predictions than using a collection of samples. In this paper, while strongly agreeing with the posterior predictive-based approach of Nguyen et al. (2014) in principle, we take the perspective that a point estimate is in many cases useful and desirable to have.

Following Nguyen et al. (2014), we do, however, have substantial reservations regarding the ubiquitous standard practice of using *a single MCMC sample* to obtain a point estimate. By interpreting MCMC as a "stochastic mode-finding algorithm" we can view this procedure as a poor-man's approximation to the mode of the posterior. For many models, with a sufficiently large amount of data the posterior will approach a multivariate Gaussian under the Bernstein-von Mises theorem, and eventually become concentrated at a single point under certain general regularity conditions, cf. (Kass et al., 1990; Kleijn and van der Vaart, 2012). In this regime, a point estimate based on a single sample will in fact suffice. More generally, however, due to posterior uncertainty this procedure is unstable and consequently statistically inefficient: a parameter estimate from a posterior sample has an *asymptotic relative efficiency (ARE) of 2*, meaning, roughly speaking, that the variance of the estimator requires twice as many observations to match the variance of the ideal estimator, in the asymptotic Gaussian regime, under general regularity conditions, cf. (Wang et al., 2015).

As a compromise between the expensive full Bayesian posterior estimation procedure and noisy "stochastic mode-finding" with one sample, in this paper we propose to use the *posterior mean*, approximated via multiple MCMC samples, as an estimator of choice in many cases. The posterior mean, if it can be precisely computed, is asymptotically efficient (i.e., it has an ARE of 1), under the conditions necessary for the Bernstein-von Mises theorem to hold, cf. (Kleijn and van der Vaart, 2012). Although we cannot compute the posterior mean exactly, and the conditions necessary for the Bernstein-von Mises theorem have not been rigorously verified for LDA, these results provide a useful intuition for why we should

Algorithm 1 Pseudocode for our proposed estimators. For simplicity, we consider only the case where a single CGS sample $\mathbf{z}$ is used.

function θ^p
Input: Burned-in CGS sample $\mathbf{z}$ with associated count matrices, documents $\mathcal{D}$
for $d = 1$ to D **do**
$\hat{\theta}^p_{d,:} := \alpha^\intercal$
for $j = 1$ to N_d **do**
$v := w_{dj}$ ▷ Retrieve the word token from the word position j in d.
for $k = 1$ to K **do**
$p_{djk} := \frac{n_{kv\neg dj} + \beta_v}{\sum_{v'=1}^{V} (n_{kv'\neg dj} + \beta_{v'})} \cdot (n_{dk\neg dj} + \alpha_k)$ ▷ During testing, the first term is replaced by ϕ_{kv}
end for
$p_{dj,:} := p_{dj,:} \ ./ \ \text{sum}(p_{dj,:})$ ▷ Normalize such that $\sum_{k=1}^{K} p_{djk} = 1$, ./ denotes elementwise division.
$\hat{\theta}^p_{d,:} := \hat{\theta}^p_{d,:} + p^\intercal_{dv,:}$
end for
$\hat{\theta}^p_{d,:} := \hat{\theta}^p_{d,:} \ ./ \ \text{sum}(\hat{\theta}^p_{d,:})$ ▷ Normalize $\hat{\theta}^p_{d,:}$ such that $\sum_{k=1}^{K} \hat{\theta}^p_{dk} = 1$
end for
return $\hat{\theta}^p$
end function

function ϕ^p
Input: Burned-in CGS sample $\mathbf{z}$ with associated count matrices, documents $\mathcal{D}_{Train}$
for $k = 1$ to K **do**
$\hat{\phi}^p_{k,:} := \beta^\intercal$
end for
for $d = 1$ to D **do**
for $j = 1$ to N_d **do**
$v := w_{dj}$
for $k = 1$ to K **do**
$p_{djk} = \frac{n_{kv\neg dj} + \beta_v}{\sum_{v'=1}^{V} (n_{kv'\neg dj} + \beta_{v'})} \cdot (n_{dk\neg dj} + \alpha_k)$
end for
$p_{dj,:} := p_{dj,:} \ ./ \ \text{sum}(p_{dj,:})$ ▷ Normalize such that $\sum_{k=1}^{K} p_{djk} = 1$.
$\hat{\phi}^p_{:,v} := \hat{\phi}^p_{:,v} + p_{dj,:}$
end for
end for
for $k = 1$ to K **do**
$\hat{\phi}^p_k := \hat{\phi}^p_{k,:} \ ./ \ \text{sum}(\hat{\phi}^p_{k,:})$ ▷ Normalize $\hat{\phi}^p_{k,:}$ such that $\sum_{v=1}^{V} \hat{\phi}^p_{kv} = 1$
end for
return $\hat{\phi}^p$
end function

prefer this approach over an estimator based on a single sample. We expect a sample-based approximation to the posterior mean to be much more stable than an estimator based on a single sample, and our empirical results below will show that this is indeed the case in practice. In our proposed algorithm, we will show how to effectively average over multiple samples almost "for free", based on samples and their Gibbs sampling transition probabilities. This is valuable in the very common setting where we are able to afford a moderate number of burn-in MCMC iterations, but we are not able to afford the much larger number of iterations to compute enough samples required to accurately estimate the posterior mean of the parameters.

It should be noted that averaging over MCMC samples for LDA, without care, can be invalidated due to a lack of *identifiability*. Multiple samples can each potentially encode different permutations of the topics, thereby destroying any correspondences between them, and Griffiths and Steyvers (2004) strongly warn against this. However, in this work, we will identify certain cases for which it is safe, and indeed highly beneficial, to average over samples in order to construct a point estimate, such as when estimating θ at test time with the topics held fixed.

2.3 Collapsed Gibbs Sampling for LDA

The Collapsed Gibbs Sampling (CGS) algorithm for LDA, introduced by Griffiths and Steyvers (2004), marginalizes out the parameters ϕ and θ, and operates only on the latent variable assignments $\mathbf{z}$. This leads to a simple but effective MCMC algorithm which mixes much more quickly than the naive Gibbs sampling algorithm. To compute the CGS updates efficiently, the algorithm maintains and makes use of several count matrices during sampling. We will employ the following notation for these count matrices: n_{kv} represents the number of times that word type v is assigned to topic k across the corpus, and n_{dk} represents the number of word tokens in document d that have been assigned to topic k. The notation for these count matrices is included in Table 2.

During sampling, CGS updates the hard assignment z_i of a word token w_i to one of the topics $k \in \{1...K\}$. This update is performed sequentially for all word tokens in the corpus, for a fixed number of iterations. The update equation giving the probability of setting z_i to topic k, conditional on w_i, d, the hyperparameters $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, and the current topic assignments of all other word tokens (represented by $\cdot$) is:

$$p(z_i = k | \, w_i = v, \, d, \, \boldsymbol{\alpha}, \, \boldsymbol{\beta}, \, \cdot) \propto \frac{n_{kv\neg i} + \beta_v}{\sum_{v'=1}^{V} (n_{kv'\neg i} + \beta_{v'})} \cdot \frac{n_{dk\neg i} + \alpha_k}{N_d + \sum_{k'=1}^{K} \alpha_{k'}} \, . \tag{1}$$

In the above equation, one excludes from all count matrices the current topic assignment of w_i, as indicated by the $\neg i$ notation. A common practice is to run the Gibbs sampler for a number of iterations before retrieving estimates for the parameters, as the Markov chain is typically in a low probability state initially and so early estimates are expected to be of poor quality. After this *burn-in* period, we can retrieve estimates for both the topic assignments

z_i as well as for the θ and ϕ parameters. Samples are taken at a regular interval, often called *sampling lag*.[2]

To compute point estimates for θ and ϕ, Rao-Blackwell estimators were employed in (Griffiths and Steyvers, 2004). Specifically, a point estimate of the probability of word type v given topic k is computed as:

$$\hat{\phi}_{kv} = \frac{n_{kv} + \beta_v}{\sum_{v'=1}^{V} (n_{kv'} + \beta_{v'})} \, . \tag{2}$$

Similarly, a point estimate for the probability of the topic k given document d is given by:

$$\hat{\theta}_{dk} = \frac{n_{dk} + \alpha_k}{N_d + \sum_{k'=1}^{K} \alpha_{k'}} \, . \tag{3}$$

During prediction (i.e., when applying CGS on documents that were unobserved during training), a common practice is to fix the ϕ distributions and set them to the ones learned during estimation. The sampling update equation presented in Equation 1 thus becomes:

$$p(z_i = k | \, w_i = v, \, d, \, \boldsymbol{\alpha}, \, \hat{\phi}, \, \cdot) \propto \hat{\phi}_{kv} \cdot \frac{n_{dk\neg i} + \alpha_k}{N_d + \sum_{k'=1}^{K} \alpha_{k'}} \, . \tag{4}$$

2.4 Labeled LDA

An extension to unsupervised LDA for supervised multi-label document classification was proposed by Ramage et al. (2009). Their algorithm, Labeled LDA (LLDA), employs a one-to-one correspondence between topics and labels. During estimation of the model, the possible assignments for a word token to a topic are constrained to the training document's observed labels. Therefore, during training, the sampling update of Equation 1, becomes:

$$p(z_i = k | \, w_i = v, \, d, \, \boldsymbol{\alpha}, \, \boldsymbol{\beta}, \, \cdot) \propto \begin{cases} \frac{n_{kv\neg i} + \beta_v}{\sum_{v'=1}^{V} (n_{kv'\neg i} + \beta_{v'})} \cdot \frac{n_{dk\neg i} + \alpha_k}{N_d + \sum_{k'=1}^{K} \alpha_{k'}}, & \text{if } k \in \mathcal{K}_d \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Inference on test documents is performed similarly to standard LDA; estimates of the label–word types distributions, ϕ, are learned on the training data and are fixed, and then the test documents' θ distributions are estimated using Equation 3. Unlike in unsupervised LDA, where topics can change from iteration to iteration (the label switching problem), in LLDA topics remain steady (anchored to a label) and therefore it is possible to average point estimates of ϕ and θ over multiple Markov chains, thereby improving performance.

Ramage et al. (2011) have introduced PLDA to relax the constraints of LLDA and exploit the unsupervised and supervised forms of LDA simultaneously. Their algorithm attempts to model hidden topics within each label, as well as unlabeled, corpus-wide latent topics.

2. Averaging over dependent samples does not introduce any bias. Using a sampling lag (a.k.a. "thinning") is merely convenient for reducing memory requirements and computation at test time.

Rubin et al. (2012) presented two extensions of the LLDA model. The first one, Prior-LDA, takes into account the label frequencies in the corpus via an informative Dirichlet prior over parameter θ (we describe this process in detail in Section 6.3). The second extension, Dependency LDA, takes into account label dependencies by following a two stage approach: first, an unsupervised LDA model is trained on the observed labels. The estimated θ' parameters incorporate information about the label dependencies. Second, the LLDA model is trained as described in the previous paragraph. During prediction, the previously estimated θ' parameters of the unsupervised LDA model are used to calculate an asymmetrical hyperparameter α_{dk}, which is in turn used to compute the θ parameters of the LLDA model.

2.5 Collapsed Variational Bayes with Zero Order Information – CVB0

Collapsed Variational Bayesian (CVB) inference (Teh et al., 2006) is a deterministic inference technique for LDA. The key idea is to improve on Variational Bayes (VB) by marginalizing out θ and ϕ as in a collapsed Gibbs sampler. The key parameters in CVB are the γ_{di} variational distributions over the topics, with γ_{dik} representing the probability of the assignment of topic k to the word token w_i in document d, under the variational distribution. As the exact implementation of CVB is computationally expensive, the authors propose a second-order Taylor expansion as an approximation to compute the parameters of interest, which nevertheless improves over VB inference with respect to prediction performance.

Asuncion et al. (2009) presented a further approximation for CVB, by using a zero order Taylor expansion approximation.[3] The update equation for CVB0 is given by:

$$\gamma_{dik} \propto \frac{n'_{kw_i\neg i} + \beta_{w_i}}{\sum_{v'=1}^{V} (n'_{kv'\neg i} + \beta_{v'})} (n'_{dk\neg i} + \alpha_k) \tag{6}$$

with $n'_{kv} = \sum_{d=1}^{D} \sum_{j:w_{dj}=v} \gamma_{djk}$, $n'_{dk} = \sum_{j=1}^{N_d} \gamma_{djk}$.

Point estimates for θ and ϕ are retrieved from the same equations as in CGS (Equations 2 and 3). Even though the above update equation looks very similar to the one for CGS, the two algorithms have a couple of key differences. First, the counts n'_{kv} and n'_{dk} for CVB0 differ from the respective ones for CGS; the former are summations over the variational probabilities while the latter sum over the topic assignments z_i of topics to words. Second, CVB0 differs from CGS in that in every pass, instead of probabilistically sampling a hard topic-assignment for every token based on the sampling distribution in Equation 1, it keeps (and updates) that probability distribution for every word token. A consequence of this procedure is that CVB0 is a deterministic algorithm, whereas CGS is a stochastic algorithm (Asuncion, 2010). The deterministic nature of CVB0 allows it to converge faster than other inference techniques.

On the other hand, the CVB0 algorithm has significantly larger memory requirements as it needs to store the variational distribution γ for every token in the corpus. More recently, a stochastic extension of CVB0, SCVB0, was presented by Foulds et al. (2013), inspired by

3. The first-order terms are zero, so this is actually equivalent to a first-order Taylor approximation.

the Stochastic Variational Bayes (SVB) algorithm of Hoffman et al. (2010a). Both SCVB0 and SVB focus on efficient and fast inference of the LDA parameters in massive-scale data scenarios. SCVB0 also improves the memory requirements of CVB0. Since CVB0 maintains full, dense probability distributions for each word token, it is unable to leverage sparsity to accelerate the algorithm, unlike CGS (Yao et al., 2009; Li et al., 2014; Yuan et al., 2015; Chen et al., 2016). Inspired by CVB0, in our work we aim to leverage the full distributional uncertainty information per word token, in the context of the parameter estimation step of the CGS algorithm, while maintaining the valuable sparsity properties of that algorithm during the training process.

3. CGS$_p$ for Document-Topic Parameter Estimation

In this section we present our new estimator for the document-topic (θ_d) parameters of LDA that make use of the full distributions of word tokens over topics. For simplicity, we first describe the case of θ_d estimation during prediction and then extend our theory to θ_d estimation during training. We present our estimator for the topic-word parameters (ϕ_k) in the following section.

3.1 Standard CGS θ_d Estimator

The standard estimator for document d's parameters θ_d from a collapsed Gibbs sample $\mathbf{z}_d$ (Equation 3), due to Griffiths and Steyvers (2004), corresponds to the posterior predictive distribution over topic assignments, i.e.,

$$\begin{aligned}
\hat{\theta}_{dk} &\triangleq p(z^{(new)} = k|\mathbf{z}_d, \boldsymbol{\alpha}) && (7)\\
&= \int p(z^{(new)} = k, \theta_d|\mathbf{z}_d, \boldsymbol{\alpha})\ d\theta_d && (8)\\
&= \int p(z^{(new)} = k|\theta_d) p(\theta_d|\mathbf{z}_d, \boldsymbol{\alpha})\ d\theta_d && (9)\\
&= \int \theta_{dk} p(\theta_d|\mathbf{z}_d, \boldsymbol{\alpha})\ d\theta_d && (10)\\
&= \mathbb{E}_{p(\theta_d|\mathbf{z}_d,\boldsymbol{\alpha})}[\theta_{dk}] && (11)\\
&= \mathbb{E}_{\text{Dirichlet}(\theta_d|\boldsymbol{\alpha}+\mathbf{n}_d)}[\theta_{dk}] && (12)\\
&= \frac{n_{dk} + \alpha_k}{N_d + \sum_{k'=1}^{K} \alpha_{k'}}\ , && (13)
\end{aligned}$$

where the last two lines follow from Dirichlet/multinomial conjugacy, and from the mean of a Dirichlet distribution, respectively. Here, $z^{(new)}$ corresponds to a hypothetical "new" word in the document. The posterior predictive distribution can be understood via the urn process interpretation of collapsed Dirichlet-multinomial models. After marginalizing out the parameters θ of a multinomial distribution $\mathbf{x} \sim \text{Multinomial}(\theta, N)$ with a Dirichlet prior $\theta \sim \text{Dirichlet}(\boldsymbol{\alpha})$, we arrive at an urn process called the Dirichlet-multinomial distribution, a.k.a. the multivariate Polya urn, cf. (Minka, 2000). The urn process is as follows:

- Begin with an empty urn
- For each k, $1 \leq k \leq K$
  - add α_k balls of color k to the urn
- For each i, $1 \leq i \leq N$
  - Reach into the urn and draw a ball uniformly at random
  - Observe its color, k. Count it, i.e., add one to x_k
  - Place the ball back in the urn, along with a *new ball of the same color.*

We can interpret the posterior predictive distribution above as the probability of the next ball in the urn, i.e., the next word in the document, if we were to add one more word, by reaching into the urn once more.

3.2 A Marginalized Estimation Principle

The topic assignment vector $\mathbf{z}_d$ for test document d is a latent variable which typically has quite substantial posterior uncertainty associated with it, and yet the most common practice is to ignore this uncertainty and construct a point estimate of θ_d from a single $\mathbf{z}_d$ sample, which can be detrimental to performance (Nguyen et al., 2014). Following Griffiths and Steyvers (2004), we assume that the predictive probability of a new word's topic assignment $z^{(new)}$ is a principled estimate of θ_d. Differently to previous work, though, we take the perspective that the latent $\mathbf{z}_d$ is a nuisance variable which should be marginalized out. This leads to a *marginalized* version of Griffiths and Steyvers' estimator,

$$\bar{\theta}_{dk} \triangleq p(z^{(new)} = k|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) = \sum_{\mathbf{z}_d} p(z^{(new)} = k, \mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) \ . \tag{14}$$

Due to its treatment of uncertainty in $\mathbf{z}_d$, we advocate for $\bar{\theta}_d$ as a gold standard principle for constructing a point estimate of θ_d. We will introduce computationally efficient Monte Carlo algorithms for approximating it below. Note that while previous works have considered averaging strategies which correspond to Monte Carlo approximations of this principle, including Griffiths and Steyvers (2004), Asuncion et al. (2009), Nguyen et al. (2014), and references therein, discussion on the estimation principle of Equation 14 that these methods approximate appears to be missing in the literature. Griffiths and Steyvers (2004) mention that such averaging strategies are possible but caution against them due to concerns regarding identifiability, however this identifiability issue does not arise at test time with topics ϕ held fixed, as in the scenario we consider here. Nguyen et al. (2014) empirically explore averaging

procedures for making predictions by approximating the posterior predictive distribution. They do not aim to compute a point estimate from multiple samples, however their *Single Average* prediction strategy can be seen to be equivalent to using a naive Monte Carlo estimate (given in Equation 26) of Equation 14 in the context of predicting held-out words.

3.2.1 Interpretations

We can interpret the estimator from Equation 14 in several other ways.

Expected Griffiths and Steyvers estimator: The estimator $\bar{\theta}_d$ can readily be seen to be the posterior mean of the Griffiths and Steyvers estimator $\hat{\theta}_d$:

$$\begin{aligned}
\bar{\theta}_{dk} = p(z^{(new)} = k|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) &= \sum_{\mathbf{z}_d} p(z^{(new)} = k, \mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) && (15)\\
&= \sum_{\mathbf{z}_d} p(z^{(new)} = k|\mathbf{z}_d, \boldsymbol{\alpha}) p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) && (16)\\
&= E_{p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha})}[p(z^{(new)} = k|\mathbf{z}_d, \boldsymbol{\alpha})] && (17)\\
&= E_{p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha})}[\hat{\theta}_{dk}] \ . && (18)
\end{aligned}$$

Posterior mean of θ_d: Griffiths and Steyvers' estimator can be viewed as the posterior mean of θ_d, given $\mathbf{z}_d$ (Equation 11). Along these lines, we can interpret $\bar{\theta}_d$ as the marginal posterior mean:

$$\begin{aligned}
\bar{\theta}_{dk} = p(z^{(new)} = k|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) &= E_{p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha})}[\hat{\theta}_{dk}] && (19)\\
&= E_{p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha})}\Big[\mathbb{E}_{p(\theta_d|\mathbf{z}_d, \boldsymbol{\alpha})}[\theta_{dk}]\Big] && (20)\\
&= E_{p(\theta_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha})}[\theta_{dk}] \ , && (21)
\end{aligned}$$

where the last line follows by the law of total expectation.

Urn model interpretation: To interpret the estimator from an urn model perspective, we can plug in Equation 13:

$$\begin{aligned}
\bar{\theta}_{dk} = p(z^{(new)} = k|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) &= \sum_{\mathbf{z}_d} p(z^{(new)} = k|\mathbf{z}_d, \boldsymbol{\alpha}) p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) && (22)\\
&= \sum_{\mathbf{z}_d} \frac{n_{dk} + \alpha_k}{N_d + \sum_{k'=1}^{K} \alpha_{k'}} p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) \ . && (23)
\end{aligned}$$

We can view this as a mixture of urn models. It can be simulated by picking an urn with colored balls in it corresponding to the count vector $\mathbf{n}_d + \boldsymbol{\alpha}$, with probability according to the posterior $p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha})$, then selecting a colored ball, corresponding to a topic assignment, from this urn uniformly at random. Alternatively, we can model this equation with a single urn, with (fractional) colored balls placed in it according to the count vector,

$$\bar{\theta}_{dk} \propto \sum_{\mathbf{z}_d} (n_{dk} + \alpha_k) p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) = \sum_{\mathbf{z}_d} p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) n_{dk} + \alpha_k \ , \tag{24}$$

and with the next ball (i.e., topic) $z^{(new)}$ selected according to the total number of balls of each color (i.e., topic) in the urn.

3.2.2 Monte Carlo algorithms for computing the estimator

Due to the intractable sum over $\mathbf{z}_d$, we cannot compute $\bar{\theta}_d$ exactly. A straightforward Monte Carlo estimate of $\bar{\theta}_d$ is given by

$$\bar{\theta}_{dk} = E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}[p(z^{(new)} = k|\mathbf{z}_d, \boldsymbol{\alpha})] = E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}\Big[\frac{n_{dk} + \alpha_k}{N_d + \sum_{k'=1}^K \alpha_{k'}}\Big] \tag{25}$$

$$\approx \frac{1}{S}\sum_{i=1}^{S} \frac{n_{dk}^{(i)} + \alpha_k}{N_d + \sum_{k'=1}^K \alpha_{k'}} \ , \tag{26}$$

where each of the count variables $\mathbf{n}_d^{(i)}$ corresponds to a sample $\mathbf{z}_d^{(i)} \sim p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha})$. Algorithmically, this corresponds to drawing multiple MCMC samples of $\mathbf{z}_d$ and averaging the Griffiths and Steyvers estimator $\hat{\theta}_d$ over them. Alternatively, by linearity of expectation we can shift the expectation inwards to rewrite the above as

$$E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}\Big[\frac{n_{dk} + \alpha_k}{N_d + \sum_{k'=1}^K \alpha_{k'}}\Big] = E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}\Big[\frac{\sum_{j=1}^{N_d} z_{djk} + \alpha_k}{N_d + \sum_{k'=1}^K \alpha_{k'}}\Big] \tag{27}$$

$$= \frac{\sum_{j=1}^{N_d} E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}[z_{djk}] + \alpha_k}{N_d + \sum_{k'=1}^K \alpha_{k'}} \ . \tag{28}$$

This leads to another Monte Carlo estimate,

$$\bar{\theta}_{dk} \approx \frac{\sum_{j=1}^{N_d} \frac{1}{S}\sum_{i=1}^{S} z_{djk}^{(i)} + \alpha_k}{N_d + \sum_{k'=1}^K \alpha_{k'}} \ . \tag{29}$$

In this case we would once again draw multiple MCMC samples of $\mathbf{z}_d$, but average over the samples to compute the proportion of times that each word is assigned to each topic, applying Griffiths and Steyvers' estimator to the resulting expected total counts. It can straightforwardly be shown that this estimator is mathematically equivalent to the estimator of Equation 26, so this algorithm will perform identically to that method. We mention this formulation for expository purposes, as it will be used as a launching point for deriving our proposed CGS$_p$ algorithm.

3.3 CGS$_p$ for θ_d

We aim to design a statistically and computationally efficient Monte Carlo algorithm which improves on the above algorithms. To do this, we first consider a hypothetical Monte Carlo algorithm which is more expensive but easily understood, using Equation 29 as a starting point. We then propose our algorithm, CGS$_p$, as a more computationally efficient version of this hypothetical method. Let $\mathbf{z}_d^{(i)} \sim p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha})$, $i \in \{1, \ldots, S\}$ be S samples from the posterior for the topic assignments, and let $\mathbf{z}_d^{(i,l)\to j}$, $l \in \{1, \ldots, L\}$ be L Gibbs samples starting from $\mathbf{z}_d^{(i)}$, where a Gibbs update to the jth topic assignment is performed. The Bayesian posterior distribution is a stationary distribution of the Gibbs sampler, so these Gibbs updated samples are still samples from the posterior, i.e., $\mathbf{z}_d^{(i,l)\to j} \sim p(\mathbf{z}_d|\mathbf{w}_d, \phi, \boldsymbol{\alpha})$.

We can therefore estimate posterior expectations of any quantity of interest using these samples instead of the $\mathbf{z}_d^{(i)}$'s, and in particular:

$$\bar{\theta}_{dk} \approx \frac{\sum_{j=1}^{N_d} \frac{1}{SL}\sum_{i=1}^{S}\sum_{l=1}^{L} z_{djk}^{(i,l)\to j} + \alpha_k}{N_d + \sum_{k'=1}^K \alpha_{k'}} \ . \tag{30}$$

This method increases the sample size, and also the effective sample size, relative to the method of Equation 29, but also greatly increases the computational effort to obtain the Monte Carlo estimator, making it relatively impractical. Fortunately, a variant of this procedure exists which is both computationally faster and has lower variance. Let $T(\mathbf{z}_d^{(i)\to j}|\mathbf{z}_d^{(i)}, \phi)$ be the transition operator for a Gibbs update on the jth topic assignment. Holding the other assignments $\mathbf{z}_{d\neg j}^{(i)}$ fixed, we have the partially collapsed Gibbs update

$$T(\mathbf{z}_{djk}^{(i)\to j} = 1|\mathbf{z}_d^{(i)}, \phi) \propto \phi_{kw_{d,j}} \frac{n_{dk\neg j}^{(i)} + \alpha_k}{N_{d\neg j} + \sum_{k'=1}^K \alpha_{k'}} \ . \tag{31}$$

By the strong law of large numbers,

$$\frac{1}{L}\sum_{l=1}^{L} z_{djk}^{(i,l)\to j} \to E_{T(\mathbf{z}_d^{(i)\to j}|\mathbf{z}_d^{(i)},\phi)}[z_{djk}^{(i)\to j}] = T(z_{djk}^{(i)\to j}|\mathbf{z}_d^{(i)}, \phi) \text{ as } L \to \infty \text{ with probability 1.} \tag{32}$$

So for the cost of a single Gibbs sweep through the document, we can obtain an effective number of inner loop samples $L = \infty$ by plugging in the Gibbs sampling probabilities as the expected topic counts in Equation 30. This leads to our proposed estimator for θ_{dk}:

$$\hat{\theta}_{dk}^p \triangleq \frac{\sum_{j=1}^{N_d} \frac{1}{S}\sum_{i=1}^{S} T(\mathbf{z}_{djk}^{(i)\to j}|\mathbf{z}_d^{(i)}, \phi) + \alpha_k}{N_d + \sum_{k'=1}^K \alpha_{k'}} \ . \tag{33}$$

This method is equivalent to averaging over an infinite number of Gibbs samples adjacent to our initial samples $i = 1, \ldots, S$ in order to estimate each word's topic assignment probabilities, and thereby estimate the expected topic counts for the document, which correspond to our estimate of θ_d. By the Rao-Blackwell theorem, it has lower variance than the method in Equation 30 with finite L. We refer to this Monte Carlo procedure as CGS$_p$, for collapsed Gibbs sampling with probabilities maintained.

The above argument makes it clear that $\hat{\theta}_d^p$ is an unbiased Monte Carlo estimator of the posterior predictive distribution, but for added rigor we can also show this directly:

$$\hat{\theta}^p_{dk} \propto \sum_{j=1}^{N_d} \frac{1}{S} \sum_{i=1}^{S} T(z^{(i)\to j}_{djk} | \mathbf{z}^{(i)}_d, \phi) + \alpha_k \tag{34}$$

$$= \sum_{j=1}^{N_d} \frac{1}{S} \sum_{i=1}^{S} E_{T(\mathbf{z}^{(i)\to j}_d | \mathbf{z}^{(i)}_d, \phi)}[z^{(i)\to j}_{djk}] + \alpha_k \tag{35}$$

$$\approx \sum_{j=1}^{N_d} E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}\left[E_{T(\mathbf{z}^{\to j}_d|\mathbf{z}_d,\phi)}[z^{\to j}_{djk}]\right] + \alpha_k \tag{36}$$

$$= \sum_{j=1}^{N_d} E_{p(\mathbf{z}^{\to j}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}[z^{\to j}_{djk}] + \alpha_k \qquad \text{(by the law of total expectation)}$$

$$= \sum_{j=1}^{N_d} E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}[z_{djk}] + \alpha_k \qquad \text{(by stationarity of the Gibbs sampler)}$$

$$= E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}[\sum_{j=1}^{N_d} z_{djk}] + \alpha_k \tag{37}$$

$$= E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}[n_{dk} + \alpha_k] \tag{38}$$

$$\propto E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}[\frac{n_{dk} + \alpha_k}{N_d + \sum_{k'=1}^{K} \alpha_{k'}}] \tag{39}$$

$$= p(z^{(new)} = k|\mathbf{w}_d, \phi, \boldsymbol{\alpha}) \ . \qquad \text{(by Equation 25)}$$

We advocate the use of this procedure even when we can only afford a single burned-in sample, i.e., $S = 1$, as a cheap approximation to $p(z^{(new)} = k|\mathbf{w}_d, \phi, \boldsymbol{\alpha})$ which can be used as a plug-in replacement to Griffiths and Steyvers (2004)'s single sample estimator $p(z^{(new)} = k|\mathbf{z}_d, \boldsymbol{\alpha})$. Indeed, the increased stability of our estimator, due to implicit averaging, is expected to be especially valuable when $S = 1$. We emphasize three primary observations arising from the above analysis.

(1) We have shown that it is valid and potentially beneficial to use "soft" probabilistic counts instead of hard assignment counts when computing parameter estimates from collapsed Gibbs samples. This ports the soft clustering methodology of CVB0 to the CGS setting at estimation time. While our approach computes soft counts via Gibbs sampling transition probabilities and CVB0 uses variational distributions, the corresponding equations to compute them are very similar, cf. Equations 1, 4, and 6.

(2) Averaging these (hard or soft) count matrices over multiple samples can be performed to improve the stability of the estimators, as long as identifiability is resolved, e.g., at test time with the topics held fixed.

(3) Averaging over either hard or soft count matrices will converge to the same solution in the long run: the marginalized version of the Griffiths and Steyvers estimator, given by Equation 14. However, we expect the use of soft counts to be much more efficient in the number of samples S, as it implicitly averages over larger numbers of samples.

This last point is crucial in many common practical applications of topic modeling where we have a limited computational budget per document at prediction time, and can afford only a modest burn-in period, and few samples after burn-in. This situation frequently arises when evaluating topic models, for which a novel approach is typically compared to multiple baseline methods on a test set containing up to tens of thousands of documents, with the entire experiment being repeated across multiple corpora and/or training folds. The computational challenge is greatly exacerbated when evaluating topic model training algorithms, for which the above must be repeated at many stages of the training process (Foulds and Smyth, 2014). For instance, Asuncion et al. (2009) state that "*in our experiments we don't perform averaging over samples for CGS . . . [in part] for computational reasons.*" Computational efficiency at test time is also essential when using topic modeling for multi-label document classification (Ramage et al., 2009, 2011; Rubin et al., 2012), for which a deployed system may need to perform inference on a never-ending stream of incoming test documents.

Pseudocode illustrating the proposed method is provided in Algorithm 1. It should be noted than any existing CGS implementation can very straightforwardly be modified to apply our proposed technique, including MALLET (McCallum, 2002a), Yahoo_LDA (Smola and Narayanamurthy, 2010), and WarpLDA (Chen et al., 2016). Our method is compatible with modern sparse and parallel implementations of CGS, as the final estimation procedure does not need to alter the training process. This can lead to a "best of both worlds" situation, where sparsity is leveraged during the expensive CGS inference process, but the full dense distributions are used during the relatively inexpensive parameter recovery step, which in many topic modeling applications is performed only once. In this setting, the improved performance is typically worth the computational overhead (which is similar to that of a single dense CGS update), as we demonstrate in Section 5.8. A practical implementation of our estimators can further leverage the fact that the count matrices are not modified during the procedure, by computing the sampling distribution only once for each distinct word type in each document, and by performing the computations in parallel. In the following we will sometimes abuse notation and denote the procedure of using $\hat{\theta}^p_d$ to estimate the θ_d's as θ^p (and similarly for the procedure to estimate ϕ, denoted ϕ^p, introduced below).

3.3.1 CGS$_p$ for estimating θ_d for training documents

During the training phase, we can apply the same procedure to estimate θ_d for training documents based on a given collapsed Gibbs sample $\mathbf{z}^{(i)}$:

$$\hat{\theta}^p_{dk} = \frac{\sum_{j=1}^{N_d} T(z^{(i)\to j}_{djk} | \mathbf{z}^{(i)}, \phi) + \alpha_k}{N_d + \sum_{k'=1}^{K} \alpha_{k'}} \ . \tag{40}$$

The collapsed Gibbs sampler does not explicitly maintain an estimate of ϕ to plug into the above. However, the Griffiths and Steyvers estimator for ϕ (Equation 2) can be obtained from the same count matrices that CGS computes, as it corresponds almost exactly to the first term on the right hand side of Equation 1. Alternatively, below we propose a novel estimator for ϕ which can also be used here with a little bit more computational effort.

Similarly to the Griffiths and Steyvers estimator, it should be noted that averaging this estimator over multiple samples from different training iterations raises issues with the

identifiability of the topics, which can potentially be problematic (Griffiths and Steyvers, 2004). We do not consider this procedure further here. It is however safe to apply Equation 33 to multiple MCMC samples for $\mathbf{z}_d$ that are generated while holding ϕ fixed, i.e., treating document d as a test document for the purposes of estimating θ_d given the $\hat{\phi}$ from $\mathbf{z}^{(i)}$.

4. CGS$_p$ for Topic - Word Types Parameter Estimation

By applying the same approach as for θ^p, plugging in Gibbs transition probabilities instead of indicator variables into Equation 2, we arrive at an analogous technique for estimating ϕ:

$$\hat{\phi}^p_{kv} = \frac{\sum_{d=1}^{D_{Train}} \sum_{j:w_{d,j}=v} T(z^{(i)\rightarrow j}_{djk}|\mathbf{z}^{(i)}) + \beta_v}{\sum_{d=1}^{D_{Train}} \sum_{j=1}^{N_d} T(z^{(i)\rightarrow j}_{djk}|\mathbf{z}^{(i)}) + \sum_{v'=1}^{V} \beta_{v'}} \ , \tag{41}$$

where $T(z^{(i)\rightarrow j}_{d,j}|\mathbf{z}^{(i)})$ corresponds to the collapsed Gibbs update in Equation 1. While the intuition of the above method is clear, there are several technical complications to an analogous derivation for it. Nevertheless, we show that the same reasoning essentially holds for ϕ, with minor approximations. In the following, we outline a justification for the ϕ^p estimator, with emphasis on the places where the argument from θ^p does not apply, along with well-principled approximation steps to resolve these differences.

First, consider the estimation principle underlying the technique. For topics ϕ, the standard Rao-Blackwellized estimator of Griffiths and Steyvers (2004), given by Equation 2, once again corresponds to the posterior predictive distribution,

$$\begin{aligned}
\hat{\phi}_{kv} &\triangleq p(w^{(new)} = v|\mathbf{z}, \mathbf{w}, \boldsymbol{\beta}, z^{(new)} = k) \\
&= \int p(w^{(new)} = v, \phi_k|\mathbf{z}, \mathbf{w}, \boldsymbol{\beta}, z^{(new)} = k) d\phi_k \\
&= \int p(w^{(new)} = v|\phi_k, z^{(new)} = k) p(\phi_k|\mathbf{w}_{z=k}, \boldsymbol{\beta}) d\phi_k \\
&= E_{Dirichlet(\mathbf{n}_k+\boldsymbol{\beta})}[\phi_{kv}] \\
&= \frac{n_{kv} + \beta_v}{\sum_{v'=1}^{V} (n_{v'k} + \beta_{v'})} \ ,
\end{aligned} \tag{42}$$

where $\mathbf{w}_{z=k}$ is the collection of word tokens assigned to topic k. This estimator again conditions on the latent variable assignments $\mathbf{z}$, which are uncertain quantities. We desire a marginalized estimator, summing out the latent variables $\mathbf{z}$. However, without conditioning on $\mathbf{z}$, the topic index k loses its meaning because the topics are not identifiable. The naive marginalized estimator

$$\bar{\phi}_{kv} \triangleq p(w^{(new)} = v|\mathbf{w}, \boldsymbol{\beta}, z^{(new)} = k) = \sum_{\mathbf{z}} p(w^{(new)} = v, \mathbf{z}|\mathbf{w}, \boldsymbol{\beta}, z^{(new)} = k) \tag{43}$$

sums over all permutations of the (collapsed representation of the) topics, and so conditioning on $z^{(new)} = k$ has no effect. The estimator in Equation 43 is therefore unfortunately not well defined. On the other hand, a sum over $\mathbf{z}$'s *is* meaningful as long as *the topics are aligned*

for all values of $\mathbf{z}$ *considered.* Suppose, by some oracle, we know that $\mathbf{z}$ belongs to some *identified subset* of assignments $\dot{Z}$, for which topic indices are aligned between all elements $\mathbf{z}' \in \dot{Z}$. We are then able to properly define our idealized estimator with respect to $\dot{Z}$,

$$\bar{\phi}^{(\dot{Z})}_{kv} \triangleq p(w^{(new)} = v|\mathbf{w}, \boldsymbol{\beta}, z^{(new)} = k, \mathbf{z} \in \dot{Z}) = \sum_{\mathbf{z} \in \dot{Z}} p(w^{(new)} = v, \mathbf{z}|\mathbf{w}, \boldsymbol{\beta}, z^{(new)} = k, \mathbf{z} \in \dot{Z}) \ . \tag{44}$$

Regarding our proposed technique, the Monte Carlo estimator in Equation 41 implicitly averages over the set of possible Gibbs samples $Z_{Gibbs}(\mathbf{z}^{(i)})$ that are adjacent to sample $\mathbf{z}^{(i)}$, i.e., the candidate $\mathbf{z}$ assignments that differ from $\mathbf{z}^{(i)}$ in a single entry. A difference of a single word is extremely unlikely to cause label switching for a realistic corpus, so the topics will almost certainly be aligned for each of the adjacent Gibbs samples that the estimator implicitly averages over. The empirical results of Nguyen et al. (2014) also support this, as they found that averaging over multiple samples from the same MCMC chain was beneficial to performance, which suggests that identifiability was not a problem even for MCMC samples that are many iterations apart. Hence, we take $\bar{\phi}^{(Z_{Gibbs}(\mathbf{z}^{(i)}))}_{kv}$ as our idealized estimator, under the supposition that $Z_{Gibbs}(\mathbf{z}^{(i)})$ is identified, and we aim to derive Equation 41 as a Monte Carlo algorithm that efficiently approximates it.

To accomplish this, following the derivation for $\hat{\theta}^p$, we would like to show that

(1) we can use a Monte Carlo estimator in which the $z_{d,j}$'s can be sampled individually, as in Equation 29, and

(2) the transition probabilities in Equation 41 can be used to implement this estimator with an effectively infinite number of adjacent Gibbs samples, as in Section 3.3.

While the previous stationarity argument for (2) still applies, there is a further complication for (1). For $\hat{\theta}^p$, linearity of expectation was sufficient to show in Equation 28 that

$$E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}\Big[\frac{\sum_{j=1}^{N_d} z_{djk} + \alpha_k}{N_d + \sum_{k'=1}^{K} \alpha_{k'}}\Big] = \frac{\sum_{j=1}^{N_d} E_{p(\mathbf{z}_d|\mathbf{w}_d,\phi,\boldsymbol{\alpha})}[z_{djk}] + \alpha_k}{N_d + \sum_{k'=1}^{K} \alpha_{k'}} \ .$$

We cannot use this argument directly for $\hat{\phi}^p$ because in this case the denominator of the Griffiths and Stevyers estimator, which is averaged over, also includes $\mathbf{z}$ terms which are involved in the expectation:

$$\bar{\phi}^{(Z_{Gibbs}(\mathbf{z}^{(i)}))}_{kv} = E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}} \sum_{j:w_{d,j}=v} z_{djk} + \beta_v}{\sum_{d=1}^{D_{Train}} \sum_{j=1}^{N_d} z_{djk} + \sum_{v'=1}^{V} \beta_{v'}}\Big] \ . \tag{45}$$

The issue here is that the adjacent Gibbs samples modify the overall counts per topic $n_k \triangleq \sum_{d=1}^{D_{Train}} \sum_{j=1}^{N_d} z_{djk}$. However, between adjacent Gibbs samples $\mathbf{z}$ and $\mathbf{z}^{(i)}$, the corresponding topic counts n_k are within ± 1 of $n^{(i)}_k$ and for a typical corpus we have that $n_k >> 1$, so the

impact of a single word is negigible, i.e.,

$$\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk} \in \Big\{ \sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}, \sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk} \pm 1 \Big\} \approx \sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk} \ ,$$
$$\text{for } \mathbf{z} \in Z_{Gibbs}(\mathbf{z}^{(i)}) \ , \ \sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk} >> 1 \ . \tag{46}$$

In this case, we see from Equation 46 that for a corpus in which $n_k >> 1$, for all practical purposes the denominator in Equation 45 is essentially constant over all adjacent Gibbs samples, and so (1) still holds approximately, and we have

$$\begin{aligned}
\bar{\phi}^{(Z_{Gibbs}(\mathbf{z}^{(i)}))}_{kv} &= E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big] \\
&\approx E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big] \\
&= \frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}[z_{djk}]+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}} \ .
\end{aligned} \tag{47}$$

A related argument was also made by Asuncion et al. (2009), who note that several topic modeling inference algorithms differ only by offsets of 1 or 0.5 to the counts, and state that "*since n_k is usually large, we do not expect [a small offset] to play a large role in learning.*" To formalize this intuition, as we show in Appendix B, the approximation can be bounded from below and from above as a function of $n^{(i)}_k$ as:

$$\begin{aligned}
&\frac{n^{(i)}_k+\sum_{v'=1}^{V}\beta_{v'}}{n^{(i)}_k+\sum_{v'=1}^{V}\beta_{v'}+1} E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big] \\
&\qquad\qquad \le E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big] \\
\le &\frac{n^{(i)}_k+\sum_{v'=1}^{V}\beta_{v'}}{n^{(i)}_k+\sum_{v'=1}^{V}\beta_{v'}-1} E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big] \ ,
\end{aligned} \tag{48}$$

for $n^{(i)}_k > 0$. Since $\lim_{n^{(i)}_k\to\infty}\frac{n^{(i)}_k+\sum_{v'=1}^{V}\beta_{v'}}{n^{(i)}_k+\sum_{v'=1}^{V}\beta_{v'}+1}=1$ and $\lim_{n^{(i)}_k\to\infty}\frac{n^{(i)}_k+\sum_{v'=1}^{V}\beta_{v'}}{n^{(i)}_k+\sum_{v'=1}^{V}\beta_{v'}-1}=1$, the left and right hand sides of Equation 47 will converge in the limit as the $n^{(i)}_k$'s go to infinity, e.g., when increasing the counts via a stream of incoming documents. In Appendix C, we provide a further empirical illustration of this approximation, by taking subsets of the documents in the corpus to vary n_k, and observing that as n_k increases the difference between the left and right sides of Equation 47 approaches zero.

Finally, using the same stationarity and law of large numbers arguments as before, we obtain the $\hat{\phi}^p$ estimator by plugging in the transition operator, and then renormalizing,

$$\hat{\phi}^p_{kv} \propto \sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}[z_{djk}]+\beta_v = \sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} T(z^{(i)\to j}_{djk}|\mathbf{z}^{(i)})+\beta_v \ . \tag{49}$$

More formally, the full argument is given mathematically as

$$\begin{aligned}
\hat{\phi}^p_{kv} &\propto \sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} T(z^{(i)\to j}_{djk}|\mathbf{z}^{(i)})+\beta_v && (50)\\
&= \sum_{d=1}^{D_{Train}}\sum_{j:w_{dj}=v} E_{T(\mathbf{z}^{(i)\to j}_d|\mathbf{z}^{(i)}_d)}[z^{(i)\to j}_{djk}]+\beta_v && (51)\\
&= \sum_{d=1}^{D_{Train}}\sum_{j:w_{dj}=v} E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}[z_{djk}]+\beta_v \\
&\qquad\qquad \text{(by stationarity of the Gibbs sampler)} \\
&\propto \frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{dj}=v} E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}[z_{djk}]+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}} && (52)\\
&= E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{dj}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big] && (53)\\
&\approx E_{p(\mathbf{z}|\mathbf{w},\beta,\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{dj}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big] && \text{(if } n_k >> 1) \\
&= p(w^{(new)}=v|\mathbf{w},\boldsymbol{\beta},z^{(new)}=k,\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)})) \ . && (54)
\end{aligned}$$

In Algorithm 1, along with θ^p's procedure, we also present the pseudocode for the ϕ^p estimator.

5. Unsupervised Learning Experiments—LDA

This section describes the experiments that we performed to study the behavior of our proposed estimators in the context of unsupervised LDA models. We describe the data sets, the evaluation procedure and the experiments performed in the unsupervised LDA setting. Apart from the experiments reported here, we have performed two additional ones reported in the Appendix, (a) comparing the document-topic counts of the Collapsed Gibbs Sampling (CGS) algorithm against the *soft* document-topic counts, in order to study their differences as the algorithm converges and (b) providing an empirical validation of the approximation that we use in Equation 47 regarding our ϕ^p estimator's derivation. The relevant code of the following experiments (along with the code of the multi-label experiments of Section 6) can be found in `https://github.com/ypapanik/cgs_p`.

Data Set	D_{Train}	D_{Test}	$\overline{N_d}$	V
BioASQ	20,000	10,000	113.55	135,186
New York Times	14,668	7,000	581.13	19,259
Reuters-21578	11,475	4,021	49.68	41,014
TASA	30,121	7,529	107.62	21,970

Table 3: Statistics for the data sets used in the unsupervised learning experiments. Average document length and number of word types refer to the training sets.

5.1 Data Sets

Four data sets were used in the unsupervised setting: a) BioASQ, b) New York Times, c) Reuters-21578 and d) TASA. The number of documents in the training and test sets, the average length of the training documents and the size of the vocabulary of word types found in the training set are presented in Table 3.

The first data set originates from the *BioASQ* challenge (Balikas et al., 2014) that deals with large-scale online multi-label classification of biomedical journal articles. This learning task is particularly challenging as the taxonomy of labels includes around $27,000$ terms, with highly imbalanced frequencies. For the unsupervised experiments of this section, we used the $30,000$ last documents of the BioASQ corpus of the year 2014, using the first $20,000$ as training documents and the rest as test documents. To construct the data set we concatenated the abstract and title of each article and removed common stopwords. The remaining unigrams were used as word types.

The second data set contains articles published by the *New York Times*, manually annotated via their indexing service. We used the same data set as used by Rubin et al. (2012), with the same training set (14,668 documents) and keeping the first 7,000 documents for testing (out of the 15,989 of the original paper).

The third data set[4] contains documents from the *Reuters* news-wire and has been widely used among researchers for almost two decades. The split that we used has 11,475 documents for training and 4,021 documents for testing. We preprocessed the corpus by removing common stopwords.

The *TASA* data set contains $37,650$ documents of diverse educational materials (e.g., health, sciences, etc) collected by Touchstone Applied Science Associates (Landauer et al., 1998). We used the first 30,121 documents as a training set and the remaining as a test set. The corpus already had stopwords and infrequent words removed, so we did not perform any further preprocessing.

5.2 Evaluation

Evaluation of LDA models typically focuses on the probability of a set of held-out documents given an already trained model (Wallach et al., 2009). In this context, one must compute the model's posterior predictive likelihood of all words in the test set, given estimates of the topic parameters ϕ and the document-level mixture parameters θ.

4. `http://disi.unitn.it/moschitti/corpora/Reuters21578-Apte-115Cat.tar.gz`

The log likelihood of a set of test documents D_{Test}, given an already estimated model M, is given by Heinrich (2004) as:

$$\text{Log Likelihood} = \sum_{d=1}^{D_{Test}} \log p(\mathbf{w}_d|M) = \sum_{d=1}^{D_{Test}} \sum_{i=1}^{N_d} \log \sum_{k=1}^{K} (\phi_{kv} \cdot \theta_{dk}) \tag{55}$$

with $w_i = v$. The perplexity will be:

$$\text{Perplexity} = \exp(-\frac{\text{Log Likelihood}}{\sum_{d=1}^{D_{Test}} N_d}) \tag{56}$$

where lower values of perplexity signify a better model.

Since θ_d is unknown for test documents, and is intractable to marginalize over, a common practice in the literature in order to compute the above likelihood is to run the CGS algorithm for a few iterations on the first half of each document, and then to compute the perplexity of the held-out data (the second half of each document), based on the trained model's posterior predictive distribution over words (Asuncion et al., 2009). This is the approach we follow.

5.3 Comparison between CGS and CGS$_p$

In this first experiment, the motivation is to validate the previously presented theory behind the ϕ^p and θ^p methods. Specifically, we are interested in verifying the following hypothesis: given a single burned-in sample, $\hat{\phi^p}$ or $\hat{\theta^p}$ will more effectively estimate the respective LDA parameters compared to the standard estimators, since the CGS$_p$ estimators make use of the full dense probabilities, through the infinite l-steps of Eq. 33. This advantage would correspond to lower perplexity values for a single burned-in sample for the CGS$_p$ estimators compared to the standard CGS estimator. Furthermore, when averaging over multiple samples to estimate the θ parameters during prediction, we expect that the θ^p and standard θ estimators will eventually converge to the same solution given enough samples or Markov chains, since θ^p aims to compute the posterior mean. However, we expect that θ^p will converge to the posterior mean more rapidly in the number of samples that are averaged over than the standard estimator θ, with correspondingly faster improvement in perplexity.

For this experiment we used all four data sets and considered four different topic number configurations (20, 50, 100, 500). For brevity, we report only the results for $K = 100$ and include the rest of the plots in Appendix D. During training, we ran one Markov chain for 200 iterations and took a single sample to calculate ϕ and ϕ^p from the same chain. During prediction, since ϕ is fixed and topics are not exchanged through the Gibbs sampler's iterations, we took multiple samples from multiple Markov chains, and averaged over these samples using the θ (standard CGS) and θ^p estimators. The burn-in period for the Gibbs sampler was set to 50 iterations and a sampling lag of 5 iterations was used. The $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ hyperparameters were symmetrical across all topics and words respectively and set such that $\alpha_k = 0.1$ and $\beta_v = 0.01$. Finally, to ensure fairness between the θ and θ^p estimators, we used the same samples from the same chains to compute the respective estimates.

In Figures 1 and 2 we show the perplexity scores against the number of samples that are averaged over, after burn-in, for one and five Markov chains respectively. All four

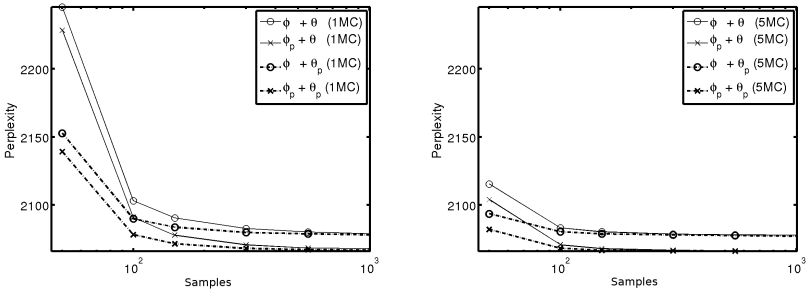
combinations of the estimators are depicted. By observing the results we notice some key points across all plots. First, if we consider the case where only one burned-in sample is used, corresponding to the left-most points in the plots, we can notice that both θ^p methods ($\phi + \theta^p$ and $\phi^p + \theta^p$) decisively outperform the other two methods. This observation is particularly important for scenarios where we can't afford to average over many samples due to computational or time limitations, as frequently occurs during the comparative evaluation of topic modeling methods, and in multi-label document classification.

We also see that the ϕ^p methods ($\phi^p + \theta$ and $\phi^p + \theta^p$) have a short but steady advantage over the respective ϕ estimators. This advantage is not diminished for more samples or Markov chains, suggesting that ϕ^p is actually a more accurate estimator than ϕ. We also remind here that in unsupervised scenarios we cannot average over samples to compute a ϕ estimate during training, since topics may be interchanged between iterations. A third remark, also aligned with our theoretical results, is that θ and θ^p converge to the same solution after a sufficient number of samples are averaged over, although θ^p converges much more rapidly. Overall, these findings confirm that indeed both θ^p and ϕ^p constitute improved estimators compared to the respective standard ones: θ^p provides more rapid convergence than using the standard estimator θ when averaging over samples for prediction on test documents, and ϕ^p improves perplexity due to implicit averaging. The rest of the plots in Appendix D are in accordance with the above observations.

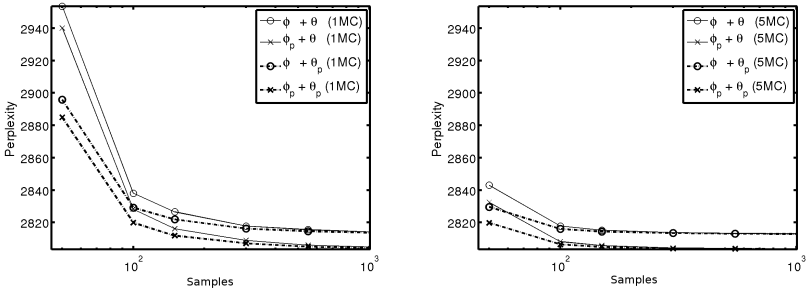
5.4 Word Association: ϕ^p vs ϕ

In this experiment, we compare ϕ^p with ϕ on a word association task. In word association, a given word, called the *norm* or *cue* word, is associated with a number of semantically related words, called *targets* or *associates*. We consider the data set provided by Nelson et al. (2004), which contains a set of 5,019 norm words and for each of them a set of associate words provided by human annotators. We aim to see how, given a specific cue word, the two LDA estimators rank the corresponding associates, in order to assess which of the two performs better at predicting the targets, in terms of the median difference in ranks. The word association task provides a useful benchmark for evaluating the extent to which the topic representations are a good model of human semantic representation (Griffiths et al., 2007).

We largely follow the same setup as that described by Griffiths et al. (2007), pages 220-224. Specifically, we use the TASA data set, removing only stopwords and word types with fewer than five and more than ten thousand occurrences in the corpus, resulting in a vocabulary of 26,186 word types. There are 4,506 norms that belong to the obtained vocabulary. Subsequently, we train an LDA model on the corpus for $K = 50, 100, 500, 1000$ and for 250 iterations, obtaining a single point estimate for ϕ^p and ϕ for each value of K at the end of the procedure. Our goal is to see how each of the two estimators rank the five most commonly associated responses to each of the norm words, as given by human subjects. Similarly to Griffiths et al. (2007) (refer to p.220 and Appendix B, p.244), in order to assess semantic association we use the conditional probability $p(w_2|w_1)$ where w_1 stands for a given



(a) BioASQ



(b) New York Times

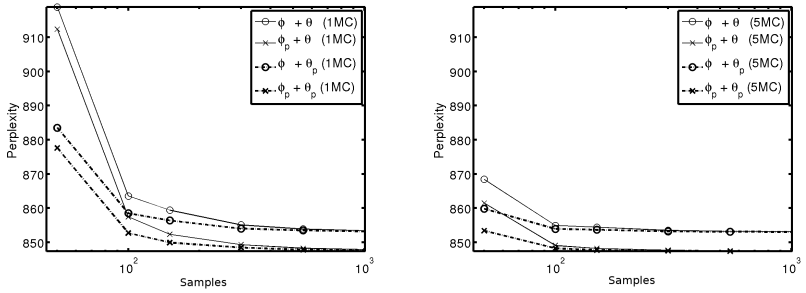
Figure 1: Perplexity against the number of samples averaged over, for the CGS_p estimators and standard CGS. Results are taken by averaging over 5 different runs. Samples are taken after a burn-in period of 50 iterations.

cue word and w_2 denotes each of the rest of the vocabulary word types. Supposing that w_1 and w_2 belong to the same topic z, and assuming a uniform prior on z, we have[5]

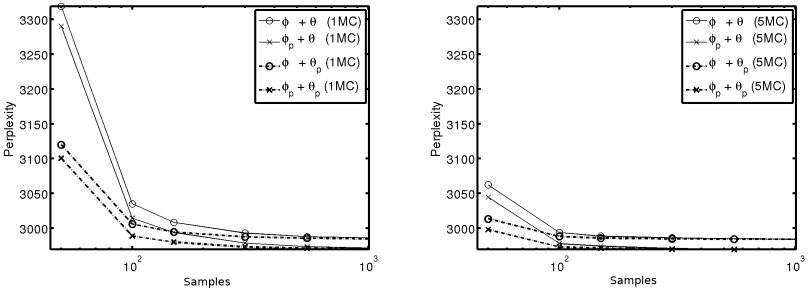
$$p(w_2|w_1) = \sum_z p(w_2|z)p(z|w_1) = \sum_z p(w_2|z)\frac{p(w_1|z)p(z)}{\sum_{z'} p(w_1|z')p(z')}$$
$$= \sum_z p(w_2|z)\frac{p(w_1|z)}{\sum_{z'} p(w_1|z')} \,. \tag{57}$$

Next, we consider the first five associate words of the norm w_1 and obtain the rank for each of them according to the probabilities $p(w_2|w_1)$, for each of ϕ^p and ϕ. In Figure 3, we

5. A uniform prior $p(z)$ follows if z is understood to be the first word in a document under the collapsed LDA urn model, with symmetric hyperparameters α. With asymmetric hyperparameters, $p(z = k) \propto \alpha_k$.



(a) Reuters-21578

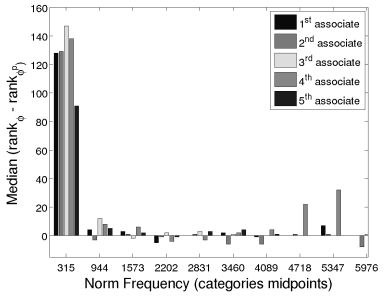


(b) TASA

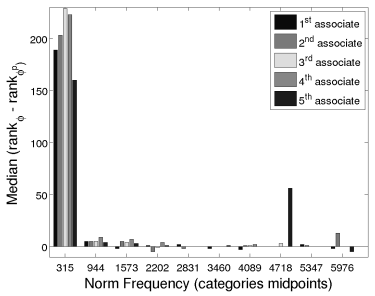
Figure 2: Perplexity against the number of samples averaged over, for the CGS_p methods and standard CGS. Results are taken by averaging over 5 different runs. Samples are taken after a burn-in period of 50 iterations.

report the median difference in ranks, $rank_\phi - rank_{\phi^p}$ against word frequency, a positive value indicating an advantage for ϕ^p. To enhance readability, we have grouped the norms according to their frequencies within the corpus, into 10 categories.

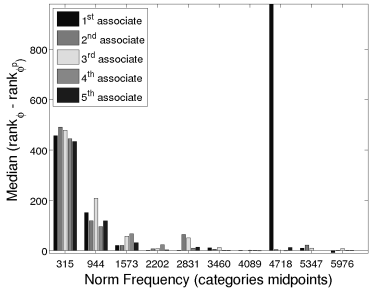
We observe two steady trends. First, ϕ^p clearly outperforms ϕ for rare norms (the first category), which is also the most populous (3,814 norms). For more frequent norms, ϕ^p and ϕ behave almost on par. This trend reveals the distinctive benefit of our proposed estimator compared to its counterpart, especially if we take into account that word type frequencies typically follow a power law like distribution in document collections. A second trend, consistent with our other experiments, relates to the number of topics K: as K increases and the complexity of the LDA model correspondingly increases, the advantage of ϕ^p becomes wider compared to ϕ. We similarly observe that ϕ^p provides a benefit for increasingly frequent words as K increases.



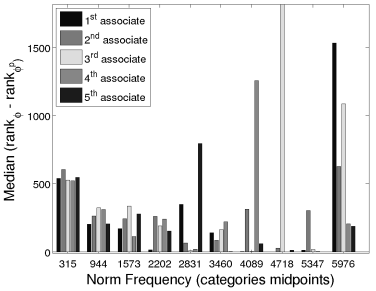
(a) K = 50



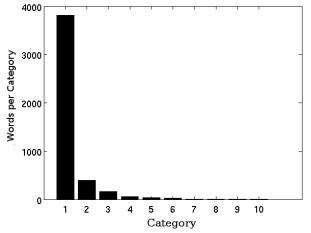
(b) K = 100



(c) K = 500



(d) K = 1000



(e) Norms per category.

Figure 3: Median difference in ranks produced by ϕ^p and ϕ estimators for the first five associate words, for a set of 4,506 norms (Nelson et al., 2004). The estimators were computed after training LDA on the TASA corpus. A positive value indicates an advantage for ϕ^p.

These results are consistent with our analysis in Section 4 which showed that ϕ^p performs implicit averaging in order to improve the stability of the topics. The nuanced uncertainty

information, and corresponding stability, provided by ϕ^p is most important for less frequent words, and for larger K, for which the posterior uncertainty in the (smaller) count values n_{kv} per (word,topic) pair is most consequential. For larger (word, topic) counts n_{kv}, the expected counts estimated by ϕ^p (Equation 47) approach the observed counts computed by ϕ due to the law of large numbers. Consequently, we typically observe little difference in the top-words lists generated by ϕ^p and ϕ, which are often used to assess the interpretability of the topics. Instead, the results of this experiment suggest that ϕ^p conveys the most benefit for tasks that depend on the word/topic probabilities for the less frequent to moderately frequent words, as in word association, and any other task which requires semantic representations of the words.

5.5 Comparison between CGS, CGS$_p$ and CVB0

In this experiment we compared the CGS, CGS$_p$ and CVB0 algorithms in terms of perplexity, across different numbers of topics. The motivation here was to examine how the CGS$_p$ method would perform compared to the rest of the algorithms, in a variety of configurations and data sets.

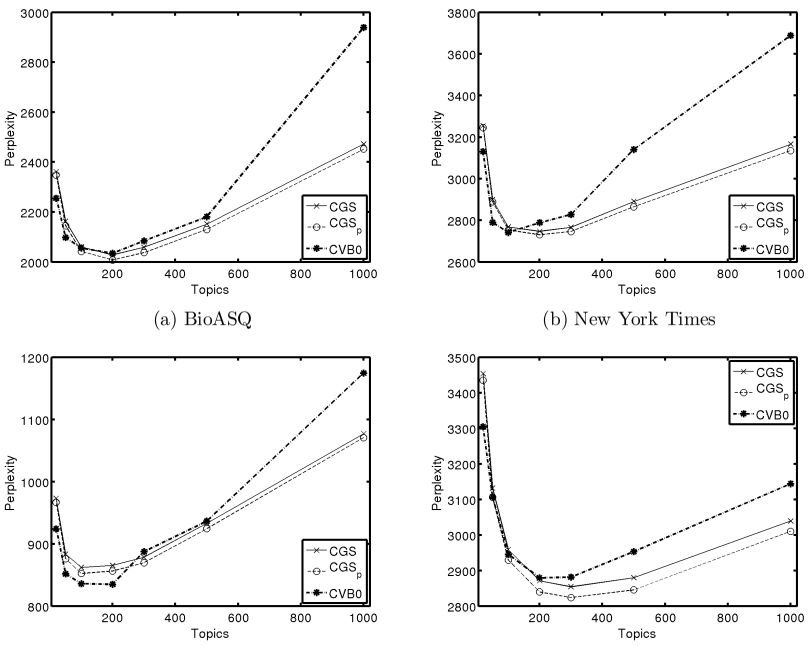
In Appendix E we report also the results of this experiment, comparing the three aforementioned algorithms with Variational Bayes (VB). Since the results for VB were steadily worse, we excluded them from Figure 4, to allow for easier comparison among CVB0, CGS and CGS$_p$. Asuncion et al. (2009) note that the disadvantage of VB versus the other algorithms can potentially be mitigated via hyperparameter learning.

For this experiment we followed a similar approach to the one described by Asuncion et al. (2009). During training we ran each chain for 200 iterations to obtain a single point estimate of the ϕ and ϕ^p distributions. During prediction, using the previously estimated ϕ, we ran 200 iterations from one chain for the first half of each document to obtain an estimate of θ. We followed the same approach to compute θ^p, using ϕ^p in that case.[6] We then computed perplexity on the second half of each document. The α_k and β_v hyperparameters were fixed across all data sets to uniform values, 0.1 and 0.01 respectively. CVB0 was run with the same parameterization.

Figure 4 shows the perplexity results for all data sets across different settings (20, 50, 100, 200, 300, 500 and 1000 topics) for all methods. Each topics configuration was run for five times, obtaining the average perplexity value. There is a strong interaction between the perplexity scores, the data sets, and the number of topics, probably due to the diverse statistics of the data sets, such as the average document length and the average number of features (i.e., word types) per document (see Table 3 in Section 5.1), that characterize the data sets. All algorithms achieve their lowest perplexity values at around 200 topics.

Despite the peculiarities of individual data sets, we can identify a broad general trend in these results. CVB0 outperforms (marginally in three out of four cases) the other methods in all data sets for lower topic number values: for the BioASQ and TASA data sets this happens up to 50 topics, for the New New York Times data set up to 100 topics, while for Reuters-21578 up to 200 topics. As the number of topics increases though, this behavior is reversed and CGS and CGS$_p$ have the upper hand after the aforementioned numbers of topics.

6. When referring to CGS$_p$, we mean that the $\phi^p + \theta^p$ estimators were used.

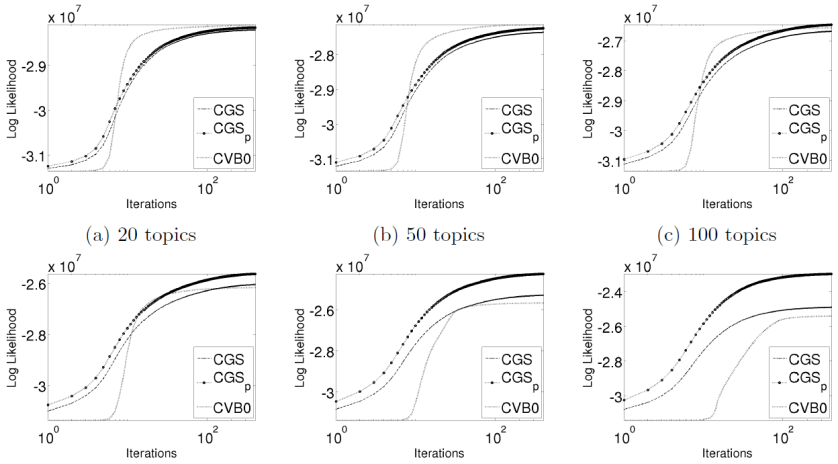


(a) BioASQ (b) New York Times

(c) Reuters-21578 (d) TASA

Figure 4: Perplexity against number of topics for the CGS$_p$ method, standard CGS and CVB0. Results are taken by averaging over 5 different runs.

A possible explanation for this observation is related to the deterministic nature of CVB0; compared to its stochastic counterpart CGS, CVB0 is more prone to getting stuck in local maxima. As the number of topics increases, we expect the hypothesis space to grow bigger, making it more difficult for CVB0 to find a global optimum. CGS on the other hand, can exploit its stochastic nature to escape local maxima and converge to a better global representation of the data. Therefore, CVB0 seems to be better suited for configurations with a small number of topics (in which case the fact that CVB0 converges a lot faster than CGS as shown by Asuncion et al. (2009) is an additional advantage), while CGS fits better in the opposite case. Similar to the previous section, when comparing CGS and CGS$_p$ we can observe a slight but steady advantage of our method over the standard estimator, across all experimental settings.



(a) 20 topics (b) 50 topics (c) 100 topics

(d) 200 topics (e) 500 topics (f) 1000 topics

Figure 5: CGS, CGS_p and CVB0 convergence in terms of log likelihood during 400 training iterations in the TASA subset for different numbers of topics. α_k and β_v were fixed to 0.1 and 0.01 respectively.

5.6 Convergence During Estimation for CGS, CGS_p and CVB0

The perplexity results of the previous section motivated us to further investigate the convergence behavior of CGS, CGS_p and CVB0 algorithms. In particular, we wanted to know the conditions under which CGS (and our modification of it) outperformed CVB0 and vice versa. Furthermore, we wanted to see how these differences evolved as the two algorithms converged upon their respective solutions.

Figure 5 presents a comparison of CVB0, CGS and CGS_p (during the estimation phase) on the TASA data set in terms of their log likelihoods over iterations, under different numbers of topics. To calculate the log likelihood for CGS_p we used the ϕ^p and θ^p estimators. For each algorithm and each topics configuration, we performed five runs and obtained the mean log-likelihood values.

The results seem to validate our previous observations with CVB0 converging faster and to a higher log likelihood than CGS and CGS_p when $|K| \leq 100$. As the number of topics grows and the number of training iterations increases, though, the performance of CGS and CGS_p catches up, and after 100 topics, they outperform CVB0. This is consistent with our hypothesis that CVB0 performance is worse than that of CGS and CGS_p as the hypothesis space grows, due to it being a deterministic algorithm and getting stuck in local maxima. These results seem to be consistent with the observations made by Teh et al. (2006, Section 4) and could eventually emerge from either (or both) of the following factors:

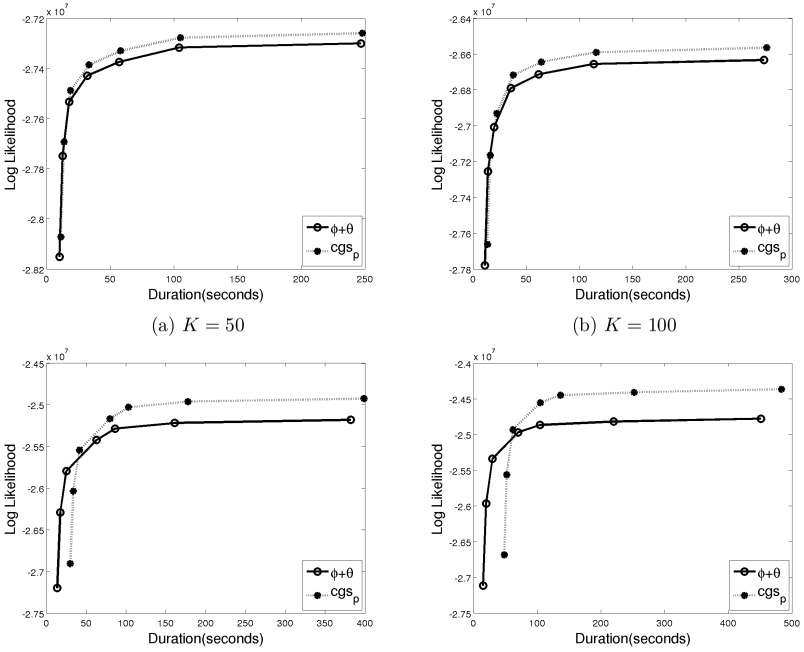
Method	Data sets			
	BioASQ	New York Times	Reuters-21578	TASA
		MALLET		
$\phi + \theta$	-1.921	-6.770	-0.590	-2.572
$\phi^p + \theta$	-1.912	-6.763	-0.585	-2.567
$\phi + \theta^p$	-1.911	-6.762	-0.584	-2.562
$\phi^p + \theta^p$	-1.903	-6.751	-0.580	-2.560
		WarpLDA		
$\phi + \theta$	-3.22	-7.092	-1.014	-2.730
$\phi^p + \theta$	-3.207	-7.081	-0.997	-2.724
$\phi + \theta^p$	-3.208	-7.079	-0.992	-2.718
$\phi^p + \theta^p$	-3.203	-7.074	-0.990	-2.716

Table 4: CGS_p vs standard CGS estimator on MALLET and WarpLDA in terms of the Log Likelihood. All values are $\times 10^7$ and represent mean values across five runs.

- Variational inference methods (VB, CVB and CVB0) approximate the true posterior with a more tractable variational distribution (Blei et al., 2003; Teh et al., 2006; Asuncion et al., 2009). Furthermore, in the case of CVB0, as illustrated by (Foulds, 2014, Section 4.4.1), the algorithm is a result of three consecutive approximations of the initial problem. On the other hand, CGS performs exact collapsed Gibbs updates and will eventually sample from the true posterior.

- CVB0 seems to be unable to avoid local maxima due to its deterministic nature, especially as the hypothesis space grows bigger, while CGS takes advantage of its stochastic nature in order to avoid them.

A secondary observation is that CGS_p increases its advantage over CGS as the number of topics grows. A final noteworthy observation from Figure 5 is that CVB0 exhibits very slow convergence behavior in the first 5 – 10 iterations, before reaching a regime of rapid convergence. While this phenomenon was not immediately visible in the results reported by Asuncion et al. (2009) (Figure 2), this apparent discrepancy is readily explained: in those results a linear scale was used on the X-axis, with data points plotted only every 20 iterations. This slow convergence behavior is in alignment with the well-known convergence speed issues for the closely related EM algorithm, which can also be understood as a variational method. For EM, Salakhutdinov et al. (2003) showed that convergence is very slow when the missing data distributions are highly uncertain. This is likely to be the case in the initial stages of CVB0 and CGS, where the count histograms are relatively flat. We hypothesize that the hard assignments of CGS allow the algorithm to escape the slow-converging high-entropy regime more rapidly than CVB0, leading to better convergence behavior in the early iterations, before the determinism of CVB0 once again gives it the upper hand (at least, with a small number of topics).



(a) $K = 50$ (b) $K = 100$

(c) $K = 500$ (d) $K = 1000$

Figure 6: Log Likelihood vs duration for CGS_p against the standard CGS estimators on TASA, with WarpLDA. The time duration for each point in the plot corresponds to the running time of WarpLDA for different numbers of iterations (50, 100, 200, 500, 1000, 2000, 5000), plus the overhead needed to compute each of the estimators at that timestep.

5.7 CGS_p with MALLET and WarpLDA

In order to exhibit how straightforward it is to plug our methods into already existing CGS LDA implementations and, most importantly, in modern CGS variants, we have employed CGS_p on top of the MALLET software package (McCallum, 2002b) and the WarpLDA (Chen et al., 2016) implementation. In all cases, we have used $K = 100$, $\alpha = 0.1$, $\beta = 0.01$. After running the respective implementations on the four datasets for 200 iterations, we have obtained a single sample for the z-assignments. Subsequently, and after calculating the n_{kw} and n_{dk} counters, we have computed the standard estimator and our proposed estimators. Table 4 depicts the results in terms of the training data log-likelihood, showing a steady advantage of our methods compared to the standard estimator.

5.8 Run-time Overhead: CGS_p vs CGS

In the previous experiments, we have observed that our proposed estimators steadily outperformed the standard CGS estimators. Nevertheless, our methods entail a computational time overhead since they require computing the full CGS probability distributions, which is roughly equivalent to a single dense standard CGS update. This overhead is most noticeable when modern sparse CGS implementations are used with a large number of topics K, since these methods can be executed in time sublinear in K, unlike the dense CGS update. In this section, we study the trade-off between the improved performance and the extra cost of our methods, as compared with the standard estimators. In particular, we are interested in examining the performance of our methods when paired with the fastest known CGS implementation, WarpLDA.

In most applications of topic models, especially when dealing with large-scale data, LDA inference is run on a given corpus for a number of iterations and a single point estimate of the ϕ and θ parameters is calculated at the end of the procedure, so we focus on that scenario for our experiment. When considering performance versus running time, based on our previous experiments we expect CGS_p to attain higher log likelihood values, but with the extra overhead of the estimators calculation, while the standard CGS estimators will need more iterations to converge to those same log likelihood values, without the overhead. Therefore, we are interested to see if, and at which point during the process of training an LDA model, employing the CGS_p estimators will be more beneficial time-wise than using the standard estimators.

In this experiment, reported in Figure 6, we run WarpLDA on the TASA data set for $K = 50, 100, 500, 1000$ and plot the training data log likelihood of CGS and CGS_p against time duration. In Appendix F we report the relevant results for the rest of the data sets. The time duration for each point in the plot corresponds to the running time of WarpLDA for different numbers of iterations (50, 100, 200, 500, 1000, 2000, 5000), plus the overhead needed to compute each of the estimators at that timestep. We observe that in the early iterations, while LDA is still converging, the standard estimators attain a higher log likelihood value faster than the CGS_p ones for $K = 500, 1000$ (the two estimators are almost identical during burn-in for $K = 50, 100$). This tendency is reversed after convergence of the procedure, with CGS_p attaining a higher log likelihood value faster than CGS for all topic configurations. In most cases we are interested in obtaining an estimate of the LDA parameters well after convergence, so we consider that the CGS_p estimators are to be preferred over the standard CGS ones, even under limited time resources. We also see that convergence takes longer with more topics, and the time before CGS_p overtakes CGS is correspondingly longer, but the improvement in log-likelihood for CGS_p over CGS becomes greater.

6. Multi-Label Learning Experiments—Prior-LDA

Another important context for comparing different approaches to LDA prediction is in a multi-label, supervised setting. Here, we considered a multi-label learning extension of LDA—Prior-LDA—and used it as a basis for comparisons of model predictions. Following the same organization as the previous section, we present the data sets, the evaluation measures, the experimental setup, and lastly the results with a discussion of their implications.

	Documents			Labels			
Data set	D_{Train}	D_{Test}	$\overline{N_d}$	K	$\overline{\lvert\mathcal{K}_d\rvert}$	Avg. Freq.	V
Delicious	12,910	3,181	16.72	983	19.06	250.34	500
BioASQ	20,000	10,000	113.55	1,032	9.05	174.97	135,186
Bibtex	4,880	2,515	54.06	159	2.38	73.05	1,806
Bookmarks	70,285	17,570	124.26	208	2.03	682.90	2,136

Table 5: Statistics for the data sets used in the supervised learning experiments. Column "Avg. Freq." refers to the average label frequency. All figures concerning labels and word types are given for the respective training sets.

6.1 Data Sets

In this series of experiments we kept the BioASQ data set and used three additional labeled ones: Delicious, Bibtex and Bookmarks. Table 5 presents their main statistics. These data sets were chosen as representative of the diversity of real-world data, where there are often many labels, and sometimes not many word tokens per training instance.

Delicious, Bibtex and Bookmarks[7], are three widely used multi-label data sets (Katakis et al., 2008; Tsoumakas et al., 2008). We did not perform any further preprocessing of these data sets. A notable aspect of Delicious is that it contains very few word tokens per instance both in the training and test sets, making accurate classification difficult. Concerning the BioASQ data set, we used the same corpus as for the unsupervised learning experiments and the same preprocessing procedure. We did not use the entire labelset of the original data set, filtering out labels appearing in less than 40 instances.

6.2 Evaluation Metrics

We considered three widely-used performance measures: the micro-averaged F1 measure (Micro-F for brevity) and the macro-averaged F1 measure (Macro-F for brevity) and the example-based F1 measure (Tsoumakas et al., 2010). These measures are a weighted function of precision and recall, and emphasize the need for a model to perform well in terms of both of these underlying measures. The Macro-F score is the average of the F1-scores that are achieved across all labels, while the example-based F1 score is the respective average of the F1 score across all test documents. The Micro-F score is the average F1 score weighted by each label's frequency. Macro-F tends to emphasize performance on infrequent labels, while Micro-F tends to emphasize performance on frequent labels.

6.3 Setup of Prior-LDA (CGS and CVB0)

The Prior-LDA model takes into account the relative label frequencies in the corpus. Test documents are biased to assign words to more frequent labels by using a non-symmetric

7. http://mulan.sourceforge.net/datasets-mlc.html

vector α_k on the θ distributions. During training we kept the α_k parameter symmetrical and set it to:

$$\alpha_k = \frac{50}{K} \tag{58}$$

while during prediction we incorporated the label frequencies by setting it to:

$$\alpha_k = 50 \cdot \frac{f_k}{\sum f_k} + \frac{30}{K} \tag{59}$$

with f_k standing for the frequency of label k. The β_v parameter was set to 0.1 across all data sets. For all data sets, we calculated performance under two different configurations:

- First, we wanted to emulate a scenario where only one sample can be afforded both for training and testing. In this case, during training we ran only one Markov chain and took a single estimate from the same chain for both ϕ and ϕ^p. Similarly, during testing, we ran one chain with the estimated ϕ and one with the estimated ϕ^p to obtain a single estimate for θ and θ^p from each chain. In this manner, we obtained the predictions for all combinations of ϕ and θ estimators.

- In the second configuration, we used 5 Markov chains and 30 samples from each chain, both during training and testing, following the same approach as above to obtain predictions from all four combinations of the ϕ and θ estimators. In this case, the motivation was to examine performance with full convergence of the relevant estimators.

In both training and prediction, we used a burn-in period of 50 iterations and a lag of 5 iterations between each sample. All samples from all chains were averaged to obtain the respective parameter estimates for each method.

For CVB0 we used the same setup as the one described above. However, as CVB0 is a deterministic algorithm that does not benefit greatly from averaging samples, in the second configuration each chain was initialized with a different random order of the documents and a different random initialization of the γ parameters.

Finally, in order to obtain a hard assignment of labels to documents from the θ distributions, we used the Meta-Labeler approach (Tang et al., 2009). In particular, we used a linear regression model to predict the number of labels per instance. To train this model, the same feature space as for the LLDA models was employed and the LibLinear package (Fan et al., 2008) was used for the implementation.

6.4 Results and Discussion

Tables 6 – 8 show the Micro-F, the Macro-F and the example-based F1 results respectively for all algorithms on the four data sets. We additionally show the average rank of each model, in terms of how it performs among the models on average across the four data sets. All results represent averages over five runs of the respective Gibbs sampler (or CVB0 algorithm). Also, to test statistical significance of the differences, we performed a one sample z-test for proportions at a significance level of 0.05.

Method	Delicious	BioASQ	Bibtex	Bookmarks	Avg Rank
			1 MC × 1 sample		
CGS	0.2126±0.0034△	0.3075±0.0041△	0.2388±0.0024△	0.1546±0.0042△	4.5
$\phi + \theta^p$	0.2472±0.0038△	0.3616±0.0037△	**0.2985±0.0029**	0.1696±0.0036△	2.5
$\phi^p + \theta$	0.2127±0.0027△	0.3081±0.0038△	0.2337±0.0026△	0.1542±0.0046△	4.5
CGS_p	0.2531±0.0032△	0.3623±0.0041△	0.2900±0.0021	0.1712±0.0027△	2.25
CVB0	**0.2761±0.0015**	**0.4122±0.0032**	0.2907±0.0014	**0.1810±0.0032**	1.25
			5 MC × 30 samples		
CGS	0.2699±0.0017△	0.4509±0.0021△	0.2873±0.0011△	0.1819±0.0011△	4.25
$\phi + \theta^p$	0.3184±0.0015	0.4628±0.0019	**0.3704±0.0013**	0.2035±0.0018	1.75
$\phi^p + \theta$	0.2699±0.0019△	0.4509±0.0024△	0.2842±0.0015△	0.1820±0.0023△	3.75
CGS_p	**0.3189±0.0012**	**0.4633±0.0009**	0.36277±0.0005	**0.2039±0.0024**	1.25
CVB0	0.2773±0.0008△	0.4125±0.0006△	0.2910±0.0002△	0.1811±0.0004△	4

Table 6: Results for the Micro-F1 measure comparing the Prior-LDA models. A △ symbol represents statistically significant difference compared to the best performing model, at a level of 0.05.

First, let us consider the results for when only a single sample was used. A first remark is that CGS_p and $\phi + \theta^p$ are significantly better than CGS and $\phi^p + \theta$ over all data sets and evaluation measures, confirming θ^p's superiority over θ. Secondly, CVB0 has a clear advantage over the rest of the methods, in the majority of the data sets and evaluation measures. Only CGS_p and $\phi + \theta^p$ manage to outperform it in three out of twelve cases (four data sets and three evaluation measures).

At first glance, these results suggest that, when only one sample from one Markov chain can be afforded, CVB0 should be perhaps considered first, followed by CGS_p and $\phi + \theta^p$ as alternatives. In order to more carefully evaluate the two algorithms for scenarios where computational time is a critical parameter, in Appendix G we performed a short investigation in order to study if and when CGS_p performance surpasses that of CVB0. Specifically, we plotted performance (Micro-F, Macro-F, example-based F1) against the number of samples taken, for all four data sets. The results show that CGS_p manages to outperform CVB0 in seven out of twelve cases, by taking only two samples from one Markov chain, while, after taking ten samples from one chain, CGS_p outperforms CVB0 in eleven out of twelve cases. Considering that there exist a number of approaches for CGS that exploit sparsity to speed up LDA estimation and prediction (Porteous et al., 2008; Yao et al., 2009; Li et al., 2014), while this is not the case for CVB0, we could claim that in many cases CGS_p is equally fit or even superior to CVB0, in terms of total computational time.[8]

Let us now focus on the scenario where we have averaged over multiple samples and multiple Markov chains to calculate predictions. In this case, model performance shifts in favor of the CGS-based predictions. Whereas the CGS algorithm benefits from averaging samples—due to the fact that each sample is ideally a draw from the posterior distribution—

8. Note that the stochastic variant SCVB0 (Foulds et al., 2013) is much more scalable than CGS in the amount of data, but less scalable in the number of topics due to dense updates.

Method	Delicious	BioASQ	Bibtex	Bookmarks	Avg Rank
			1 MC × 1 sample		
CGS	0.0337±0.0012)△	0.1767±0.0039△	0.1582±0.0033△	0.0928±0.0017△	4.5
$\phi + \theta^p$	**0.0597±0.0020**	0.2160±0.0031△	0.2019±0.0030△	0.1009±0.0023△	2.25
$\phi^p + \theta$	0.0332±0.0015△	0.1769±0.0040△	0.1548±0.0031△	0.0930±0.0019△	4.5
CGS_p	0.0585±0.0018	0.2168±0.0036△	0.1946±0.0036△	0.1015±0.0016△	2.25
CVB0	0.0466±0.0010△	**0.3039**±0.0014	**0.2232±0.0021**	**0.1169±0.0008**	1.5
			5 MC × 30 samples		
CGS	0.0373±0.0006△	0.3011±0.0014△	0.2144±0.0013△	0.1167±0.0009△	4.5
$\phi + \theta^p$	**0.0874±0.0004**	0.3138±0.0016	**0.2693±0.0011**	0.1294±0.0008	1.5
$\phi^p + \theta$	0.0362±0.0011△	0.3016±0.0015△	0.2086±0.0017△	0.1168±0.0013△	4.5
CGS_p	0.0855±0.0012	**0.3146±0.0016**	0.2579±0.0021	**0.1299±0.0011**	1.5
CVB0	0.0481±0.0003△	0.3075±0.0005	0.2237±0.0008△	0.1173± 0.0006△	3

Table 7: Results for the Macro-F measure. A △ symbol represents statistically significant difference compared to the best performing model, at a level of 0.05.

CVB0 is deterministic, and achieves very small improvements (if any), presumably since it tends to converge to the same specific maximum. When five chains are used, CGS_p and $\phi + \theta^p$ are significantly better than CVB0 in all data sets and for all evaluation measures. CGS and $\phi^p + \theta$ on the other hand, seem to achieve equivalent performance to CVB0 overall.

Among the CGS-based methods, we observe that, even if θ and θ^p should theoretically converge to the same solution (at least in an unsupervised learning context) even after a total of 150 samples (30 samples from five chains) their difference is significantly large for all evaluation measures and all data sets (the BioASQ data set represents partly an exception with smaller differences between the four methods). Another important note concerns the comparison between models predicting with ϕ^p and those predicting with ϕ. Results are mixed in this case, with models predicting with ϕ^p having a slight, but not statistically significant advantage.

Overall, the supervised experiments show a consistent advantage of the CGS_p and $\phi + \theta^p$ estimators over the respective standard CGS estimator. Also, CGS_p and $\phi + \theta^p$ are competitive with CVB0 in scenarios where only one sample can be afforded. Finally, when averaging over many samples and Markov chains, CGS_p and $\phi + \theta^p$ distinctly outperform CVB0.

7. Conclusions

In this work we proposed a novel method for accurately estimating the document-wise and topic-wise LDA parameters from collapsed Gibbs samples by implicitly averaging over many samples, leading to improved estimation with little extra computational cost. Our algorithm can be interpreted as using soft clustering information, as in the CVB0 collapsed variational inference algorithm, but in a Markov chain Monte Carlo setting. This allows us to use all of

Method	Delicious	BioASQ	Bibtex	Bookmarks	Avg Rank
			1 MC × 1 sample		
CGS	0.2047±0.0019△	0.2930±0.0037△	0.2546±0.0031△	0.2007±0.0031△	4.25
$\phi + \theta^p$	0.2389±0.0023△	0.3400±0.0042	**0.3127±0.0036**	0.2152±0.0032△	2.5
$\phi^p + \theta$	0.2035±0.0014△	0.2942±0.0034△	0.2472±0.0028△	0.1999±0.0029△	4.75
CGS_p	0.2444±0.0019△	0.3402±0.0037	0.3048±0.0033	0.2166±0.0031△	2
CVB0	**0.2672±0.0015**	**0.3504±0.0018**	0.2995±0.0017△	**0.2225±0.0021**	1.5
			5 MC × 30 samples		
CGS	0.2625±0.0007△	0.4180±0.0029	0.2958±0.0023△	0.2226±0.0005△	4.25
$\phi + \theta^p$	**0.3118±0.0011**	0.4269±0.0020	**0.3807±0.0011**	0.2424±0.0016	1.5
$\phi^p + \theta$	0.2610±0.0012△	0.4181±0.0023	0.2922±0.0016△	0.2229±0.0007△	4
CGS_p	0.3086±0.0007	**0.4276±0.0025**	0.3733±0.0021	**0.2431±0.0009**	1.5
CVB0	0.2690±0.0004△	0.3568±0.0021△	0.2997±0.0005△	0.2227±0.0003△	3.75

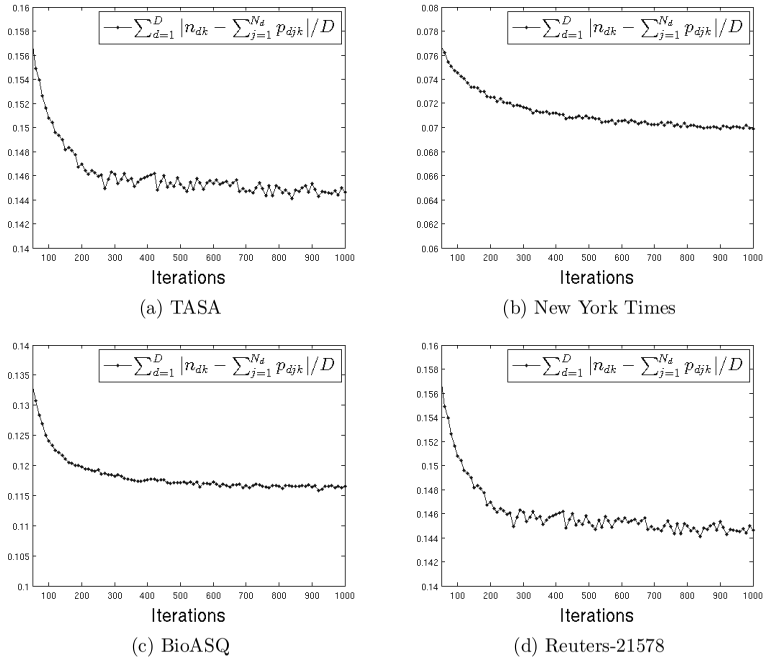
Table 8: Results for the example-based F-measure. A △ symbol represents statistically significant difference compared to the best performing model, at a level of 0.05.

the uncertainty information encoded in the dense Gibbs sampling transition probabilities to recover the parameters from a CGS sample, while still leveraging the sparsity inherent in the CGS algorithm during the training process. We extensively investigated the performance of the proposed estimators for both unsupervised and supervised topic models, and with comparison to CVB0. Our results demonstrate that our θ^p and ϕ^p estimators are more effective than the standard estimator in terms of perplexity, and that θ^p also improves multi-label classification performance for the supervised Prior-LDA model.

Our experiments showed an advantage of our CGS-based estimators over CVB0 in the majority of cases, and also revealed the consequences of the deterministic nature of that algorithm. In the unsupervised scenario, CVB0 has a clear advantage for LDA models with few topics, while this phenomenon is reversed as the posterior landscape becomes more complicated with a greater number of topics. In the multi-label learning case, CVB0 has the upper hand when only one sample can be afforded, while it is decisively outperformed by CGS_p when averaging over multiple samples. The success of our methods illustrates the value of averaging over multiple MCMC samples for LDA, even when only a point estimate is required. In future work, we anticipate that our ideas can be adapted to other models with collapsed representations, including latent variable blockmodels for social networks and other network data (Kemp et al., 2006), and hidden Markov models (Goldwater and Griffiths, 2007).

Acknowledgments

The authors would like to thank the BioASQ team for kindly providing the BioASQ corpus that was used throughout the experiments. Furthermore, we would like to thank the anonymous reviewers of the previous versions of this paper for their elaborate and



(a) TASA (b) New York Times

(c) BioASQ (d) Reuters-21578

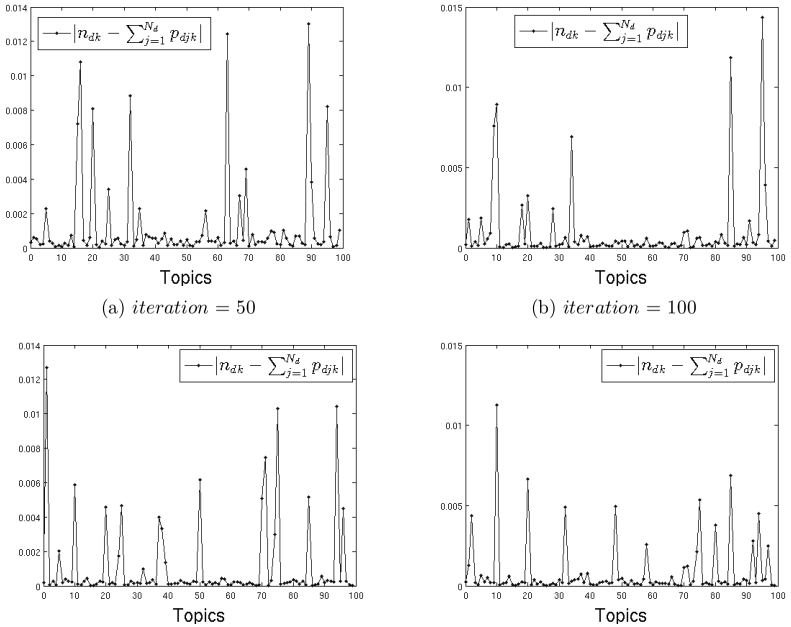
Figure 7: Hard and soft document-topic counts absolute difference against the number of iterations. The counts are normalized to 1 before taking their difference, hence the difference taking values between 0 and 1.

constructive comments that greatly helped in improving and extending this work. Yannis Papanikolaou and Grigorios Tsoumakas have been partially funded by Atypon Systems Inc., Santa Clara, California, USA.

Appendix A. Hard Counts vs Soft Counts

Algorithmically (refer to Algorithm 1), the difference between the θ and θ^p estimators lies in the difference between the hard counts n_{dk} and the soft counts given by $\sum_{j=1}^{N_d} p(d, j, k)$, p being the sampling update equation of CGS given in equation 1. In this experiment, we examine how similar are the two counts. For this reason, we have run CGS on the four data sets setting $K = 100$, $\alpha = 0.1$, $\beta = 0.01$ for 50 iterations and calculated the sum of the absolute differences between the two counts, or, in other words:

$$\frac{\sum_{d=1}^{D} \sum_{k=1}^{K} |n_{dk} - \sum_{j=1}^{N_d} p(d, j, k)|}{D} \tag{60}$$



(a) *iteration* = 50 (b) *iteration* = 100

(c) *iteration* = 500 (d) *iteration* = 1000

Figure 8: Absolute difference between the hard and the soft counts across the K different topics, for iteration 50, 100 and 1000, for the first document of the TASA collection.

Figure 7 shows the absolute sum of differences across the number of iterations. We can observe a steady difference among the two counts, that diminishes but does not go away with the number of iterations. Furthermore, in Figure 8 we depict for the TASA data set and the first document of the collection, the absolute difference between the hard and the soft counts across the K different topics, for iteration 50, 100 and 1000. Similarly to our previous observations, we see that the two counts are similar, but present a small but steady difference across topics, for all iterations.

Appendix B. Bounding the Approximation for ϕ_p

We wish to bound the error of the approximation step used in the derivation for the ϕ_p technique, given in Equation 47:

$$E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big]$$
$$\approx E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big] .$$

Assume that all $n_k^{(i)} > 0$. Then, starting from the left hand side of the above, we have:

$$\begin{aligned}
&E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big]\\
=&E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\times\frac{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big]\\
=&\sum_{\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)})}\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\times\frac{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\\
&\qquad\times p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))\\
\geq&\sum_{\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)})}\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\times\frac{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+1+\sum_{v'=1}^{V}\beta_{v'}}\\
&\qquad\times p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))\\
=&\frac{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+1+\sum_{v'=1}^{V}\beta_{v'}}E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big]\\
=&\frac{n_k^{(i)}+\sum_{v'=1}^{V}\beta_{v'}}{n_k^{(i)}+\sum_{v'=1}^{V}\beta_{v'}+1}E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big] .
\end{aligned} \tag{61}$$

Similarly, for a lower bound, we have:

$$\begin{aligned}
&E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big]\\
\leq&\sum_{\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)})}\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\times\frac{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}-1+\sum_{v'=1}^{V}\beta_{v'}}\\
&\qquad\times p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))\\
=&\frac{n_k^{(i)}+\sum_{v'=1}^{V}\beta_{v'}}{n_k^{(i)}+\sum_{v'=1}^{V}\beta_{v'}-1}E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big] .
\end{aligned} \tag{62}$$

Appendix C. Empirical validation of the approximation for ϕ^p

When deriving our ϕ^p estimator, in order to prove that ϕ^p represents the expected value of the standard CGS ϕ estimator, we take the following approximation: we consider that the denominator of Equation 45 is not changing, between two adjacent Gibbs samples. Since z_{djk} is within ± 1 of $z^{(i)}_{djk}$ when taking adjacent Gibbs samples and in a typical corpus $n_k >> 1$, we can assume that $n_k \simeq n_k \pm 1$. Using this approximation, we are able to consider that the denominator in the left hand of Equation 47 is a constant when taking an adjacent sample. Hence, the expected value of the fraction will be the expected value of the numerator, divided by the denominator.

In order to empirically validate this argument, we considered the following experiment: We have run CGS on the New York Times data set for 100 iterations and for $K = 100$. Next we computed explicitly, the left and the right sides of Equation 47 for all topics and word types. Subsequently, we have measured the accuracy of the approximation of Equation 47 for a varying size of documents (and therefore n_k), by plotting the absolute difference |left hand side - right hand side| for each topic, as a function of the size of D, or $\frac{|a-b|}{K}$ where

$$a = E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}\Big[\frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} z_{djk}+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z_{djk}+\sum_{v'=1}^{V}\beta_{v'}}\Big]$$

$$b = \frac{\sum_{d=1}^{D_{Train}}\sum_{j:w_{d,j}=v} E_{p(\mathbf{z}|\mathbf{w},\boldsymbol{\beta},\mathbf{z}\in Z_{Gibbs}(\mathbf{z}^{(i)}))}[z_{djk}]+\beta_v}{\sum_{d=1}^{D_{Train}}\sum_{j=1}^{N_d} z^{(i)}_{djk}+\sum_{v'=1}^{V}\beta_{v'}} \quad (63)$$

Figure 9 plots the results from the above experiment. As expected, by increasing the considered subset of documents and therefore n_k, the error of the approximation for each topic approaches zero.

Appendix D.

Figures 10 – 13 present additional results for the experiment of Section 5.3, using different topic configurations ($K = 20$, 50, 500) for each of the four data sets (BioASQ, New York Times, Reuters-21578, TASA) respectively. These results are aligned with the conclusions made in Section 5.3 and are reported here as further empirical validation of the theory behind CGS$_p$.

Appendix E.

In this section we report in Figure 14, additionally to the results of Section 5.5, the perplexity across different topic configurations for VB against CVB0, CGS and CGS$_p$. We employed the lda-c package, publicly available at `https://github.com/blei-lab/lda-c`, keeping all parameters to default for the Variational EM procedure.

Apart from the New York Times data set, where VB manages to outperform CVB0 for $K > 200$ and the CGS algorithms for $K = 200$, the algorithm performs worse than other methods in all other cases. These results seem to be aligned with previous results in the literature (Teh et al., 2006; Asuncion et al., 2009).

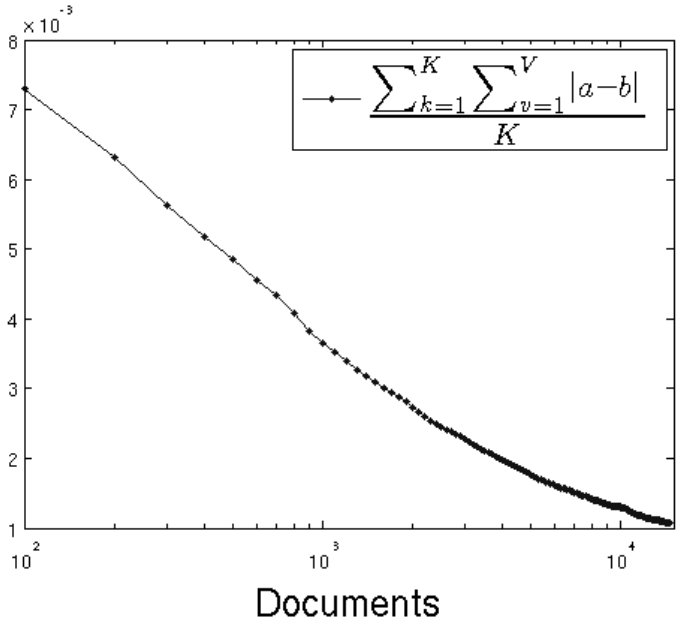


Figure 9: Absolute difference, per topic, between the left and right hand of Equation 47. As the number of documents increases, the difference approaches 0. Here, a, b are given in Equation 63 and the sum is divided by K in order to obtain the per-topic differences.

Appendix F.

We report in Figures 15 - 17 additional results for the rest of the data sets, for our experiment of section 5.8, regarding the run-time overhead of the CGS$_p$ estimators. The results validate the conclusions drawn in section 5.8.

Appendix G.

We report the results of an additional experiment that we performed in order to investigate how many samples CGS$_p$, $\phi + \theta^p$ and $\phi^p + \theta$ would require to outperform CVB0 (Figures 18 – 21). The same parameter setup was used as in the rest of the multi-label experiments of Section 6.

References

L. AlSumait, D. Barbara, and C. Domeniconi. On-line LDA: Adaptive Topic Models for Mining Text Streams with Applications to Topic Detection and Tracking. In *Data Mining,*

2008. ICDM '08. Eighth IEEE International Conference on, pages 3–12, Dec 2008.

Arthur Asuncion. Approximate Mean Field for Dirichlet-Based Models. In *ICML Workshop on Topic Models*. Citeseer, 2010.

Arthur Asuncion, Max Welling, Padhraic Smyth, and Yee Whye Teh. On Smoothing and Inference for Topic Models. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 27–34, Arlington, Virginia, United States, 2009. AUAI Press. ISBN 978-0-9749039-5-8.

Arthur U Asuncion. *Distributed and accelerated inference algorithms for probabilistic graphical models*. PhD thesis, University of California, Irvine, 2011.

Georgios Balikas, Ioannis Partalas, Axel-Cyrille Ngonga Ngomo, Anastasia Krithara, and Georgios Paliouras. Results of the BioASQ Track of the Question Answering Lab at CLEF 2014. In *Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014.*, pages 1181–1193, July 2014.

David M. Blei and Jon D. McAuliffe. Supervised Topic Models. In *NIPS*, 2007.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003. ISSN 1532-4435.

Liangliang Cao and Li Fei-Fei. Spatially Coherent Latent Topic Model for Concurrent Segmentation and Classification of Objects and Scenes. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, Oct 2007. doi: 10.1109/ICCV.2007.4408965.

Jianfei Chen, Kaiwei Li, Jun Zhu, and Wenguang Chen. Warplda: a cache efficient o (1) algorithm for latent dirichlet allocation. *Proceedings of the VLDB Endowment*, 9(10): 744–755, 2016.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A Library for Large Linear Classification. *J. Mach. Learn. Res.*, 9:1871–1874, June 2008. ISSN 1532-4435.

J. R. Foulds and P. Smyth. Annealing paths for the evaluation of topic models. In *Proceedings of the Thirtieth Conference Conference on Uncertainty in Artificial Intelligence (UAI)*, 2014.

James R. Foulds. *Latent Variable Modeling for Networks and Text: Algorithms, Models and Evaluation Techniques*. PhD thesis, University of California, Irvine, 2014.

James R. Foulds, Levi Boyles, Christopher DuBois, Padhraic Smyth, and Max Welling. Stochastic Collapsed Variational Bayesian Inference for Latent Dirichlet Allocation. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 446–454, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2174-7. doi: 10.1145/2487575.2487697.

Andre Gohr, Alexander Hinneburg, Rene Schult, and Myra Spiliopoulou. Topic Evolution in a Stream of Documents. In *In SDM*, pages 859–872, 2009.

Sharon Goldwater and Tom Griffiths. A Fully Bayesian Approach to Unsupervised Part-of-Speech Tagging. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 744–751, 2007.

T. L. Griffiths and M. Steyvers. Finding Scientific Topics. *Proceedings of the National Academy of Sciences*, 101(Suppl. 1):5228–5235, April 2004.

Thomas L Griffiths, Mark Steyvers, and Joshua B Tenenbaum. Topics in semantic representation. *Psychological review*, 114(2):211, 2007.

Gregor Heinrich. Parameter Estimation for Text Analysis. Technical report, 2004.

Matthew Hoffman, Francis R. Bach, and David M. Blei. Online Learning for Latent Dirichlet Allocation. In J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 856–864. Curran Associates, Inc., 2010a.

Matthew Hoffman, David M. Blei, and Francis Bach. Online Learning for Latent Dirichlet Allocation. In *NIPS*, 2010b.

R.E. Kass, L. Tierney, and J.B. Kadane. The Validity of Posterior Expansions Based on Laplace's Method. *Bayesian and likelihood methods in statistics and econometrics*, 7:473, 1990.

Ioannis Katakis, Grigorios Tsoumakas, and Ioannis Vlahavas. Multilabel Text Classification for Automated Tag Suggestion. In *Proceedings of the ECML/PKDD 2008 Discovery Challenge*, Antwerp, Belgium, 2008.

Charles Kemp, Joshua B Tenenbaum, Thomas L Griffiths, Takeshi Yamada, and Naonori Ueda. Learning Systems of Concepts with an Infinite Relational Model. In *AAAI*, volume 3, page 5, 2006.

B.J.K. Kleijn and A.W. van der Vaart. The Bernstein-Von-Mises Theorem Under Misspecification. *Electronic Journal of Statistics*, 6:354–381, 2012.

Thomas K. Landauer, Peter W. Foltz, and Darrell Laham. An Introduction to Latent Semantic Analysis. *Discourse Processes*, (25):259–284, 1998.

Aaron Q Li, Amr Ahmed, Sujith Ravi, and Alexander J Smola. Reducing the Sampling Complexity of Topic Models. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 891–900. ACM, 2014.

Andrew Kachites McCallum. MALLET: A Machine Learning for Language Toolkit. http://mallet.cs.umass.edu, 2002a.

Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. 2002b.

T.P. Minka. Estimating a Dirichlet Distribution. Technical report, Microsoft Research, 2000.

Douglas L Nelson, Cathy L McEvoy, and Thomas A Schreiber. The university of south florida free association, rhyme, and word fragment norms. *Behavior Research Methods, Instruments, & Computers*, 36(3):402–407, 2004.

David Newman, Arthur U. Asuncion, Padhraic Smyth, and Max Welling. Distributed Inference for Latent Dirichlet Allocation. In *NIPS*, volume 20, pages 1081–1088, 2007.

Viet-An Nguyen, Jordan L. Boyd-Graber, and Philip Resnik. Sometimes Average is Best: The Importance of Averaging for Prediction using MCMC Inference in Topic Modeling. In *EMNLP*, pages 1752–1757, 2014.

Ian Porteous, David Newman, Alexander Ihler, Arthur Asuncion, Padhraic Smyth, and Max Welling. Fast Collapsed Gibbs Sampling for Latent Dirichlet Allocation. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 569–577, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-193-4. doi: 10.1145/1401890.1401960.

Jonathan K Pritchard, Matthew Stephens, and Peter Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155(2):945–959, 2000.

Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. Labeled LDA: A Supervised Topic Model for Credit Attribution in Multi-labeled Corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 248–256, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. ISBN 978-1-932432-59-6.

Daniel Ramage, Christopher D. Manning, and Susan Dumais. Partially Labeled Topic Models for Interpretable Text Mining. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 457–465, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0813-7. doi: 10.1145/2020408.2020481.

Timothy N. Rubin, America Chambers, Padhraic Smyth, and Mark Steyvers. Statistical Topic Models for Multi-label Document Classification. *Mach. Learn.*, 88(1-2):157–208, July 2012. ISSN 0885-6125. doi: 10.1007/s10994-011-5272-5.

Ruslan Salakhutdinov, Sam Roweis, and Zoubin Ghahramani. Optimization with EM and Expectation-Conjugate-Gradient. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 672–679, 2003.

Alexander Smola and Shravan Narayanamurthy. An Architecture for Parallel Topic Models. *Proceedings of the VLDB Endowment*, 3(1-2):703–710, 2010.

Lei Tang, Suju Rajan, and Vijay K. Narayanan. Large Scale Multi-Label Classification via MetaLabeler. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 211–220, New York, NY, USA, 2009. ACM.

Yee Whye Teh, David Newman, and Max Welling. A Collapsed Variational Bayesian Inference Algorithm for Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pages 1353–1360, 2006.

G. Tsoumakas, I. Katakis, and I. Vlahavas. Effective and Efficient Multilabel Classification in Domains with Large Number of Labels. In *Proc. ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08)*, pages 30–44, 2008.

Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Mining Multi-label Data. In Oded Maimon and Lior Rokach, editors, *Data Mining and Knowledge Discovery Handbook*, chapter 34, pages 667–685. Springer, 2nd edition, 2010.

Hanna M. Wallach, Iain Murray, Ruslan Salakhutdinov, and David Mimno. Evaluation Methods for Topic Models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1105–1112, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553515.

Yu-Xiang Wang, Stephen E. Fienberg, and Alex Smola. Privacy for Free: Posterior Sampling and Stochastic Gradient Monte Carlo. *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 2493–2502, 2015.

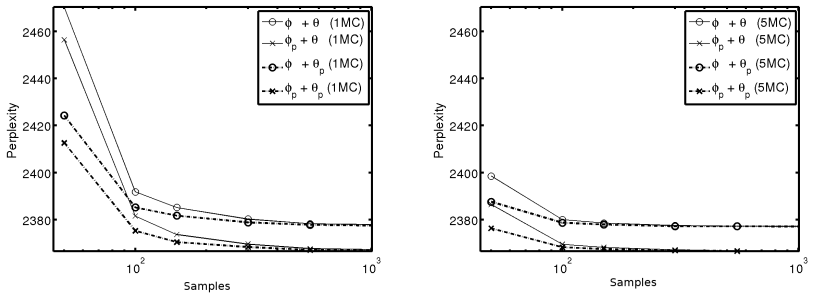
Limin Yao, David Mimno, and Andrew McCallum. Efficient Methods for Topic Model Inference on Streaming Document Collections. In *KDD*, 2009.

Jinhui Yuan, Fei Gao, Qirong Ho, Wei Dai, Jinliang Wei, Xun Zheng, Eric Po Xing, Tie-Yan Liu, and Wei-Ying Ma. Lightlda: Big topic models on modest computer clusters. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1351–1361. ACM, 2015.

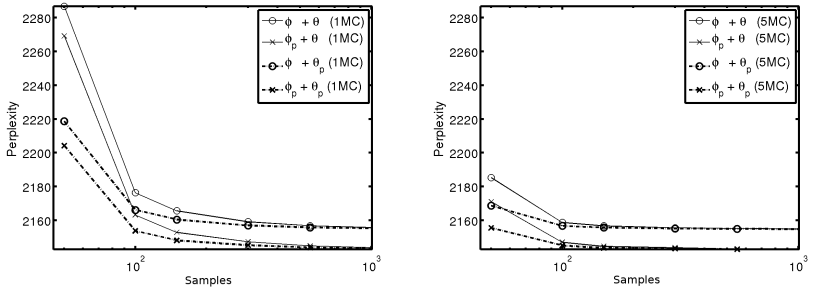
Haizheng Zhang, Baojun Qiu, C.L. Giles, Henry C. Foley, and J. Yen. An LDA-based Community Structure Discovery Approach for Large-Scale Social Networks. In *Intelligence and Security Informatics, 2007 IEEE*, pages 200–207, May 2007. doi: 10.1109/ISI.2007.379553.

Bin Zheng, David C. McLean, and Xinghua Lu. Identifying Biological Concepts from a Protein-Related Corpus with a Probabilistic Topic Model. *BMC bioinformatics*, 7(1):58, 2006.

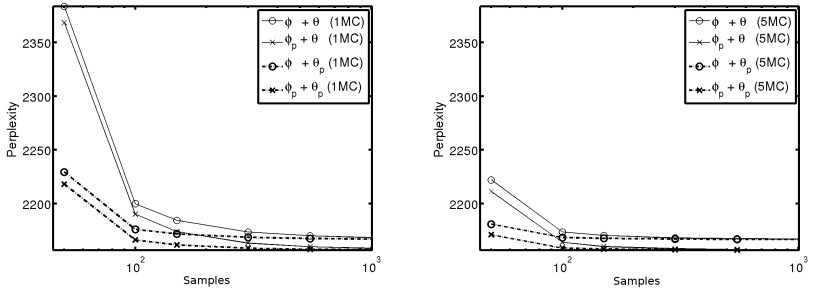
Jun Zhu, Amr Ahmed, and Eric P. Xing. MedLDA: Maximum Margin Supervised Topic Models for Regression and Classification. In *ICML*, 2009.



(a) $K = 20$

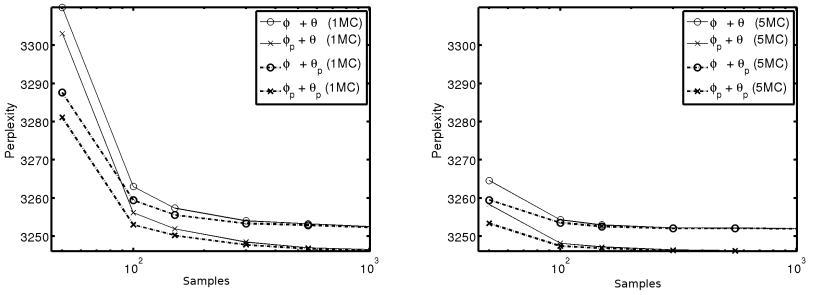


(b) $K = 50$

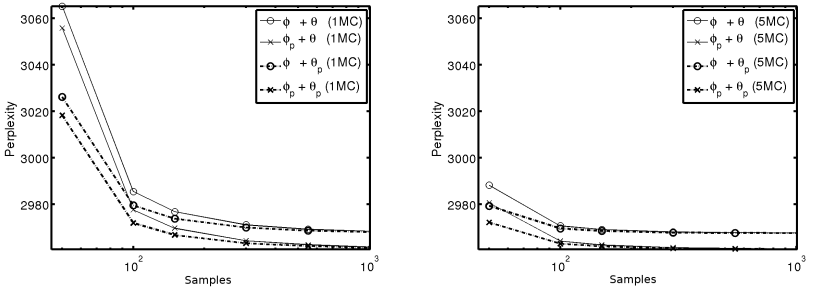


(c) $K = 500$

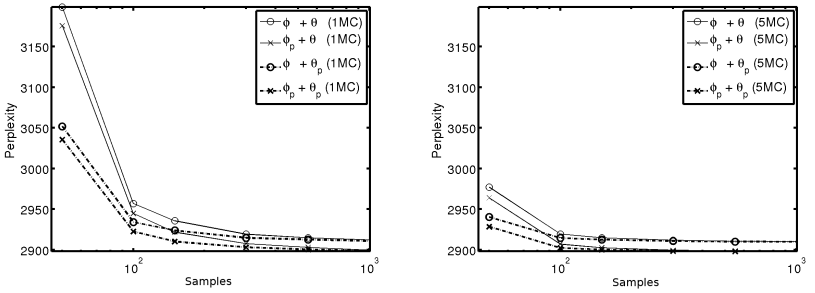
Figure 10: Perplexity against the number of samples for the CGS$_p$ method and standard CGS for the BioASQ data set. Results are taken by averaging over 5 different runs.



(a) $K = 20$

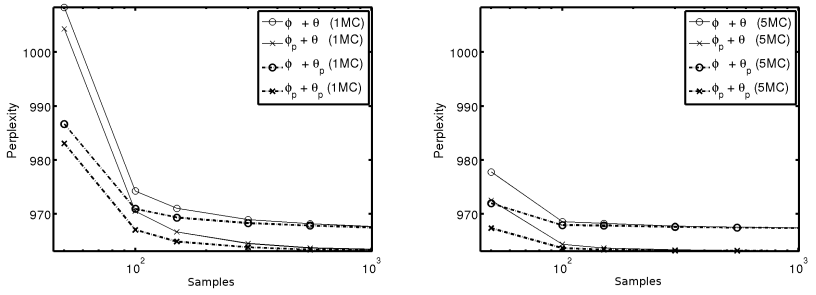


(b) $K = 50$

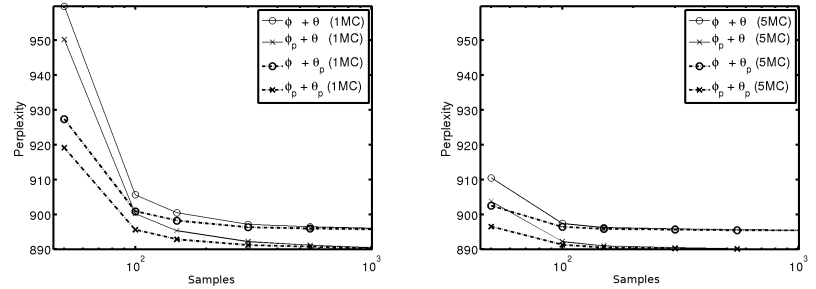


(c) $K = 500$

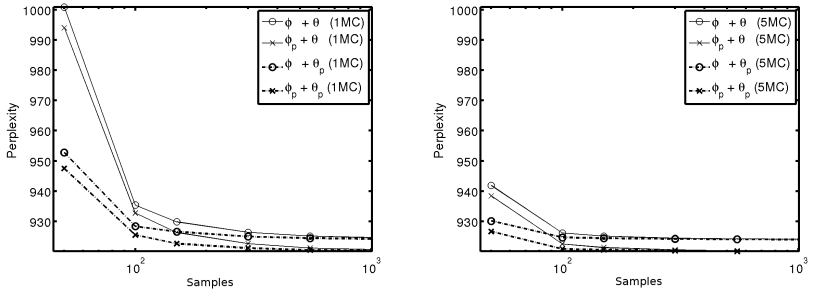
Figure 11: Perplexity against the number of samples for the CGS$_p$ methods and standard CGS for the New York Times data set. Results are taken by averaging over 5 different runs.



(a) $K = 20$

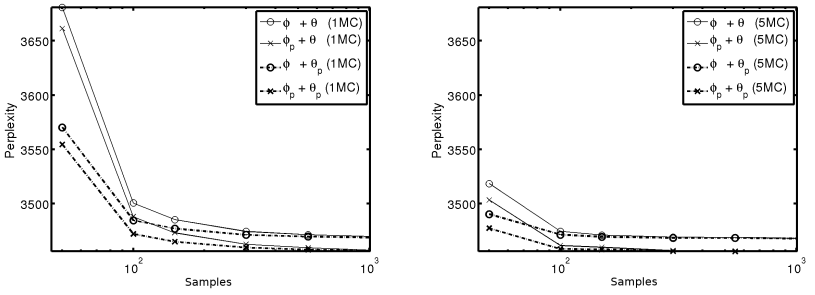


(b) $K = 50$

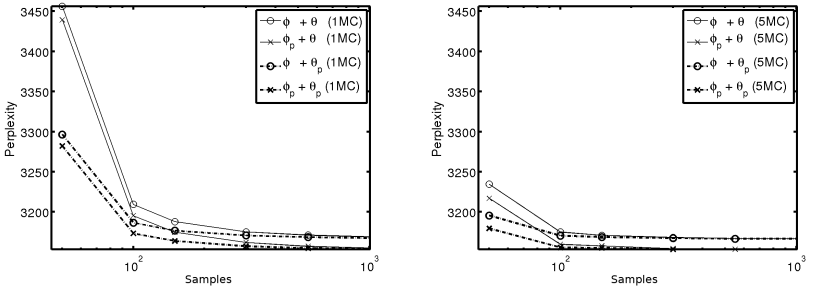


(c) $K = 500$

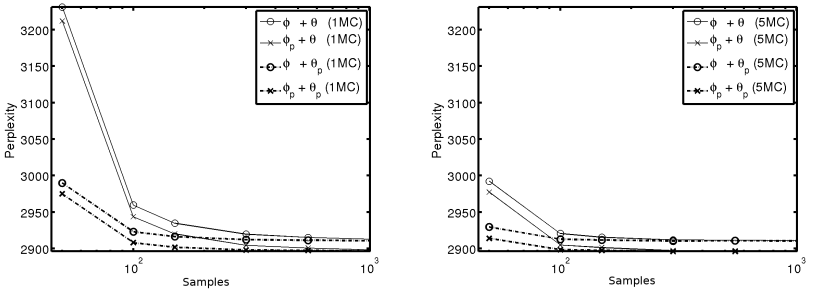
Figure 12: Perplexity against the number of samples for the CGS$_p$ methods and standard CGS for the Reuters-21578 data set. Results are taken by averaging over 5 different runs.



(a) $K = 20$

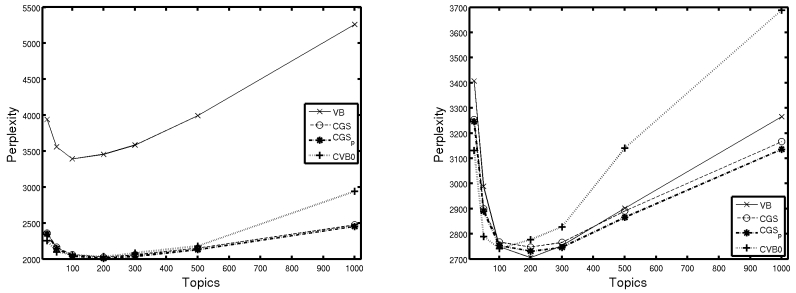


(b) $K = 50$



(c) $K = 500$

Figure 13: Perplexity against the number of samples for the CGS$_p$ methods and standard CGS for the TASA data set. Results are taken by averaging over 5 different runs.



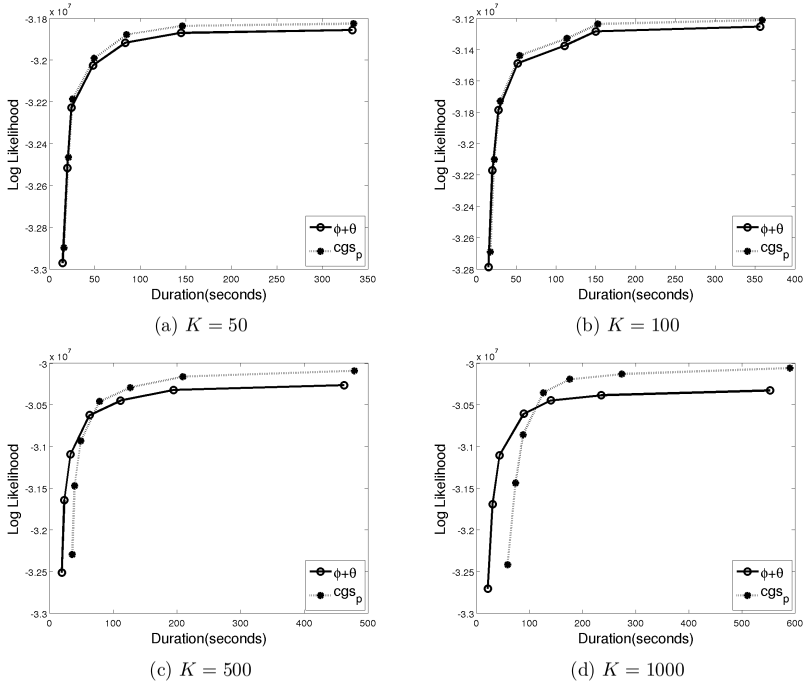
(a) BioASQ

(b) New York Times

(c) Reuters-21578

(d) TASA

Figure 14: Perplexity against number of topics for the CGS$_p$ method, standard CGS, CVB0 and VB. Results are taken by averaging over 5 different runs.



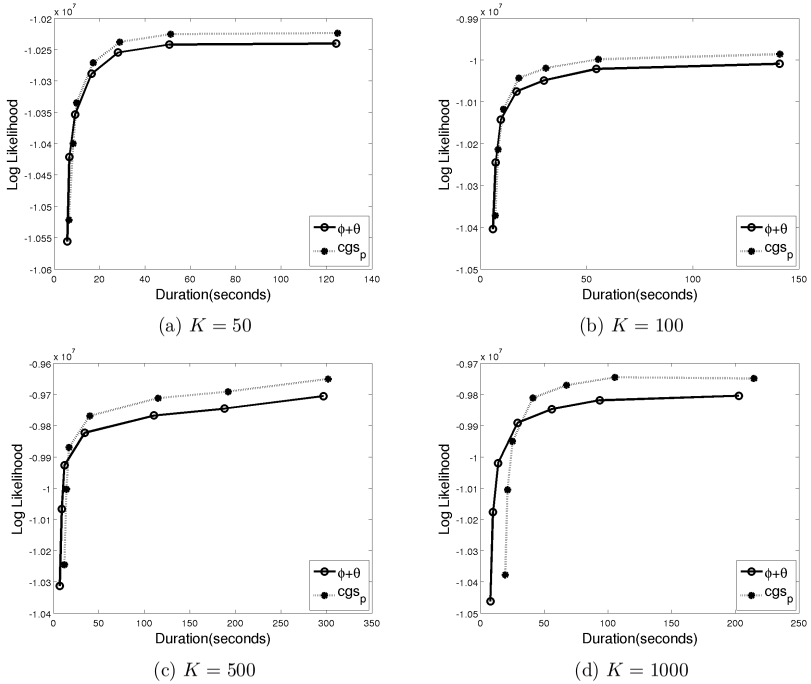
(a) $K = 50$

(b) $K = 100$

(c) $K = 500$

(d) $K = 1000$

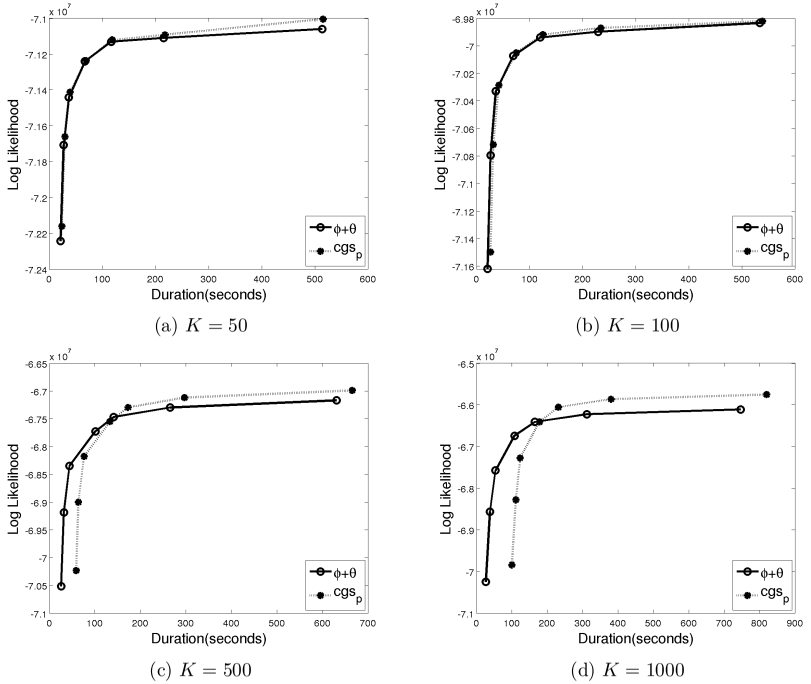
Figure 15: Log Likelihood vs duration for CGS$_p$ against the standard CGS estimators on BioASQ, with WarpLDA. The time duration for each point in the plot corresponds to the running time of WarpLDA for different numbers of iterations (50, 100, 200, 500, 1000, 2000, 5000), plus the overhead needed to compute each of the estimators at that timestep.



(a) $K = 50$ (b) $K = 100$

(c) $K = 500$ (d) $K = 1000$

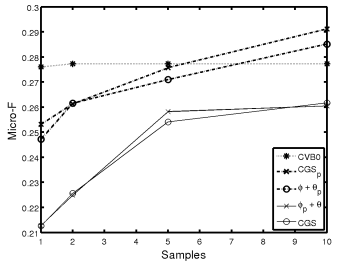
Figure 16: Log Likelihood vs duration for CGS_p against the standard CGS estimators on Reuters-21578, with WarpLDA. The time duration for each point in the plot corresponds to the running time of WarpLDA for different numbers of iterations (50, 100, 200, 500, 1000, 2000, 5000), plus the overhead needed to compute each of the estimators at that timestep.

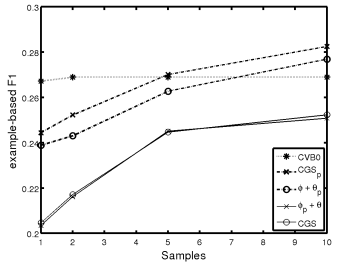


(a) $K = 50$ (b) $K = 100$

(c) $K = 500$ (d) $K = 1000$

Figure 17: Log Likelihood vs duration for CGS_p against the standard CGS estimators on New York Times, with WarpLDA. The time duration for each point in the plot corresponds to the running time of WarpLDA for different numbers of iterations (50, 100, 200, 500, 1000, 2000, 5000), plus the overhead needed to compute each of the estimators at that timestep.





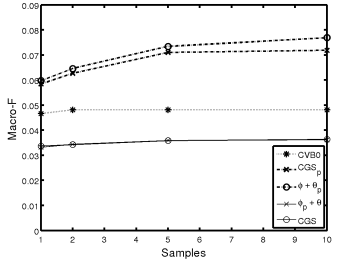
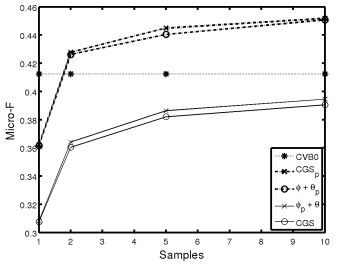
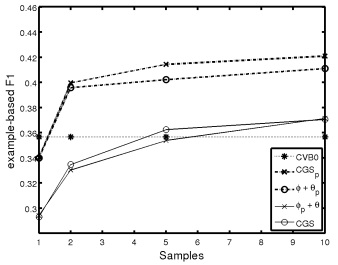


Figure 18: Delicious.





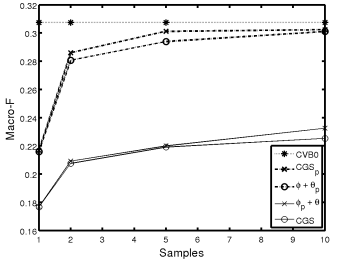
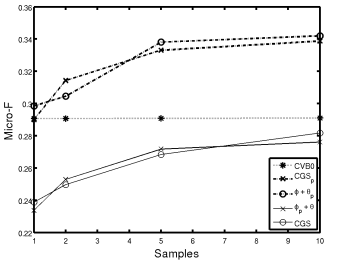
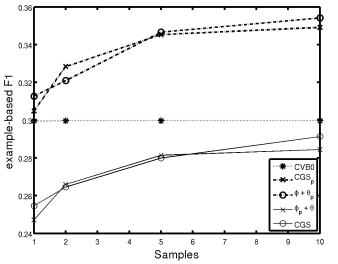


Figure 19: BioASQ.





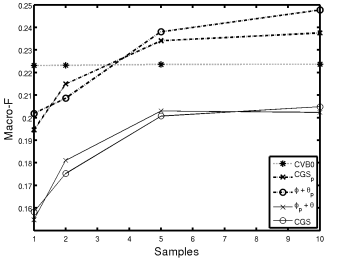
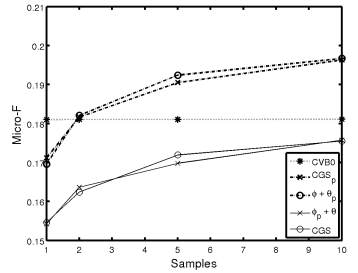
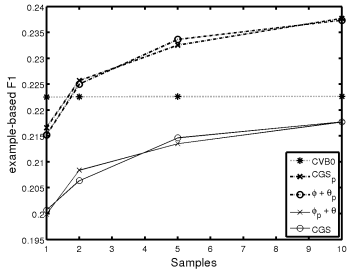


Figure 20: Bibtex.





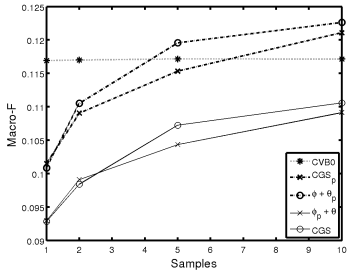


Figure 21: Bookmarks.

Journal of Machine Learning Research 18 (2017) 1-29 Submitted 3/17; Revised 7/17; Published 7/17

Fundamental Conditions for Low-CP-Rank Tensor Completion

Morteza Ashraphijuo ASHRAPHIJUO@EE.COLUMBIA.EDU
Columbia University
New York, NY 10027, USA

Xiaodong Wang WANGX@EE.COLUMBIA.EDU
Columbia University
New York, NY 10027, USA

Editor: Animashree Anandkumar

Abstract

We consider the problem of low canonical polyadic (CP) rank tensor completion. A completion is a tensor whose entries agree with the observed entries and its rank matches the given CP rank. We analyze the manifold structure corresponding to the tensors with the given rank and define a set of polynomials based on the sampling pattern and CP decomposition. Then, we show that finite completability of the sampled tensor is equivalent to having a certain number of algebraically independent polynomials among the defined polynomials. Our proposed approach results in characterizing the maximum number of algebraically independent polynomials in terms of a simple geometric structure of the sampling pattern, and therefore we obtain the deterministic necessary and sufficient condition on the sampling pattern for finite completability of the sampled tensor. Moreover, assuming that the entries of the tensor are sampled independently with probability p and using the mentioned deterministic analysis, we propose a combinatorial method to derive a lower bound on the sampling probability p, or equivalently, the number of sampled entries that guarantees finite completability with high probability. We also show that the existing result for the matrix completion problem can be used to obtain a loose lower bound on the sampling probability p. In addition, we obtain deterministic and probabilistic conditions for unique completability. It is seen that the number of samples required for finite or unique completability obtained by the proposed analysis on the CP manifold is orders-of-magnitude lower than that is obtained by the existing analysis on the Grassmannian manifold.

Keywords: Low-rank tensor completion, canonical polyadic (CP) decomposition, finite completability, unique completability, algebraic geometry, Bernstein's theorem.

1. Introduction

The fast progress in data science and technology has given rise to the extensive applications of multi-way datasets, which allow us to take advantage of the inherent correlations across different attributes. The classical matrix analysis limits its efficiency in exploiting the correlations across different features from a multi-way perspective. In contrast, analysis of multi-way data (tensor), which was originally proposed in the field of psychometrics and recently found applications in machine learning and signal processing, is capable of taking full advantage of these correlations (De Lathauwer, 2009; Kolda and Bader, 2009; Abraham et al., 2012; Grasedyck et al., 2013; Muti and Bourennane, 2007; Signoretto et al., 2011). The problem of low-rank tensor completion, i.e., reconstructing a tensor from a subset of its entries given the rank, which is generally NP hard (Hillar and Lim, 2013), arises in compressed sensing (Lim and Comon, 2010; Sidiropoulos and Kyrillidis, 2012; Gandy et al., 2011), visual data reconstruction (Liu et al., 2013, 2016), seismic data processing (Kreimer et al., 2013; Ely et al., 2013), etc. Existing approaches to low-rank data completion mainly focus on convex relaxation of matrix rank (Candès and Recht, 2009; Candès and Tao, 2010; Cai et al., 2010; Ashraphijuo et al., 2016b; Candès et al., 2013; Ashraphijuo et al., 2015) or different convex relaxations of tensor ranks (Gandy et al., 2011; Tomioka et al., 2010; Signoretto et al., 2014; Romera-Paredes and Pontil, 2013). In addition, there are some other works which provide a completion of the given rank using alternating minimization, e.g., for low-CP-rank tensor (Jain and Oh, 2014) and low-TT-rank tensor (Wang et al., 2016).

Tensors consisting of real-world datasets usually have a low rank structure. The manifold of low-rank tensors has recently been investigated in several works (Grasedyck et al., 2013; Ashraphijuo et al., 2016a; Ashraphijuo and Wang, 2017). In this paper, we focus on the canonical polyadic (CP) decomposition (Harshman, 1970; ten Berge and Sidiropoulos, 2002; Carroll and Chang, 1970; Stegeman and Sidiropoulos, 2007) and the corresponding CP rank, but in general there are other well-known tensor decompositions including Tucker decomposition (Kolda, 2001; Grasedyck, 2010; Kressner et al., 2014), tensor-train (TT) decomposition (Oseledets, 2011; Holtz et al., 2012), tubal rank decomposition (Kilmer et al., 2013) and several other methods (De Lathauwer et al., 2000; Papalexakis et al., 2012). Note that most existing literature on tensor completion based on various optimization formulations use CP rank (Gandy et al., 2011; Krishnamurthy and Singh, 2013).

Deterministic conditions on the locations of the sampled entries (sampling pattern) are obtained through algebraic geometry analyses on Grassmannian manifold that lead to finite/unique solutions to the matrix completion problem (Pimentel-Alarcón et al., 2016d). Also, deterministic conditions on the sampling patterns have been studied for subspace clustering in (Pimentel-Alarcón et al., 2016c, 2015, 2016a,b). In this paper, we study the fundamental conditions on the sampling pattern to ensure finite or unique number of completions, where these fundamental conditions are independent of the correlations of the entries of the tensor, in contrast to the common assumption adopted in literature such as incoherence. Given the rank of a matrix, Pimentel-Alarcón *et. al.* in (Pimentel-Alarcón et al., 2016d) obtains such fundamental conditions on the sampling pattern for finite completability of the matrix. Previously, we treated the same problem for multi-view matrix (Ashraphijuo et al., 2017c,b), tensor given its Tucker rank (Ashraphijuo et al., 2016a, 2017a), and tensor given its TT rank(Ashraphijuo and Wang, 2017). In this paper, the structure of the CP decomposition and the geometry of the corresponding manifold are investigated to obtain the fundamental conditions for finite completability given its CP rank. These recently developed algebraic geometry analyses can also be used to approximate the rank of partially sampled data (Ashraphijuo et al., 2017d).

To emphasize the contribution of this work, we highlight the differences and challenges in comparison with the Tucker and TT tensor models. In CP decomposition, the notion of tensor multiplication is different from those for Tucker and TT, and therefore the geometry of the manifold and the algebraic variety are completely different. Moreover, in CP decomposition we are dealing with the sum of several tensor products, which is not the case in Tucker and TT decompositions, and therefore the equivalence classes or geometric patterns that are needed to study the algebraic variety are different. Moreover, CP rank is a scalar and the ranks of matricizations and unfoldings are not given in contrast with the Tucker and TT models.

Let $\mathcal{U}$ denote the sampled tensor and Ω denote the binary sampling pattern tensor that is of the same dimension and size as $\mathcal{U}$. The entries of Ω that correspond to the observed entries of $\mathcal{U}$ are equal to 1 and the rest of the entries are set as 0. Assume that the entries of $\mathcal{U}$ are sampled

©2017 Morteza Ashraphijuo, and Xiaodong Wang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/17-189.html.

independently with probability p. This paper is mainly concerned with treating the following three problems.

Problem (i): Given the CP rank, characterize the necessary and sufficient conditions on the sampling pattern Ω, under which there exist only finitely many completions of $\mathcal{U}$.

We consider the CP decomposition of the sampled tensor, where all rank-1 tensors in this decomposition are unknown and we only have some entries of $\mathcal{U}$. Then, each sampled entry results in a polynomial such that the variables of the polynomial are the entries of the rank-1 tensors in the CP decomposition. We propose a novel analysis on the CP manifold to obtain the maximum number of algebraically independent polynomials, among all polynomials corresponding to the sampled entries, in terms of the geometric structure of the sampling pattern Ω. We show that if the maximum number of algebraically independent polynomials is a given number, then the sampled tensor $\mathcal{U}$ is finitely completable. Due to the fundamental differences between the CP decomposition and the Tucker or TT decomposition, this analysis is completely different from our previous works (Ashraphijuo et al., 2016a; Ashraphijuo and Wang, 2017). Moreover, note that our proposed algebraic geometry analysis on the CP manifold is not a simple generalization of the existing analysis on the Grassmannian manifold (Pimentel-Alarcón et al., 2016d) even though the CP decomposition is a generalization of rank factorization of a matrix, as almost every step needs to be developed anew.

Problem (ii): Characterize conditions on the sampling pattern to ensure that there is exactly one completion for the given CP rank.

Similar to Problem (i), our approach is to study the algebraic independence of the polynomials corresponding to the samples. We exploit the properties of a set of minimally algebraically dependent polynomials to add additional constraints on the sampling pattern such that each of the rank-1 tensors in the CP decomposition can be determined uniquely.

As we will observe later, the deterministic conditions for finite or unique completability are combinatorial in nature, which are hard to verify if they hold true in practice. Therefore, we also provide a probabilistic analysis to verify the validity of such conditions based on the number of samples or the sampling probability. However, we cannot guarantee the validity of the constraints with probability one anymore (i.e., not deterministically anymore) and instead we show that the combinatorial conditions hold true with high probability if the sampling probability is above certain threshold.

Problem (iii): Provide a lower bound on the total number of sampled entries or the sampling probability p such that the proposed conditions on the sampling pattern Ω for finite and unique completability are satisfied with high probability.

We develop several combinatorial tools together with our previous graph theory results in (Ashraphijuo et al., 2016a) to obtain lower bounds on the total number of sampled entries, i.e., lower bounds on the sampling probability p, such that the deterministic conditions for Problems (i) and (ii) are met with high probability. Particularly, it is shown in (Krishnamurthy and Singh, 2013), $\mathcal{O}(nr^{\frac{d-1}{2}}d^2\log(r))$ samples are required to recover the tensor $\mathcal{U} \in \mathbb{R}^{\overbrace{n \times \ldots \times n}^{d}}$ of rank r. Recall that in this paper, we obtain the number samples to ensure finite/unique completability that is independent of the completion algorithm. As we show later, using the existing analysis on the Grassmannian manifold results in $\mathcal{O}(n^{\frac{d+1}{2}}\max\{d\log(n)+\log(r), r\})$ samples to ensure finite/unique completability. However, our proposed analysis on the CP manifold results in $\mathcal{O}(n^2\max\{\log(nrd), r\})$ samples to guarantee the finiteness of the number of completions, which is significantly lower than that given in (Krishnamurthy and Singh, 2013). Hence, the fundamental conditions for tensor completion motivate new optimization formulation to close the gap in the number of required samples.

Tucker decomposition consists of a core tensor multiplied by a matrix along each dimension. TT decomposition of a d-way tensor consists of the train-wise multiplication of a matrix and $d-2$ three-way tensors and finally another matrix. Tucker and TT ranks are defined as the vectors consisting of rank of matricizations and rank of unfoldings, respectively. The structure of the manifold of the tensors with a fixed Tucker (or TT) rank vector is fundamentally different with that of the CP manifold.

Note that the general framework of using rank factorization to study the algebraic independence of the polynomials through counting the number of involved variables in the polynomials is indeed similar to that in (Pimentel-Alarcón et al., 2016d; Ashraphijuo et al., 2016a; Ashraphijuo and Wang, 2017) to some extent. However, since the manifold considered in this paper (CP manifold) is fundamentally different from the Grassmannian or Tucker or TT manifold, all results in this paper are original. In particular, we mention some of the main differences: (i) geometry of the manifold, (ii) the equivalence class for the basis and consequently (iii) the canonical basis, (iv) the structure of the polynomials, etc., are fundamentally different from those in the literature. Hence, the deterministic analyses are fundamentally different (excluding Lemma 7). As a consequence, the probabilistic analysis is also basically different as the combinatorial condition (6) is obtained based on the defined geometry and equivalence class for CP decomposition. However, some lemmas including the generalization of Hall's theorem for bipartite graphs (Lemma 18) or Lemma 17 that applies pigeonhole principle are taken from our previous papers (Ashraphijuo et al., 2016a; Ashraphijuo and Wang, 2017) in probabilistic analysis.

The remainder of this paper is organized as follows. In Section 2, the preliminaries and problem statement are presented. In Section 3, we develop necessary and sufficient deterministic conditions for finite completability. In Section 4, we develop probabilistic conditions for finite completability. In Section 5, we consider unique completability and obtain both deterministic and probabilistic conditions. Some numerical results are provided in Section 6. Finally, Section 7 concludes the paper.

2. Background

2.1 Preliminaries and Notations

In this paper, it is assumed that a d-way tensor $\mathcal{U} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is sampled. Throughout this paper, we use CP rank as the rank of a tensor, which is defined as the minimum number r such that there exist $\mathbf{a}_i^l \in \mathbb{R}^{n_i}$ for $1 \leq i \leq d$ and $1 \leq l \leq r$ and

$$\mathcal{U} = \sum_{l=1}^{r} \mathbf{a}_1^l \otimes \mathbf{a}_2^l \otimes \ldots \otimes \mathbf{a}_d^l, \tag{1}$$

or equivalently,

$$\mathcal{U}(x_1, x_2, \ldots, x_d) = \sum_{l=1}^{r} \mathbf{a}_1^l(x_1)\mathbf{a}_2^l(x_2)\ldots\mathbf{a}_d^l(x_d), \tag{2}$$

where $\otimes$ denotes the tensor product (outer product) and $\mathcal{U}(x_1, x_2, \ldots, x_d)$ denotes the entry of the sampled tensor with coordinates $\vec{x} = (x_1, x_2, \ldots, x_d)$ and $\mathbf{a}_i^l(x_i)$ denotes the x_i-th entry of vector $\mathbf{a}_i^l$. Note that $\mathbf{a}_1^l \otimes \mathbf{a}_2^l \otimes \ldots \otimes \mathbf{a}_d^l \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is a rank-1 tensor, $l = 1, 2, \ldots, r$.

Denote Ω as the binary sampling pattern tensor that is of the same size as $\mathcal{U}$ and $\Omega(\vec{x}) = 1$ if $\mathcal{U}(\vec{x})$ is observed and $\Omega(\vec{x}) = 0$ otherwise. Also define $x^+ \triangleq \max\{0, x\}$.

For a nonempty $I \subset \{1, \ldots, d\}$, define $N_I \triangleq \Pi_{i \in I}\, n_i$ and also denote $\bar{I} \triangleq \{1, \ldots, d\} \backslash I$. Let $\widetilde{\mathbf{U}}_{(I)} \in \mathbb{R}^{N_I \times N_{\bar{I}}}$ be the unfolding of the tensor $\mathcal{U}$ corresponding to the set I such that $\mathcal{U}(\vec{x}) = \widetilde{\mathbf{U}}_{(I)}(\widetilde{M}_I(x_{i_1}, \ldots, x_{i_{|I|}}), \widetilde{M}_{\bar{I}}(x_{i_{|I|+1}}, \ldots, x_{i_d}))$, where $I = \{i_1, \ldots, i_{|I|}\}$, $\bar{I} = \{i_{|I|+1}, \ldots, i_d\}$, $\widetilde{M}_I : (x_{i_1}, \ldots, x_{i_{|I|}}) \to \{1, 2, \ldots, N_I\}$ and $\widetilde{M}_{\bar{I}} : (x_{i_{|I|+1}}, \ldots, x_{i_d}) \to \{1, 2, \ldots, \bar{N}_{\bar{I}}\}$ are two bijective mappings. For $i \in \{1, \ldots, d\}$ and $I = \{i\}$, we denote the unfolding corresponding to I by $\mathbf{U}_{(i)}$ and we call it the i-th matricization of tensor $\mathcal{U}$.

2.2 Problem Statement and A Motivating Example

We are interested in finding necessary and sufficient deterministic conditions on the sampling pattern tensor Ω under which there are infinite, finite, or unique completions of the sampled tensor $\mathcal{U}$ that satisfy the given CP rank constraint. Furthermore, we are interested in finding probabilistic conditions on the number of samples or the sampling probability that ensure the obtained deterministic conditions for finite and unique completability hold with high probability.

To motivate our proposed analysis in this paper on the CP manifold, we compare the following two approaches using a simple example to emphasize the exigency of our proposed analysis: (i) analyzing each of the unfoldings individually, (ii) analyzing based on the CP decomposition.

Consider a three-way tensor $\mathcal{U} \in \mathbb{R}^{2 \times 2 \times 2}$ with CP rank of 1. Assume that the entries $(1,1,1)$, $(2,1,1)$, $(1,2,1)$ and $(1,1,2)$ are observed. As a result of Lemma 11 in this paper, all unfoldings of this tensor are rank-1 matrices. It is shown in Section II of (Ashraphijuo et al., 2016a) that having any 4 entries of a rank-1 2×4 matrix, there are infinitely many completions for it. As a result, any unfolding of $\mathcal{U}$ is infinitely many completable given only the corresponding rank constraint. Next, using the CP decomposition (1), we show that there are only finitely many completions of the sampled tensor of CP rank 1.

Define $\mathbf{a}_1^1 = [x\ x']^\top \in \mathbb{R}^2$, $\mathbf{a}_2^1 = [y\ y']^\top \in \mathbb{R}^2$ and $\mathbf{a}_3^1 = [z\ z']^\top \in \mathbb{R}^2$. Then, according to (1), we have the following

$$\begin{aligned} \mathcal{U}(1,1,1) &= xyz, & \mathcal{U}(2,2,1) &= x'y'z, \\ \mathcal{U}(2,1,1) &= x'yz, & \mathcal{U}(2,1,2) &= x'yz', \\ \mathcal{U}(1,2,1) &= xy'z, & \mathcal{U}(1,2,2) &= xy'z', \\ \mathcal{U}(1,1,2) &= xyz', & \mathcal{U}(2,2,2) &= x'y'z'. \end{aligned} \tag{3}$$

Recall that $(1,1,1), (2,1,1), (1,2,1)$, and $(1,1,2)$ are the observed entries. Hence, the unknown entries can be determined uniquely in terms of the 4 observed entries as

$$\begin{aligned} \mathcal{U}(2,2,1) &= x'y'z = \frac{\mathcal{U}(2,1,1)\mathcal{U}(1,2,1)}{\mathcal{U}(1,1,1)}, \\ \mathcal{U}(2,1,2) &= x'yz' = \frac{\mathcal{U}(2,1,1)\mathcal{U}(1,1,2)}{\mathcal{U}(1,1,1)}, \\ \mathcal{U}(1,2,2) &= xy'z' = \frac{\mathcal{U}(1,2,1)\mathcal{U}(1,1,2)}{\mathcal{U}(1,1,1)}, \\ \mathcal{U}(2,2,2) &= x'y'z' = \frac{\mathcal{U}(2,1,1)\mathcal{U}(1,2,1)\mathcal{U}(1,1,2)}{\mathcal{U}(1,1,1)\mathcal{U}(1,1,1)}. \end{aligned} \tag{4}$$

Therefore, based on the CP decomposition, the sampled tensor $\mathcal{U}$ is finitely (uniquely) many completable. Hence, this example illustrates that collapsing a tensor into a matrix results in loss of information and thus motivate the investigation of the tensor CP manifold.

3. Deterministic Conditions for Finite Completability

In this section, we characterize the necessary and sufficient condition on the sampling pattern for finite completability of the sampled tensor given its CP rank. In Section 3.1, we define a polynomial based on each observed entry and through studying the geometry of the manifold of the corresponding CP rank, we transform the problem of finite completability of $\mathcal{U}$ to the problem of including enough number of algebraically independent polynomials among the defined polynomials for the observed entries. In Section 3.2, a binary tensor is constructed based on the sampling pattern Ω, which allows us to study the algebraic independence of a subset of polynomials among all defined polynomials based on the samples. In Section 3.3, we characterize the connection between the maximum number of algebraically independent polynomials among all the defined polynomials and finite completability of the sampled tensor.

3.1 Geometry

Suppose that the sampled tensor $\mathcal{U}$ is chosen generically from the manifold of the tensors in $\mathbb{R}^{n_1 \times \cdots \times n_d}$ of rank r. Assume that $\mathbf{a}_i^l$ vectors are unknown for $1 \leq i \leq d$ and $1 \leq l \leq r$. For notational simplicity, define the tuples $\mathbb{A}_l = (\mathbf{a}_1^l, \mathbf{a}_2^l, \ldots, \mathbf{a}_d^l)$ for $l = 1, \ldots, r$ and $\mathbb{A} = (\mathbb{A}_1, \ldots, \mathbb{A}_r)$. Moreover, define $\mathbf{A}_i = [\mathbf{a}_i^1 | \mathbf{a}_i^2 | \ldots | \mathbf{a}_i^r] \in \mathbb{R}^{n_i \times r}$. Note that each of the sampled entries results in a polynomials in terms of the entries of $\mathbb{A}$ as in (2).

Here, we briefly mention the following two facts to highlight the fundamentals of our proposed analysis.

- **Fact** 1: As it can be seen from (2), any observed entry $\mathcal{U}(\vec{x})$ results in an equation that involves one entry of $\mathbf{a}_i^l$, $i = 1, \ldots, d$ and $l = 1, \ldots, r$. Considering the entries of $\mathbb{A}$ as variables (right-hand side of (2)), each observed entry results in a polynomial in terms of these variables. Moreover, for any observed entry $\mathcal{U}(\vec{x})$, the value of x_i specifies the location of the entry of $\mathbf{a}_i^l$ that is involved in the corresponding polynomial, $i = 1, \ldots, d$ and $l = 1, \ldots, r$.
- **Fact** 2: It can be concluded from Bernstein's theorem (Sturmfels, 2002) that in a system of n polynomials in n variables with coefficients chosen generically, the polynomials are

algebraically independent with probability one, and therefore there exist only finitely many solutions.

The following assumption will be used frequently in this paper.

Assumption 1: Each row of the d-th matricization of the sampled tensor, i.e., $\mathbf{U}_{(d)}$ includes at least r observed entries.

Observe that each row of the d-th matricization of the sampled tensor includes $n_1 \times n_2 \times \ldots \times n_{d-1}$ entries. In order to show that Assumption 1 is a very mild assumption, as an example assume that $n_1 = n_2 = \cdots = n_{d-1} = n$. Then, each row of $\mathbf{U}_{(d)}$ has n^{d-1} entries and Assumption 1 requires r sampled entries among them. As we consider the low-rank scenario, we show in Section 4 how realistic this assumption is in terms of the sampling probability.

Lemma 1 *Given $\mathbf{A}_i$'s for $i = 1, \ldots, d-1$ and Assumption 1, $\mathbf{A}_d$ can be determined uniquely.*

Proof Each row of $\mathbf{A}_d$ has r entries and also as it can be seen from (2), each observed entry in the i-th row of $\mathbf{U}_{(d)}$ results in a degree-1 polynomial in terms of the r entries of the i-th row of $\mathbf{A}_d$. Since Assumption 1 holds, for each row of $\mathbf{A}_d$ that has r variables, we have at least r degree-1 polynomials. Genericity of the coefficients of these polynomials results that each row of $\mathbf{A}_d$ can be determined uniquely. ■

As a result of Lemma 1, we can obtain $\mathbf{A}_d$ in terms of the entries of $\mathbf{A}_i$'s for $i = 1, \ldots, d-1$. As mentioned earlier, each observed entry is equivalent to a polynomial in the format of (2). Consider all such polynomials excluding those that have been used to obtain $\mathbf{A}_d$ (r samples at each row of $\mathbf{U}_{(d)}$) and denote this set of polynomials in terms of the entries of $\mathbf{A}_i$'s for $i = 1, \ldots, d-1$ by $\mathcal{P}(\Omega)$.

We are interested in defining an equivalence class such that each class includes exactly one of the decompositions among all rank-r decompositions of a particular tensor and the pattern in Lemma 3 characterizes such an equivalence class. Lemma 2 below is a re-statement of Lemma 1 in (Ashraphijuo and Wang, 2017), which characterizes such an equivalence class or equivalently geometric pattern for a matrix instead of tensor. This lemma will be used to show Lemma 3 later.

Lemma 2 *Let $\mathbf{X}$ denote a generically chosen matrix from the manifold of $n_1 \times n_2$ matrices of rank r and also $\mathbf{Q} \in \mathbb{R}^{r \times r}$ be a given full rank matrix. Then, there exists a unique decomposition $\mathbf{X} = \mathbf{YZ}$ such that $\mathbf{Y} \in \mathbb{R}^{n_1 \times r}$, $\mathbf{Z} \in \mathbb{R}^{r \times n_2}$ and $\mathbf{P} = \mathbf{Q}$, where $\mathbf{P} \in \mathbb{R}^{r \times r}$ represents a submatrix*[1] *of $\mathbf{Y}$.*

In Lemma 3, we generalize Lemma 2 and characterize the similar pattern for a multi-way tensor. Assuming that $\mathbf{P}$ represents the submatrix of $\mathbf{Y}$ consists of the first r columns and the first r rows of $\mathbf{Y}$ and also $\mathbf{Q}$ is equal to the $r \times r$ identity matrix, this pattern is called the canonical decomposition of $\mathbf{X}$. The canonical decomposition is shown for a rank-2 matrix as the following

$$\begin{array}{|c|c|c|c|}\hline 1 & -1 & 0 & -1 \\ \hline 2 & 2 & 4 & 6 \\ \hline -1 & 3 & 2 & 5 \\ \hline 1 & 2 & 3 & 5 \\ \hline \end{array} = \begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & 1 \\ \hline y_1 & y_2 \\ \hline y_3 & y_4 \\ \hline \end{array} \times \begin{array}{|c|c|c|c|}\hline x_1 & x_2 & x_3 & x_4 \\ \hline x_5 & x_6 & x_7 & x_8 \\ \hline \end{array},$$

1. Specified by a subset of rows and a subset of columns (not necessarily consecutive).

where x_i's and y_i's can be determined uniquely as

$$\begin{array}{|c|c|}\hline y_1 & y_2 \\ \hline y_3 & y_4 \\ \hline \end{array} = \begin{array}{|c|c|}\hline -2 & \frac{1}{2} \\ \hline -\frac{1}{2} & \frac{3}{4} \\ \hline \end{array} \text{ and } \begin{array}{|c|c|c|c|}\hline x_1 & x_2 & x_3 & x_4 \\ \hline x_5 & x_6 & x_7 & x_8 \\ \hline \end{array} = \begin{array}{|c|c|c|c|}\hline 1 & -1 & 0 & -1 \\ \hline 2 & 2 & 4 & 6 \\ \hline \end{array}.$$

Also, the above canonical decomposition can be written as the following

$$\begin{array}{|c|c|c|c|}\hline 1 & -1 & 0 & -1 \\ \hline 2 & 2 & 4 & 6 \\ \hline -1 & 3 & 2 & 5 \\ \hline 1 & 2 & 3 & 5 \\ \hline \end{array} = \begin{array}{|c|}\hline 1 \\ \hline 0 \\ \hline y_1 \\ \hline y_3 \\ \hline \end{array} \times \begin{array}{|c|c|c|c|}\hline x_1 & x_2 & x_3 & x_4 \\ \hline \end{array} + \begin{array}{|c|}\hline 0 \\ \hline 1 \\ \hline y_2 \\ \hline y_4 \\ \hline \end{array} \times \begin{array}{|c|c|c|c|}\hline x_5 & x_6 & x_7 & x_8 \\ \hline \end{array}.$$

Generalization of the canonical decomposition for multi-way tensor is as the following

$$\mathbf{a}_1^1 = \begin{array}{|c|}\hline 1 \\ \hline 0 \\ \hline \vdots \\ \hline 0 \\ \hline \mathbf{a}_1^1(r+1) \\ \hline \vdots \\ \hline \mathbf{a}_1^1(n_1) \\ \hline \end{array}, \ldots, \mathbf{a}_1^r = \begin{array}{|c|}\hline 0 \\ \hline 0 \\ \hline \vdots \\ \hline 1 \\ \hline \mathbf{a}_1^r(r+1) \\ \hline \vdots \\ \hline \mathbf{a}_1^r(n_1) \\ \hline \end{array},$$

and for $i \in \{2, \ldots, d-1\}$

$$\mathbf{a}_i^1 = \begin{array}{|c|}\hline 1 \\ \hline \mathbf{a}_i^1(2) \\ \hline \vdots \\ \hline \mathbf{a}_i^1(n_i) \\ \hline \end{array}, \ldots, \mathbf{a}_i^r = \begin{array}{|c|}\hline 1 \\ \hline \mathbf{a}_i^r(2) \\ \hline \vdots \\ \hline \mathbf{a}_i^r(n_i) \\ \hline \end{array}.$$

Lemma 3 *Let $j \in \{1, \ldots, d-1\}$ be a fixed number and define $J = \{1, \ldots, d-1\} \backslash \{j\}$. Assume that the full rank matrix $\mathbf{Q}_j \in \mathbb{R}^{r \times r}$ and matrices $\mathbf{Q}_i \in \mathbb{R}^{1 \times r}$ with nonzero entries for $i \in J$ are given. Also, let $\mathbf{P}_i$ denote an arbitrary submatrix of $\mathbf{A}_i$, $i = 1, 2, \ldots, d-1$, where $\mathbf{P}_j \in \mathbb{R}^{r \times r}$ and $\mathbf{P}_i \in \mathbb{R}^{1 \times r}$ for $i \in J$. Then, with probability one, there exists exactly one rank-r decomposition of $\mathcal{U}$ such that $\mathbf{P}_i = \mathbf{Q}_i$, $i = 1, \ldots, d-1$.*

Proof First we claim that there exists at most one rank-r decomposition of $\mathcal{U}$ such that $\mathbf{P}_i = \mathbf{Q}_i$, $i = 1, \ldots, d-1$. We assume that $\mathbf{P}_i = \mathbf{Q}_i$, $i = 1, \ldots, d-1$ and also $\mathcal{U}$ is given. Then, it suffices to show that the rest of the entries of $\mathbb{A}$ can be determined in at most a unique way (no more than one solution) in terms of the given parameters such that (1) holds. Note that if a variable can be determined uniquely through two different ways (two sets of samples or equations), in general either it can be determined uniquely if both ways result in the same value or it does not have any solution otherwise. Let y_i denote the row number of submatrix $\mathbf{P}_i \in \mathbb{R}^{1 \times r}$ for $i \in J$ and $Y_j = \{y_j^1, \ldots, y_j^r\}$ denote the row numbers of submatrix $\mathbf{P}_j \in \mathbb{R}^{r \times r}$.

As the first step of proving our claim, we show that $\mathbf{A}_d$ can be determined uniquely. Consider the subtensor $\mathcal{U}' = \mathcal{U}(y_1, \ldots, y_{j-1}, Y_j, y_{j+1}, \ldots, y_{d-1}, :) \in \mathbb{R}^{\overbrace{1 \times \ldots \times 1}^{j-1} \times r \times \overbrace{1 \times \ldots \times 1}^{d-j-1} \times n_d}$ which

includes rn_d entries. Having CP decomposition (2), each entry of $\mathcal{U}'$ results in one degree-1 polynomial in terms of the entries of $\mathbf{A}_d$ with coefficients in terms of the entries of $\mathbf{Q}_i$'s. Let the matrix $\mathbf{U}' \in \mathbb{R}^{r\times n_d}$ represent the rn_d entries of $\mathcal{U}'$. Moreover, define $\mathbf{C} = [\mathbf{c}_1|\dots|\mathbf{c}_r] \in \mathbb{R}^{r\times r}$ where $\mathbf{c}_l = (\Pi_{i\in J}\mathbf{P}_i(1,l))\,\mathbf{q}_j^l \in \mathbb{R}^{r\times 1}$ for $l = 1,\dots,r$ and $\mathbf{q}_j^l \in \mathbb{R}^{r\times 1}$ is the l-th column of $\mathbf{Q}_j$.

Observe that CP decomposition (2) for the entries of $\mathcal{U}'$ can be written as $\mathbf{U}' = \mathbf{C}\mathbf{A}_d^\top$. Recall that $\mathbf{Q}_j$ is full rank, and therefore $\mathbf{q}_j^l$'s are linearly independent, $l = 1,\dots,r$. Also, a system of equations with at least m linearly independent degree-1 polynomials in m variables does not have more than one solution. Hence, $\mathbf{c}_l$'s are also linearly independent for $l = 1,\dots,r$, and therefore $\mathbf{C}$ is full rank. As a result, $\mathbf{A}_d$ can be determined uniquely. In the second step, similar to the first step, we can show that the rest of $\mathbf{A}_i$'s have at most one solution having one entry of $\mathbf{A}_d$ which has been already obtained.

Finally, we also claim that there exists at least one rank-r decomposition of $\mathcal{U}$ such that $\mathbf{P}_i = \mathbf{Q}_i$, $i = 1,\dots,d-1$. We show this by induction on d. For $d = 2$, this is a result of Lemma 2. Induction hypothesis states that the claim holds for $d = k-1$ and we need to show that it also holds for $d = k$. Since by merging dimension $k-1$ and k for each of the rank-1 tensors of the corresponding CP decomposition and using induction hypothesis this step reduces to showing a rank-1 matrix can be decomposed to two vectors such that one component of one of them is given which is again a special case of Lemma 2 for rank-1 scenario. ■

Assume that $\mathcal{S}$ denotes the set of all possible $\mathbf{A}_i$'s for $i = 1,\dots,d-1$ given $\mathbf{A}_d$ without any polynomial constraint. Lemma 3 results in a pattern that characterizes exactly one rank-r decomposition among all rank-r decompositions, and therefore the dimension of $\mathcal{S}$ is equal to the number of unknowns, i.e., number of entries of $\mathbf{A}_i$'s for $i = 1,\dots,d-1$ excluding those that are involved in the pattern $\mathbf{P}_i$'s in Lemma 3 which is $r(\sum_{i=1}^{d-1} n_i) - r^2 - r(d-2)$.

Lemma 4 *For almost every $\mathcal{U}$, the sampled tensor is finitely completable if and only if the maximum number of algebraically independent polynomials in $\mathcal{P}(\Omega)$ is equal to $r(\sum_{i=1}^{d-1} n_i) - r^2 - r(d-2)$.*

Proof The proof is omitted due to the similarity to the proof of Lemma 2 in (Ashraphijuo et al., 2016a) with the only difference that here the dimension is $r(\sum_{i=1}^{d-1} n_i) - r^2 - r(d-2)$ instead of $\left(\Pi_{i=1}^{j} n_i\right)\left(\Pi_{i=j+1}^{d} r_i\right) - \left(\sum_{i=j+1}^{d} r_i^2\right)$ which is the dimension of the core in Tucker decomposition. ■

3.2 Constraint Tensor

In this section, we provide a procedure to construct a binary tensor $\breve{\Omega}$ based on Ω such that $\mathcal{P}(\breve{\Omega}) = \mathcal{P}(\Omega)$ and each polynomial can be represented by one d-way subtensor of $\breve{\Omega}$ which belongs to $\mathbb{R}^{n_1\times n_2\times\cdots\times n_{d-1}\times 1}$. Using $\breve{\Omega}$, we are able to recognize the observed entries that have been used to obtain the $\mathbf{A}_d$ in terms of the entries of $\mathbf{A}_1,\dots,\mathbf{A}_{d-1}$, and we can study the algebraic independence of the polynomials in $\mathcal{P}(\Omega)$ which is directly related to finite completability through Lemma 4.

For each subtensor $\mathcal{Y}$ of the sampled tensor $\mathcal{U}$, let $N_\Omega(\mathcal{Y})$ denote the number of sampled entries in $\mathcal{Y}$. Specifically, consider any subtensor $\mathcal{Y} \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_{d-1}\times 1}$ of the tensor $\mathcal{U}$. Then, since r of the polynomials have been used to obtain $\mathbf{A}_d$, $\mathcal{Y}$ contributes $N_\Omega(\mathcal{Y}) - r$ polynomial equations in terms of the entries of $\mathbf{A}_1,\dots,\mathbf{A}_{d-1}$ among all $N_\Omega(\mathcal{U}) - rn_d$ polynomials in $\mathcal{P}(\Omega)$.

The sampled tensor $\mathcal{U}$ includes n_d subtensors that belong to $\mathbb{R}^{n_1\times n_2\times\cdots\times n_{d-1}\times 1}$ and let $\mathcal{Y}_i$ for $1 \le i \le n_d$ denote these n_d subtensors. Define a binary valued tensor $\breve{\mathcal{Y}}_i \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_{d-1}\times k_i}$, where $k_i = N_\Omega(\mathcal{Y}_i) - r$ and its entries are described as the following. We can look at $\breve{\mathcal{Y}}_i$ as k_i tensors each belongs to $\mathbb{R}^{n_1\times n_2\times\cdots\times n_{d-1}\times 1}$. For each of the mentioned k_i tensors in $\breve{\mathcal{Y}}_i$ we set the entries corresponding to the r observed entries that are used to obtain $\mathbf{A}_d$ equal to 1. For each of the other k_i observed entries that have not been used to obtain $\mathbf{A}_d$, we pick one of the k_i tensors of $\breve{\mathcal{Y}}_i$ and set its corresponding entry (the same location as that specific observed entry) equal to 1 and set the rest of the entries equal to 0. In the case that $k_i = 0$ we simply ignore $\breve{\mathcal{Y}}_i$, i.e., $\breve{\mathcal{Y}}_i = \emptyset$

By putting together all n_d tensors in dimension d, we construct a binary valued tensor $\breve{\Omega} \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_{d-1}\times K}$, where $K = \sum_{i=1}^{n_d} k_i = N_\Omega(\mathcal{U}) - rn_d$ and call it the **constraint tensor**. Observe that each subtensor of $\breve{\Omega}$ which belongs to $\mathbb{R}^{n_1\times n_2\times\cdots\times n_{d-1}\times 1}$ includes exactly $r+1$ nonzero entries. In the following we show this procedure for an example.

Example 1 Consider an example in which $d = 3$ and $r = 2$ and $\mathcal{U} \in \mathbb{R}^{3\times 3\times 3}$. Assume that $\Omega(x,y,z) = 1$ if $(x,y,z) \in \mathcal{S}$ and $\Omega(x,y,z) = 0$ otherwise, where

$$\mathcal{S} = \{(1,1,1),(1,2,1),(2,3,1),(3,3,1),(1,1,2),(2,1,2),(3,2,2),(1,3,3),(3,2,3)\},$$

represents the set of observed entries. Hence, observed entries $(1,1,1),(1,2,1),(2,3,1),(3,3,1)$ belong to $\mathcal{Y}_1$, observed entries $(1,1,2),(2,1,2),(3,2,2)$ belong to $\mathcal{Y}_2$, and observed entries $(1,3,3),(3,2,3)$ belong to $\mathcal{Y}_3$. As a result, $k_1 = 4-2 = 2$, $k_2 = 3-2 = 1$, and $k_3 = 2-2 = 0$. Hence, $\breve{\mathcal{Y}}_1 \in \mathbb{R}^{3\times 3\times 2}$, $\breve{\mathcal{Y}}_2 \in \mathbb{R}^{3\times 3\times 1}$, and $\breve{\mathcal{Y}}_3 = \emptyset$, and therefore the constraint tensor $\breve{\Omega}$ belongs to $\mathbb{R}^{3\times 3\times 3}$.

Also, assume that the entries that we use to obtain $\mathbf{A}_3$ in terms of the entries of $\mathbf{A}_1$ and $\mathbf{A}_2$ are $(2,3,1)$, $(3,3,1)$, $(1,1,2)$, $(2,1,2)$, $(1,3,3)$ and $(3,2,3)$. Note that $\breve{\mathcal{Y}}_1(2,3,1) = \breve{\mathcal{Y}}_1(2,3,2) = \breve{\mathcal{Y}}_1(3,3,1) = \breve{\mathcal{Y}}_1(3,3,2) = 1$ (correspond to entries of $\mathcal{Y}_1$ that have been used to obtain $\mathbf{A}_3$), and also for the two other observed entries we have $\breve{\mathcal{Y}}_1(1,1,1) = 1$ (correspond to $\mathcal{U}(1,1,1)$) and $\breve{\mathcal{Y}}_1(1,2,2) = 1$ (correspond to $\mathcal{U}(1,2,1)$) and the rest of the entries of $\breve{\mathcal{Y}}_1$ are equal to zero. Similarly, $\breve{\mathcal{Y}}_2(1,1,1) = \breve{\mathcal{Y}}_2(2,1,1) = \breve{\mathcal{Y}}_2(3,2,1) = 1$ and the rest of the entries of $\breve{\mathcal{Y}}_2$ are equal to zero.

Then, $\breve{\Omega}(x,y,z) = 1$ if $(x,y,z) \in \breve{\mathcal{S}}$ and $\breve{\Omega}(x,y,z) = 0$ otherwise, where

$$\breve{\mathcal{S}} = \{(1,1,1),(1,2,2),(2,3,1),(2,3,2),(3,3,1),(3,3,2),(1,1,3),(2,1,3),(3,2,3)\}.$$

Note that each subtensor of $\breve{\Omega}$ that belongs to $\mathbb{R}^{n_1\times\dots\times n_{d-1}\times 1}$ represents one of the polynomials in $\mathcal{P}(\Omega)$ besides showing the polynomials that have been used to obtain $\mathbf{A}_d$. More specifically, consider a subtensor of $\breve{\Omega}$ that belongs to $\mathbb{R}^{n_1\times\dots\times n_{d-1}\times 1}$ with $r+1$ nonzero entries. Observe that exactly r of them correspond to the observed entries that have been used to obtain $\mathbf{A}_d$. Hence, this subtensor represents a polynomial after replacing entries of $\mathbf{A}_d$ by the expressions in terms of entries of $\mathbf{A}_1,\dots,\mathbf{A}_{d-1}$, i.e., a polynomial in $\mathcal{P}(\Omega)$.

3.3 Algebraic Independence

In this section, we obtain the maximum number of algebraically independent polynomials in $\mathcal{P}(\breve{\Omega})$ in terms of the simple geometrical structure of nonzero entries of Ω, i.e., the locations of the sampled entries. On the other hand, Lemma 4 provides the required number of algebraically independent polynomials in $\mathcal{P}(\Omega)$ for finite completability. Hence, at the end of this section, we obtain the necessary and sufficient deterministic conditions on the sampling pattern for finite completability.

According to Lemma 3, as we consider one particular equivalence class some of the entries of $\mathbf{A}_i$'s are known, i.e., $\mathbf{P}_1, \ldots, \mathbf{P}_{d-1}$ in the statement of the lemma. Therefore, in order to find the number of variables (unknown entries of $\mathbf{A}_i$'s) in a set of polynomials, we should subtract the number of known entries in the corresponding pattern from the total number of involved entries. Also, recall that the sampled tensor is chosen generically from the corresponding manifold, and therefore according to Fact 2, the independency of the polynomials can be studied through studying the number of variables involved in each subset of them.

Definition 5 *Let $\breve{\Omega}' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t}$ be a subtensor of the constraint tensor $\breve{\Omega}$. Let $m_i(\breve{\Omega}')$ denote the number of nonzero rows of $\breve{\mathbf{\Omega}}'_{(i)}$. Also, let $\mathcal{P}(\breve{\Omega}')$ denote the set of polynomials that correspond to nonzero entries of $\breve{\Omega}'$.*

The following lemma gives an upper bound on the maximum number of algebraically independent polynomials in the set $\mathcal{P}(\breve{\Omega}')$ for an arbitrary subtensor $\breve{\Omega}' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t}$ of the constraint tensor. Note that $\mathcal{P}(\breve{\Omega}')$ includes exactly t polynomials as each subtensor belonging to $\mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times 1}$ represents one polynomial.

Lemma 6 *Suppose that Assumption 1 holds. Consider an arbitrary subtensor $\breve{\Omega}' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t}$ of the constraint tensor $\breve{\Omega}$. The maximum number of algebraically independent polynomials in $\mathcal{P}(\breve{\Omega}')$ is at most*

$$r\left(\left(\sum_{i=1}^{d-1} m_i(\breve{\Omega}')\right) - \min\left\{\max\left\{m_1(\breve{\Omega}'), \ldots, m_{d-1}(\breve{\Omega}')\right\}, r\right\} - (d-2)\right). \tag{5}$$

Proof As a consequence of Fact 2, the maximum number of algebraically independent polynomials in a subset of polynomials of $\mathcal{P}(\breve{\Omega}')$ is at most equal to the total number of variables that are involved in the corresponding polynomials. Note that by observing the structure of (2) and Fact 1, the number of entries of $\mathbf{A}_i$ that are involved in the polynomials $\mathcal{P}(\breve{\Omega}')$ is equal to $rm_i(\breve{\Omega}')$, $i = 1, \ldots, d-1$. Therefore, the total number of entries of $\mathbf{A}_1, \ldots, \mathbf{A}_{d-1}$ that are involved in the polynomials $\mathcal{P}(\breve{\Omega}')$ is equal to $r\left(\sum_{i=1}^{d-1} m_i(\breve{\Omega}')\right)$. However, some of the entries of $\mathbf{A}_1, \ldots, \mathbf{A}_{d-1}$ are known and depending on the equivalence class we should subtract them from the total number of involved entries.

For a fixed number j in Lemma 3, it is easily verified that the total number of variables (unknown entries) of $\mathbf{A}_1, \ldots, \mathbf{A}_{d-1}$ that are involved in the polynomials $\mathcal{P}(\breve{\Omega}')$ is equal to $r\sum_{i\in J}\left(m_i(\breve{\Omega}') - 1\right)^+ + r\left(m_j(\breve{\Omega}') - r\right)^+$, with $J = \{1, \ldots, d-1\}\backslash\{j\}$. Note that $\left(m_i(\breve{\Omega}') - 1\right)^+ = m_i(\breve{\Omega}') - 1$, $\left(m_j(\breve{\Omega}') - r\right)^+ = m_j(\breve{\Omega}') - \min\left\{m_j(\breve{\Omega}'), r\right\}$. However, j is not a fixed number in general. Therefore, the maximum number of known entries of $\mathbf{A}_1, \ldots, \mathbf{A}_{d-1}$ that are involved in the polynomials $\mathcal{P}(\breve{\Omega}')$ is equal to $r\left(\min\left\{\max\left\{m_1(\breve{\Omega}'), \ldots, m_{d-1}(\breve{\Omega}')\right\}, r\right\} - (d-2)\right)$, which results that the number of variables of $\mathbf{A}_1, \ldots, \mathbf{A}_{d-1}$ that are involved in the polynomials $\mathcal{P}(\breve{\Omega}')$ is equal to (5). ■

The set of polynomials corresponding to $\breve{\Omega}'$, i.e., $\mathcal{P}(\breve{\Omega}')$ is called minimally algebraically dependent if the polynomials in $\mathcal{P}(\breve{\Omega}')$ are algebraically dependent but polynomials in every of its proper subsets are algebraically independent. The following lemma provides an important property about a set of minimally algebraically dependent $\mathcal{P}(\breve{\Omega}')$. This lemma will be used later to derive the maximum number of algebraically independent polynomials in $\mathcal{P}(\breve{\Omega}')$.

Lemma 7 *Suppose that Assumption 1 holds. Consider an arbitrary subtensor $\breve{\Omega}' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t}$ of the constraint tensor $\breve{\Omega}$. Assume that polynomials in $\mathcal{P}(\breve{\Omega}')$ are minimally algebraically dependent. Then, the number of variables (unknown entries) of $\mathbf{A}_1, \ldots, \mathbf{A}_{d-1}$ that are involved in $\mathcal{P}(\breve{\Omega}')$ is equal to $t-1$.*

Proof Let the set of polynomials $\{p_1, , p_2 \ldots, p_t\}$ represent $\mathcal{P}(\breve{\Omega}')$. Then, $\mathcal{P}(\breve{\Omega}')\backslash p_i$ (all polynomials in $\mathcal{P}(\breve{\Omega}')$ excluding p_i) is a set of algebraically independent polynomials, $i = 1, \ldots, t$. The number of involved variables in $\mathcal{P}(\breve{\Omega}')\backslash p_i$ is at least equal to the number of polynomials, i.e., $t-1$. Therefore, the number of involved variables in $\mathcal{P}(\breve{\Omega}')$ is at least $t-1$.

By contradiction, assume that the number of variables that are involved in the set of polynomials $\mathcal{P}(\breve{\Omega}')$ is at least t. Then, we claim that for any subset of the polynomials of $\mathcal{P}(\breve{\Omega}')$ the number of involved variables is at least equal to the number of polynomials in that subset. If the subset is equal to $\mathcal{P}(\breve{\Omega}')$, the claim is just the assumption of contradiction. If the subset is a proper subset of $\mathcal{P}(\breve{\Omega}')$, according to the assumption in the statement of Lemma, the polynomials are algebraically independent, and therefore, the claim holds. Hence, the polynomials in $\mathcal{P}(\breve{\Omega}')$ are algebraically independent which contradicts the assumption. ■

Given an arbitrary subtensor $\breve{\Omega}' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t}$ of the constraint tensor $\breve{\Omega}$, we are interested in obtaining the maximum number of algebraically independent polynomials in $\mathcal{P}(\breve{\Omega}')$ based on the structure of nonzero entries of $\breve{\Omega}'$. The next lemma can be used to characterize this number in terms of a simple geometric structure of nonzero entries of $\breve{\Omega}'$.

Lemma 8 *Suppose that Assumption 1 holds. Consider an arbitrary subtensor $\breve{\Omega}' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t}$ of the constraint tensor $\breve{\Omega}$. The polynomials in $\mathcal{P}(\breve{\Omega}')$ are algebraically independent if and only if for any $t' \in \{1, \ldots, t\}$ and any subtensor $\breve{\Omega}'' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t'}$ of $\breve{\Omega}'$ we have*

$$r\left(\left(\sum_{i=1}^{d-1} m_i(\breve{\Omega}'')\right) - \min\left\{\max\left\{m_1(\breve{\Omega}''), \ldots, m_{d-1}(\breve{\Omega}'')\right\}, r\right\} - (d-2)\right) \geq t'. \tag{6}$$

Proof First, assume that all polynomials in $\mathcal{P}(\breve{\Omega}')$ are algebraically independent. Also, by contradiction assume that there exists a subtensor $\breve{\Omega}'' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t'}$ of $\breve{\Omega}'$ that (6) does not hold for. Note that $\mathcal{P}(\breve{\Omega}'')$ includes t' polynomials. On the other hand, according to Lemma 6, the maximum number of algebraically independent polynomials in $\mathcal{P}(\breve{\Omega}'')$ is no greater than the LHS of (6), and therefore the polynomials in $\mathcal{P}(\breve{\Omega}'')$ are not algebraically independent. Hence, the polynomials in $\mathcal{P}(\breve{\Omega}')$ are not algebraically independent as well.

In order to prove the other side of the statement, assume that the polynomials in $\mathcal{P}(\breve{\Omega}')$ are algebraically dependent. Hence, there exists a subset of the polynomials that are minimally algebraically dependent and let us denote it by $\mathcal{P}(\breve{\Omega}'')$, where $\breve{\Omega}'' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t'}$ is a subtensor of $\breve{\Omega}'$. As stated in Lemma 7, the number of involved variables in polynomials in $\mathcal{P}(\breve{\Omega}'')$ is equal to $t'-1$. On the other hand, in the proof of Lemma 6, we showed that the number involved variables is at least

equal the LHS of (6). Therefore, the LHS of (6) is less than or equal to $t'-1$ or equivalently

$$r\left(\left(\sum_{i=1}^{d-1} m_i(\breve{\Omega}'')\right) - \min\left\{\max\left\{m_1(\breve{\Omega}''),\ldots,m_{d-1}(\breve{\Omega}'')\right\},r\right\} - (d-2)\right) < t'. \tag{7}$$

■

Finally, the following theorem characterizes the necessary and sufficient condition on $\breve{\Omega}$ for finite completability of the sampled tensor $\mathcal{U}$.

Theorem 9 *Suppose that Assumption* 1 *holds. For almost every $\mathcal{U}$, the sampled tensor $\mathcal{U}$ is finitely completable if and only if $\breve{\Omega}$ contains a subtensor $\breve{\Omega}' \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_{d-1}\times t}$ such that (i) $t = r(\sum_{i=1}^{d-1} n_i) - r^2 - r(d-2)$ and (ii) for any $t' \in \{1,\ldots,t\}$ and any subtensor $\breve{\Omega}'' \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_{d-1}\times t'}$ of $\breve{\Omega}'$,* (6) *holds.*

Proof Lemma 4 states that for almost every $\mathcal{U}$, there exist finitely many completions of the sampled tensor if and only if $\mathcal{P}(\breve{\Omega})$ includes $r(\sum_{i=1}^{d-1} n_i) - r^2 - r(d-2)$ algebraically independent polynomials. Moreover, according to Lemma 8, polynomials corresponding to a subtensor $\breve{\Omega}' \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_{d-1}\times t}$ of the constraint tensor are algebraically independent if and only if condition (ii) of the statement of the Theorem holds. Therefore, for almost every $\mathcal{U}$, conditions (i) and (ii) hold if and only if the sampled tensor $\mathcal{U}$ is finitely completable. ■

Note that the condition given in (6) is combinatorial in nature and hard to verify in practice. In the next section, we provide a lower bound on the number of samples so that both Assumption 1 and the combinatorial condition given in (6) hold true with high probability (not deterministically, i.e., with probability one anymore).

4. Probabilistic Conditions for Finite Completability

Assume that the entries of the tensor are sampled independently with probability p. In this section, we are interested in obtaining a condition in terms of the number of samples, i.e., the sampling probability, to ensure the finite completability of the sampled tensor with high probability. In Section 4.1, we apply the existing results on the Grassmannian manifold in (Pimentel-Alarcón et al., 2016d) on any of the unfoldings of the sampled tensor to derive the mentioned probabilistic condition. In Section 4.2, we obtain the conditions on the number of samples to ensure that conditions (i) and (ii) in the statement of Theorem 9 hold with high probability or in other words, to ensure the finite completability with high probability. For the notational simplicity in this section, we assume that $n_1 = n_2 = \cdots = n_d$, i.e., $\mathcal{U} \in \mathbb{R}^{n\times n\times\ldots\times n}$.

4.1 Unfolding Approach

In this section, we are interested in applying the existing analysis based on the Grassmannian manifold to obtain probabilistic conditions on the sampling pattern for finite completability with high probability. The following theorem restates Theorem 3 in (Pimentel-Alarcón et al., 2016d).

Theorem 10 *Consider an $n\times N$ matrix with the given rank k and let $0<\epsilon<1$ be given. Suppose $k \le \frac{n}{6}$ and that each* **column** *of the sampled matrix is observed in at least l entries, distributed uniformly at random and independently across entries, where*

$$l > \max\left\{12\ \log\left(\frac{n}{\epsilon}\right) + 12, 2k\right\}. \tag{8}$$

Also, assume that $k(n-k) \le N$. Then, with probability at least $1-\epsilon$, the sampled matrix will be finitely completable.

Observe that in the case of $1<k<n-1$, the assumption $k(n-k)\le N$ results that $n<N$ which is very important to check when we apply this theorem. In order to use Theorem 10, we need the following lemma to obtain an upper bound on the rank of unfoldings of $\mathcal{U}$.

Lemma 11 *Consider an arbitrary nonempty $I \subset \{1,\ldots,d\}$ and recall that r denotes the CP rank of $\mathcal{U}$. Then, rank $\left(\widetilde{\mathbf{U}}_{(I)}\right) \le r$.*

Proof In order to show rank $\left(\widetilde{\mathbf{U}}_{(I)}\right) \le r$, we show the existence of a CP decomposition of $\widetilde{\mathbf{U}}_{(I)}$ with r components, i.e., we show that there exist $\mathbf{a}_i^l \in \mathbb{R}^{m_i}$ for $1\le i\le 2$, $1\le l\le r$, $m_1 \triangleq N_I$ and $m_2 \triangleq N_{\bar{I}}$ such that

$$\widetilde{\mathbf{U}}_{(I)} = \sum_{l=1}^{r} \mathbf{a}_1^l \otimes \mathbf{a}_2^l. \tag{9}$$

In order to do so, recall that since the CP rank of $\mathcal{U}$ is r, there exist $\mathbf{b}_i^l \in \mathbb{R}^{n_i}$ for $1\le i\le d$ and $1\le l\le r$ such that

$$\mathcal{U} = \sum_{l=1}^{r} \mathbf{b}_1^l \otimes \mathbf{b}_2^l \otimes \ldots \otimes \mathbf{b}_d^l. \tag{10}$$

Define $\mathcal{A}_1^l = \mathbf{b}_{i_1}^l \otimes \ldots \otimes \mathbf{b}_{i_{|I|}}^l$ and $\mathcal{A}_2^l = \mathbf{b}_{i_{|I|+1}}^l \otimes \ldots \otimes \mathbf{b}_{i_d}^l$ for $1\le l\le l$, where $I = \{i_1,\ldots,i_{|I|}\}$, $\bar{I} = \{i_{|I|+1},\ldots,i_d\}$. Let $\mathbf{a}_1^l$ and $\mathbf{a}_2^l$ denote the vectorizations of $\mathcal{A}_1^l$ and $\mathcal{A}_2^l$ with the same bijective mappings $\widetilde{M}_I : (x_{i_1},\ldots,x_{i_{|I|}}) \to \{1,2,\ldots,N_I\}$ and $\widetilde{M}_{\bar{I}} : (x_{i_{|I|+1}},\ldots,x_{i_d}) \to \{1,2,\ldots,N_{\bar{I}}\}$ of the unfolding $\widetilde{\mathbf{U}}_{(I)}$. Hence, there exist $\mathbf{a}_i^l \in \mathbb{R}^{m_i}$ for $1\le i\le 2$, $1\le l\le r$ such that (9) holds. ■

Remark 12 *Assume that $k \le k' \le \frac{n}{6}$, $l > \max\left\{12\ \log\left(\frac{n}{\epsilon}\right)+12, 2k'\right\}$ and $k'(n-k') \le N$. Then, we have $l > \max\left\{12\ \log\left(\frac{n}{\epsilon}\right)+12, 2k\right\}$ since $k\le k'$. Moreover, we have $k(n-k)\le N$ since $k+k'<n$ which results $k(n-k)<k'(n-k')$.*

Lemma 13 *Let $I = \{i_1,\ldots,i_{|I|}\}$ be an arbitrary nonempty and proper subset of $\{1,\ldots,d\}$. Assume that $|I| < \frac{d}{2}$ and $r \le \frac{n}{6}$, where r is the CP rank of the sampled tensor $\mathcal{U}$. Moreover, assume that each* **column** *of $\widetilde{\mathbf{U}}_{(I)}$ is observed in at least l entries, distributed uniformly at random and independently across entries, where*

$$l > \max\left\{12\ \log\left(\frac{N_I r}{\epsilon}\right) + 12, 2r\right\}. \tag{11}$$

Then, with probability at least $1-\epsilon$, the sampled tensor will be finitely completable.

Proof According to Lemma 11, $r_I \triangleq \text{rank}\left(\widetilde{\mathbf{U}}_{(I)}\right) \leq r$. Note that $N_I \leq n^{\frac{d-1}{2}} = \frac{n^{\frac{d+1}{2}}}{n} \leq \frac{N_{\bar{I}}}{n}$ which results that $r(N_I - r) \leq N_{\bar{I}}$. Furthermore, according to Remark 12, we have $r_I(N_I - r_I) \leq N_{\bar{I}}$ and also

$$l > \max\left\{12\ \log\left(\frac{N_I r}{\epsilon}\right) + 12, 2r_I\right\}. \tag{12}$$

Therefore, according to Theorem 10, $\widetilde{\mathbf{U}}_{(I)}$ is finitely completable for an arbitrary value of r_I that belongs to $\{1, \ldots, r\}$ with probability at least $1 - \frac{\epsilon}{r}$. Hence, with probability at least $1 - \epsilon$, for all possible values of r_I, $\widetilde{\mathbf{U}}_{(I)}$ is finitely completable, i.e., $\widetilde{\mathbf{U}}_{(I)}$ is finitely completable. In order to complete to proof, it suffices to observe that finite completability of any of the unfoldings of $\mathcal{U}$ results the finite completability of $\mathcal{U}$. ■

Remark 14 *In the case of $|I| > \frac{d}{2}$ in Lemma 13, we can simply consider the transpose of $\widetilde{\mathbf{U}}_{(I)}$ to have the similar results.*

Remark 15 *Lemma 13 requires*

$$n^{d-|I|} \max\left\{12\ \log\left(\frac{n^{|I|}r}{\epsilon}\right) + 12, 2r\right\} \tag{13}$$

samples in total to ensure the finite completability of $\mathcal{U}$ with probability at least $1 - \epsilon$. Hence, the best bound on the total number of samples to ensure the finite completability with probability at least $1 - \epsilon$ will be obtained when $|I| = \lfloor \frac{d-1}{2} \rfloor$, which is

$$n^{\lceil \frac{d+1}{2} \rceil} \max\left\{12\ \log\left(\frac{n^{\lfloor \frac{d-1}{2} \rfloor}r}{\epsilon}\right) + 12, 2r\right\}. \tag{14}$$

4.2 CP Approach

In this section, we present an approach based on the tensor CP decomposition instead of unfolding. Conditions (i) and (ii) in Theorem 9 ensure finite completability with probability one. Here, using combinatorial methods, we derive a lower bound on the number of sampled entries, i.e., the sampling probability, which ensures conditions (i) and (ii) in Theorem 9 hold with high probability. We first provide a few lemmas from our previous works. Lemma 16 below is Lemma 5 in (Ashraphijuo et al., 2017c), which will be used later.

Lemma 16 *Assume that $r' \leq \frac{n}{6}$ and also each column of $\mathbf{\Omega}_{(1)}$ (first matricization of Ω) includes at least l nonzero entries, where*

$$l > \max\left\{9\ \log\left(\frac{n}{\epsilon}\right) + 3\ \log\left(\frac{k}{\epsilon}\right) + 6, 2r'\right\}. \tag{15}$$

Let $\mathbf{\Omega}'_{(1)}$ be an arbitrary set of $n - r'$ columns of $\mathbf{\Omega}_{(1)}$. Then, with probability at least $1 - \frac{\epsilon}{k}$, every subset $\mathbf{\Omega}''_{(1)}$ of columns of $\mathbf{\Omega}'_{(1)}$ satisfies

$$m_1(\Omega'') - r' \geq t, \tag{16}$$

where t is the number of columns of $\mathbf{\Omega}''_{(1)}$ and $m_1(\Omega'')$ is the number of nonzero rows of $\mathbf{\Omega}''_{(1)}$.

The proof of the above lemma is based on a simple combinatorial idea to upper bound the number of possible patterns that do not satisfy the mentioned property.

Lemma 17 *Let $j \in \{1, 2, \ldots, d-1\}$ be a fixed number and $I = \{1, 2, \ldots, j\}$. Consider an arbitrary set $\widetilde{\mathbf{\Omega}}'_{(I)}$ of $n - r'$ columns of $\widetilde{\mathbf{\Omega}}_{(I)}$, where $r' \leq r \leq \frac{n}{6}$. Assume that $n > 200$, and also each column of $\widetilde{\mathbf{\Omega}}_{(I)}$ includes at least l nonzero entries, where*

$$l > \max\left\{27\ \log\left(\frac{n}{\epsilon}\right) + 9\ \log\left(\frac{2k}{\epsilon}\right) + 18, 6r'\right\}, \tag{17}$$

where $k \leq r$. Then, with probability at least $1 - \frac{\epsilon}{2k}$, each column of $\widetilde{\mathbf{\Omega}}'_{(I)}$ includes more than $l_0 \triangleq \max\left\{9\ \log\left(\frac{n}{\epsilon}\right) + 3\ \log\left(\frac{2k}{\epsilon}\right) + 6, 2r'\right\}$ observed entries of Ω with different values of the i-th coordinate, i.e., the i-th matricization of the tensor Ω' that corresponds to $\widetilde{\mathbf{\Omega}}'_{(I)}$ includes more than l_0 nonzero rows, $1 \leq i \leq j$.

Proof The proof is omitted due to the similarity to the proof for Lemma 9 in (Ashraphijuo and Wang, 2017). ■

The following lemma is Lemma 8 in (Ashraphijuo et al., 2016a), which states that if the property in Lemma 16 holds for the sampling pattern Ω, it will be satisfied for $\breve{\Omega}$ as well.

Lemma 18 *Let r' be a given nonnegative integer and $1 \leq i \leq j \leq d-1$ and $I = \{1, 2, \ldots, j\}$. Assume that there exists an $n^j \times (n - r')$ matrix $\widetilde{\mathbf{\Omega}}'_{(I)}$ composed of $n - r'$ columns of $\widetilde{\mathbf{\Omega}}_{(I)}$ such that each column of $\widetilde{\mathbf{\Omega}}'_{(I)}$ includes at least $r' + 1$ nonzero entries and satisfies the following property:*

- *Denote an $n^j \times t$ matrix (for any $1 \leq t \leq n - r'$) composed of any t columns of $\widetilde{\mathbf{\Omega}}'_{(I)}$ by $\widetilde{\mathbf{\Omega}}''_{(I)}$. Then*

$$m_i(\Omega'') - r' \geq t, \tag{18}$$

where $\widetilde{\mathbf{\Omega}}''_{(I)}$ is the unfolding of Ω'' corresponding to the set I.

Then, there exists an $n^j \times (n - r')$ matrix $\widetilde{\breve{\mathbf{\Omega}}}'_{(I)}$ such that: each column has exactly $r' + 1$ entries equal to one, and if $\widetilde{\breve{\mathbf{\Omega}}}'_{(I)}(x, y) = 1$ then we have $\widetilde{\mathbf{\Omega}}'_{(I)}(x, y) = 1$. Moreover, $\widetilde{\breve{\mathbf{\Omega}}}'_{(I)}$ satisfies the above-mentioned property.

The proof of the above lemma is based a novel generalization of Hall's theorem for bipartite graph. This generalization is shown by strong induction on the number of nodes.

Lemma 19 *Assume that $n > 200$, $1 \leq i \leq j \leq d-1$ and $I = \{1, 2, \ldots, j\}$. Consider r disjoint sets $\widetilde{\mathbf{\Omega}}'_{l_{(I)}}$, each with $n - r'_l$ columns of $\widetilde{\mathbf{\Omega}}_{(I)}$ for $1 \leq l \leq r$, where $r'_l \leq r \leq \frac{n}{6}$. Let $\widetilde{\mathbf{\Omega}}'_{(I)}$ denote the union of all r sets of columns $\widetilde{\mathbf{\Omega}}'_{l_{(I)}}$'s. Assume that each column of $\widetilde{\mathbf{\Omega}}_{(I)}$ includes at least l nonzero entries, where*

$$l > \max\left\{27\ \log\left(\frac{n}{\epsilon}\right) + 9\ \log\left(\frac{2rk}{\epsilon}\right) + 18, 6r'_i\right\}. \tag{19}$$

Then, there exists an $n^j \times r(n-r'_i)$ matrix $\widetilde{\breve{\Omega}}'_{(I)}$ such that: each column has exactly r'_i+1 entries equal to one, and if $\widetilde{\breve{\Omega}}'_{(I)}(x,y)=1$ then we have $\widetilde{\Omega}'_{(I)}(x,y)=1$ and also it satisfies the following property: with probability at least $1-\frac{\epsilon}{k}$, every subset $\widetilde{\breve{\Omega}}''_{(I)}$ of columns of $\widetilde{\breve{\Omega}}'_{(I)}$ satisfies the following inequality

$$r\left(m_i(\breve{\Omega}'')-r'_i\right)\geq t, \tag{20}$$

where t is the number of columns of $\widetilde{\breve{\Omega}}''_{(I)}$ and $\breve{\Omega}''$ is the tensor corresponding to unfolding $\widetilde{\breve{\Omega}}''_{(I)}$.

Proof We first claim that with probability at least $1-\frac{\epsilon}{kr}$, every subset $\widetilde{\Omega}''_{l_{(I)}}$ of columns of $\widetilde{\Omega}'_{l_{(I)}}$ satisfies

$$m_i(\Omega''_l)-r'_i\geq t, \tag{21}$$

where t is the number of columns of $\widetilde{\Omega}''_{l_{(I)}}$ and Ω''_l is the tensor corresponding to unfolding $\widetilde{\Omega}''_{l_{(I)}}$. For simplicity we denote the above-mentioned property by Property I. According to Lemma 17, with probability at least $1-\frac{\epsilon}{2kr}$, the i-th matricization of the tensor Ω''_l includes more than $\max\left\{9\ \log\left(\frac{n}{\epsilon}\right)+3\ \log\left(\frac{2kr}{\epsilon}\right)+6, 2r'_i\right\}$ nonzero rows, $1\leq i\leq j$, and we denote this property by Property II. On the other hand, given that Property II holds for Ω''_l and according to Lemma 16, with probability at least $1-\frac{\epsilon}{2kr}$, Property I holds for Ω''_l as well. Hence, with probability at least $1-\frac{\epsilon}{kr}$, Property I holds for Ω''_l, which completes the proof our earlier claim.

Consequently, according to Lemma 18, with probability at least $1-\frac{\epsilon}{kr}$, there exists an $n^j\times(n-r'_i)$ matrix $\widetilde{\breve{\Omega}}'_{l_{(I)}}$ such that: each column has exactly r'_i+1 entries equal to one, and if $\widetilde{\breve{\Omega}}'_{l_{(I)}}(x,y)=1$ then we have $\widetilde{\Omega}'_{l_{(I)}}(x,y)=1$ and also $\breve{\Omega}'_l$ satisfies Property I. Finally define $\widetilde{\breve{\Omega}}'_{(I)}\triangleq\left[\widetilde{\breve{\Omega}}'_{1_{(I)}}|\widetilde{\breve{\Omega}}'_{2_{(I)}}|\ldots|\widetilde{\breve{\Omega}}'_{r_{(I)}}\right]$. Since each $\breve{\Omega}'_l$ satisfies Property I with probability at least $1-\frac{\epsilon}{kr}$, all $\breve{\Omega}'_l$'s satisfy Property I with probability at least $1-\frac{\epsilon}{k}$, simultaneously. Consider an arbitrary subset $\widetilde{\breve{\Omega}}''_{(I)}$ of columns of $\widetilde{\breve{\Omega}}'_{(I)}$. Let $\widetilde{\breve{\Omega}}''_{l_{(I)}}$ denote those columns of $\widetilde{\breve{\Omega}}''_{(I)}$ that belong to $\widetilde{\breve{\Omega}}'_{l_{(I)}}$ and define t_l as the number of columns of $\widetilde{\breve{\Omega}}''_{l_{(I)}}$, $1\leq l\leq r$, and define t as the number of columns of $\widetilde{\breve{\Omega}}''_{(I)}$. Without loss of generality, assume that $t_1\leq t_2\leq\ldots\leq t_r$. Also, assume that all $\breve{\Omega}'_l$'s satisfy Property I. Hence, we have

$$t=\sum_{l=1}^{r}t_l\leq rt_r\leq r\left(m_i(\breve{\Omega}''_r)-r'_i\right)\leq r\left(m_i(\breve{\Omega}'')-r'_i\right). \tag{22}$$

■

Theorem 20 *Assume that $d>2$, $n>\max\{200, r(d-2)\}$, $r\leq\frac{n}{6}$ and $I=\{1,2,\ldots,d-2\}$. Assume that each column of $\widetilde{\breve{\Omega}}_{(I)}$ includes at least l nonzero entries, where*

$$l>\max\left\{27\ \log\left(\frac{n}{\epsilon}\right)+9\ \log\left(\frac{2r(d-2)}{\epsilon}\right)+18, 6r\right\}. \tag{23}$$

Then, with probability at least $1-\epsilon$, for almost every $\mathcal{U}\in\mathbb{R}^{\overbrace{n\times\ldots\times n}^{d}}$, there exist only finitely many completions of the sampled tensor $\mathcal{U}$ with CP rank r.

Proof Define the $(d-1)$-way tensor $\mathcal{U}'\in\mathbb{R}^{\overbrace{n\times\ldots\times n}^{d-2}\times n^2}$ which is obtained through merging the $(d-1)$-th and d-th dimensions of the tensor $\mathcal{U}$. Observe that the finiteness of the number of completions of the tensor $\mathcal{U}'$ of rank r ensures the finiteness of the number of completions of the tensor $\mathcal{U}$ of rank r. For notational simplicity, let Ω and $\breve{\Omega}$ denote the $(d-1)$-way sampling pattern and constraint tensors corresponding to $\mathcal{U}'$, respectively. In order to complete the proof it suffices to show with probability at least $1-\epsilon$, conditions (i) and (ii) in Theorem 9 hold for this modified $(d-1)$-way tensor.

Now, we apply Lemma 19 for each of the numbers $r'_1=1,\ldots,r'_{d-3}=1,r'_{d-2}=r$. Also, note that since $n>r(d-2)$ we conclude $n^2>r(n-r)+(d-3)r(n-1)$, and therefore $\widetilde{\Omega}_{(I)}$ includes more than $r(n-r)+(d-3)r(n-1)$ columns. According to Lemma 19, there exist $\breve{\Omega}'_i$ for $1\leq i\leq d-2$ such that: (i) each column of $\widetilde{\breve{\Omega}}'_{i_{(I)}}$ includes r'_i+1 nonzero entries for $1\leq i\leq d-2$, and if $\widetilde{\breve{\Omega}}'_{i_{(I)}}(x,y)=1$ then we have $\widetilde{\Omega}'_{i_{(I)}}(x,y)=1$, (ii) $\widetilde{\breve{\Omega}}'_{i_{(I)}}$ includes $r(n-1)$ and $r(n-r)$ columns for $1\leq i\leq d-3$ and $i=d-2$, respectively, (iii) with probability at least $1-\frac{\epsilon}{d-2}$, every subset $\widetilde{\breve{\Omega}}''_{i_{(I)}}$ of columns of $\widetilde{\breve{\Omega}}'_{i_{(I)}}$ satisfies (20) for $r'_1=1,\ldots,r'_{d-3}=1,r'_{d-2}=r$.

Recall that each column of $\widetilde{\Omega}_{(I)}$ includes $r+1$ nonzero entries, and therefore for $1\leq i\leq d-3$ that we have $r'_i+1=2$, the column of $\widetilde{\Omega}_{(I)}$ corresponding to an column of $\widetilde{\breve{\Omega}}'_{i_{(I)}}$ has $r-1$ more nonzero entries.

Observe that $\max\left\{9\ \log\left(\frac{n}{\epsilon}\right)+3\ \log\left(\frac{2kr}{\epsilon}\right)+6, 2r\right\}\geq 2r\geq(r-1)+2$. According to Lemma 17 and given (23), for each column of $\widetilde{\breve{\Omega}}'_{i_{(I)}}$ there exists another $r-1$ zero entries (x_s,y_s) for $s\in\{1,\ldots,r-1\}$ in different rows of the $(d-2)$-th matricization of $\breve{\Omega}'_i$ from the two nonzero entries such that $\widetilde{\Omega}'_{i_{(I)}}(x_s,y_s)=1, i=1,\ldots,d-3$. Hence, for each column of $\widetilde{\breve{\Omega}}'_{i_{(I)}}$, we substitute it with the column of $\widetilde{\Omega}_{(I)}$ that the value of such $r-1$ entries (in different rows of the $(d-2)$-th matricization of $\breve{\Omega}'_i$) is 1 instead of 0 , $i=1,\ldots,d-3$. Therefore, each column of $\widetilde{\breve{\Omega}}'_{i_{(I)}}$ includes exactly $r+1$ nonzero entries for $1\leq i\leq d-3$ such that if $\widetilde{\breve{\Omega}}'_{i_{(I)}}(x,y)=1$ then we have $\widetilde{\Omega}'_{i_{(I)}}(x,y)=1$.

Let $\widetilde{\breve{\Omega}}'_{(I)}=\left[\widetilde{\breve{\Omega}}'_{1_{(I)}}|\ldots|\widetilde{\breve{\Omega}}'_{d-2_{(I)}}\right]$, which includes $r(n-r)+(d-3)r(n-1)$ columns. Therefore, $\breve{\Omega}'\in\mathbb{R}^{\overbrace{n\times\ldots\times n}^{d-2}\times t}$ is a subtensor of the constraint tensor such that $t=r(\sum_{i=1}^{d-2}n)-r^2-r(d-3)$ and also and with probability at least $1-\epsilon$, every subset $\widetilde{\breve{\Omega}}''_{i_{(I)}}$ of columns of $\widetilde{\breve{\Omega}}'_{i_{(I)}}$ satisfies (20) for $r'_1=1,\ldots,r'_{d-3}=1,r'_{d-2}=r$, simultaneously for $i=1,\ldots,d-2$.

Consider an arbitrary subset $\widetilde{\breve{\Omega}}''_{(I)}$ of columns of $\widetilde{\breve{\Omega}}'_{(I)}$. Let $\widetilde{\breve{\Omega}}''_{i_{(I)}}$ denote those columns of $\widetilde{\breve{\Omega}}''_{(I)}$ that belong to $\widetilde{\breve{\Omega}}'_{i_{(I)}}$ and define t_i as the number of columns of $\widetilde{\breve{\Omega}}''_{i_{(I)}}$, $1\leq i\leq d-2$, and define t

as the number of columns of $\widetilde{\breve{\Omega}}''_{(I)}$. Also, assume that all $\breve{\Omega}'_l$'s satisfy (20) for $r'_1 = 1, \ldots, r'_{d-3} = 1, r'_{d-2} = r$. Then, we have the following two scenarios:

(i) $t_{d-2} = 0$: Hence, we have $t = \sum_{i=1}^{d-3} t_i$. Moreover, we have

$$t_i \leq r\left(m_i(\breve{\Omega}''_i) - 1\right)^+ \leq r\left(m_i(\breve{\Omega}'') - 1\right)^+ = r\left(m_i(\breve{\Omega}'') - 1\right), \tag{24}$$

for $1 \leq i \leq d-3$. Recall that each column of $\widetilde{\breve{\Omega}}'_{i_{(I)}}$ includes at least r nonzero entries in different rows of the $(d-2)$-th matricization of $\breve{\Omega}'_i$ for $1 \leq i \leq d-3$. On the other hand, since $\widetilde{\breve{\Omega}}''_{(I)}$ includes at least one column of $\left[\widetilde{\breve{\Omega}}'_{1_{(I)}} | \ldots | \widetilde{\breve{\Omega}}'_{d-3_{(I)}}\right]$ (recall that $t_{d-2} = 0$), we have

$$r \leq m_{d-2}(\breve{\Omega}''), \tag{25}$$

which also results that $\min\left\{\max\left\{m_1(\breve{\Omega}''), \ldots, m_{d-2}(\breve{\Omega}'')\right\}, r\right\} = r$.

Therefore, having (24) and (25), we conclude

$$t = \sum_{i=1}^{d-3} t_i \leq \sum_{i=1}^{d-3} r\left(m_i(\breve{\Omega}'') - 1\right) \leq \sum_{i=1}^{d-3} r\left(m_i(\breve{\Omega}'') - 1\right) + r\left(m_{d-2}(\breve{\Omega}'') - r\right) \tag{26}$$

$$= r\left(\sum_{i=1}^{d-2} m_i(\breve{\Omega}'')\right) - r^2 - r(d-3)$$

$$= r\left(\left(\sum_{i=1}^{d-2} m_i(\breve{\Omega}'')\right) - \min\left\{\max\left\{m_1(\breve{\Omega}''), \ldots, m_{d-2}(\breve{\Omega}'')\right\}, r\right\} - (d-3)\right).$$

(ii) $t_{d-2} > 0$: Hence, we have

$$t_{d-2} \leq r\left(m_{d-2}(\breve{\Omega}''_{d-2}) - r\right) \leq r\left(m_{d-2}(\breve{\Omega}'') - r\right), \tag{27}$$

which also results that $\min\left\{\max\left\{m_1(\breve{\Omega}''), \ldots, m_{d-2}(\breve{\Omega}'')\right\}, r\right\} = r$. Moreover, similar to scenario (i), (24) holds. Therefore, having (24) and (27), we conclude

$$t = \sum_{i=1}^{d-2} t_i \leq \sum_{i=1}^{d-3} r\left(m_i(\breve{\Omega}'') - 1\right) + r\left(m_{d-2}(\breve{\Omega}'') - r\right) = r\left(\sum_{i=1}^{d-2} m_i(\breve{\Omega}'')\right) - r^2 - r(d-3) \tag{28}$$

$$= r\left(\left(\sum_{i=1}^{d-2} m_i(\breve{\Omega}'')\right) - \min\left\{\max\left\{m_1(\breve{\Omega}''), \ldots, m_{d-2}(\breve{\Omega}'')\right\}, r\right\} - (d-3)\right).$$

■

Remark 21 *Observe that each row of the d-matricization of $\mathcal{U}$ consists of exactly n columns of $\widetilde{\mathbf{\Omega}}_{(I)}$, where $I = \{1, 2, \ldots, d-2\}$. Hence, given* (23)*, Assumption* 1 *holds with probability one since Assumption* 1 *only requires r sampled entries at each row of the d-matricization of $\mathcal{U}$. Moreover, note that in the above theorem we showed that given* (23)*, constraint* (6) *holds with high probability and this ensures finite completability with high probability (since Assumption* 1 *holds with probability one).*

Note that for the general values of $n_1, \ldots, n_d$ the same proof will still work, but instead of the assumption $n > \max\{200, r(d-2)\}$, we need another assumption in terms of $n_1, \ldots, n_d$ to ensure that the corresponding unfolding has enough number of columns.

Remark 22 *A tensor $\mathcal{U}$ that satisfies the properties in the statement of Theorem 20 requires*

$$n^2 \max\left\{27\ \log\left(\frac{n}{\epsilon}\right) + 9\ \log\left(\frac{2r(d-2)}{\epsilon}\right) + 18, 6r\right\} \tag{29}$$

samples to be finitely completable with probability at least $1-\epsilon$, which is lower than the number of samples required by the unfolding approach given in (14) *by orders of magnitude.*

The following lemma is taken from (Ashraphijuo et al., 2016a) and is used in Lemma 24 to derive a lower bound on the sampling probability that results (23) with high probability.

Lemma 23 *Consider a vector with n entries where each entry is observed with probability p independently from the other entries. If $p > p' = \frac{k}{n} + \frac{1}{\sqrt[4]{n}}$, then with probability at least $\left(1 - \exp(-\frac{\sqrt{n}}{2})\right)$, more than k entries are observed.*

The proof of the above lemma is simply based on Azuma's inequality.

Lemma 24 *Assume that $d > 2$, $n > \max\{200, r(d-2)\}$ and $r \leq \frac{n}{6}$. Moreover, assume that the sampling probability satisfies*

$$p > \frac{1}{n^{d-2}} \max\left\{27\ \log\left(\frac{n}{\epsilon}\right) + 9\ \log\left(\frac{2r(d-2)}{\epsilon}\right) + 18, 6r\right\} + \frac{1}{\sqrt[4]{n^{d-2}}}. \tag{30}$$

Then, with probability at least $(1-\epsilon)\left(1 - \exp(-\frac{\sqrt{n^{d-2}}}{2})\right)^{n^2}$, $\mathcal{U}$ is finitely completable.

Proof According to Lemma 23, (30) results that each column of $\widetilde{\mathbf{\Omega}}_{(I)}$ includes at least l nonzero entries, where $I = \{1, 2, \ldots, d-2\}$ and l satisfies (23) with probability at least $\left(1 - \exp(-\frac{\sqrt{n^{d-2}}}{2})\right)$. Therefore, with probability at least $\left(1 - \exp(-\frac{\sqrt{n^{d-2}}}{2})\right)^{n^2}$, all n^2 columns of $\widetilde{\mathbf{\Omega}}_{(I)}$ satisfy (23). Hence, according to Theorem 20, with probability at least $(1-\epsilon)\left(1 - \exp(-\frac{\sqrt{n^{d-2}}}{2})\right)^{n^2}$, $\mathcal{U}$ is finitely completable. ■

5. Deterministic and Probabilistic Conditions for Unique Completability

In this section, we are interested in characterizing the deterministic and probabilistic conditions on the sampling pattern for unique completability. In previous sections we characterized the corresponding conditions for finite completability in Theorem 9 and Theorem 20. However, for matrix and tensor completion problems, finite completability does not necessarily imply unique completability (Ashraphijuo et al., 2016a). In this section, we add some additional mild restrictions on

Ω in the statement of Theorem 9 to ensure unique completability (deterministic) and also increase the number of samples given in the statement of Theorem 20 mildly to ensure unique completability with high probability (probabilistic). As the first step of this procedure, we use the following lemma for minimally algebraically dependent polynomials to obtain the variables involved in these polynomials uniquely. Hence, by obtaining all entries of the CP decomposition of the sampled tensor $\mathcal{U}$ we can show the uniqueness of $\mathcal{U}$.

Lemma 25 *Suppose that Assumption 1 holds. Let $\breve{\Omega}' \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_{d-1} \times t}$ be an arbitrary subtensor of the constraint tensor $\breve{\Omega}$. Assume that polynomials in $\mathcal{P}(\breve{\Omega}')$ are minimally algebraically dependent. Then, all variables (unknown entries) of $\mathbf{A}_1, \ldots, \mathbf{A}_{d-1}$ that are involved in $\mathcal{P}(\breve{\Omega}')$ can be determined uniquely.*

Proof According to Lemma 7, the number of involved variables in polynomials in $\mathcal{P}(\breve{\Omega}') = \{p_1, p_2, \ldots, p_t\}$ is $t-1$ and are denoted by $\{x_1, \ldots, x_{t-1}\}$. Moreover, as mentioned in the proof of Lemma 7, $\mathcal{P}(\breve{\Omega}')\backslash p_i$ is a set of algebraically independent polynomials and the number of involved variables is $t-1, i = 1, \ldots, t$. Consider an arbitrary variable x_1 that is involved in polynomials in $\mathcal{P}(\breve{\Omega}')$ and without loss of generality, assume that x_1 is involved in p_1.

On the other hand, as mentioned all variables $\{x_1, \ldots, x_{t-1}\}$ are involved in algebraically independent polynomials in $\mathcal{P}(\breve{\Omega}')\backslash p_1$. Hence, tuple $(x_1, \ldots, x_{t-1})$ can be determined finitely. Given that one of these finitely many tuples $(x_1, \ldots, x_{t-1})$ satisfy polynomial equation p_1, with probability one there does not exist another tuple $(x_1, \ldots, x_{t-1})$ among them that satisfies polynomial equation p_1. This is because with probability zero a tuple satisfy a polynomial equation in which the coefficients are chosen generically, and also the fact that the number of such tuples is finite. ■

Condition (i) in Theorem 26 results in $r(\sum_{i=1}^{d-1} n_i) - r^2 - r(d-2)$ algebraically independent polynomials in terms of the entries of $\mathbf{A}_1, \ldots, \mathbf{A}_{d-1}$, i.e., results in finite completability. Hence, adding a single polynomial corresponding to any observed entry to these $r(\sum_{i=1}^{d-1} n_i) - r^2 - r(d-2)$ algebraically independent polynomials results in a set of algebraically dependent polynomials. Then, according to Lemma 25 a subset of the entries of $\mathbf{A}_1, \ldots, \mathbf{A}_{d-1}$ can be determined uniquely and these additional polynomials are captured in the structure of condition (ii) such that all entries of CP decomposition can be determined uniquely.

Theorem 26 *Suppose that Assumption 1 holds. Also, assume that there exist disjoint subtensors $\breve{\Omega}' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t}$ and $\breve{\Omega}'^i \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t_i}$ (for $1 \leq i \leq 2d-2$) of the constraint tensor such that the following conditions hold:*

(i) $t = r(\sum_{i=1}^{d-1} n_i) - r^2 - r(d-2)$ and for any $t' \in \{1, \ldots, t\}$ and any subtensor $\breve{\Omega}'' \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t'}$ of $\breve{\Omega}'$, (6) holds.

(ii) for $i \in \{1, \ldots, d-1\}$ we have $t_i = n_i - 1$ and for $i \in \{d, \ldots, 2d-2\}$ we have $t_i = n_{i-d+1} - r$. Also, for any $t'_i \in \{1, \ldots, t_i\}$ and any subtensor $\breve{\Omega}''^i \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t'_i}$ of the tensor $\breve{\Omega}'^i$, the following inequalities hold

$$m_i(\breve{\Omega}''^i) - 1 \geq t'_i, \qquad \text{for } i \in \{1, \ldots, d-1\}, \tag{31}$$

and

$$m_{i-d+1}(\breve{\Omega}''^i) - r \geq t'_i, \qquad \text{for } i \in \{d, \ldots, 2d-2\}. \tag{32}$$

Then, for almost every $\mathcal{U}$, there exists only a unique tensor that fits in the sampled tensor $\mathcal{U}$, and has CP rank r.

Proof As we showed in the proof of Theorem 9, $\mathcal{P}(\breve{\Omega}')$ includes $t = r(\sum_{i=1}^{d-1} n_i) - r^2 - r(d-2)$ algebraically independent polynomials which results the finite completability of the sampled tensor $\mathcal{U}$. Let $\{p_1, \ldots, p_t\}$ denote these t algebraically independent polynomials in $\mathcal{P}(\breve{\Omega}')$. Now, having $\{p_1, \ldots, p_t\}$ and $\mathcal{P}(\breve{\Omega}'^i)$ for $1 \leq i \leq 2d-2$, and using Lemma 25 several times, we show the unique completability. Recall that t is the number of total variables among the polynomials, and therefore union of any polynomial p_0 and $\{p_1, \ldots, p_t\}$ is a set of algebraically dependent polynomials. Hence, there exists a set of polynomials $\mathcal{P}(\breve{\Omega}'')$ such that $\mathcal{P}(\breve{\Omega}'') \subset \{p_1, \ldots, p_t\}$ and also polynomials in $\mathcal{P}(\breve{\Omega}'') \cup p_0$ are minimally algebraically dependent polynomials. Therefore, according to Lemma 25, all variables involved in the polynomials $\mathcal{P}(\breve{\Omega}'') \cup p_0$ can be determined uniquely, and consequently, all variables involved in p_0 can be determined uniquely.

We can repeat the above procedure for any polynomial $p_0 \in \mathcal{P}(\breve{\Omega}'^i)$ to determine the involved variables uniquely with the help of $\{p_1, \ldots, p_t\}$, $i = 1, \ldots, 2d-2$. Hence, for any polynomial $p_0 \in \mathcal{P}(\breve{\Omega}'^i)$ or $p_0 \in \mathcal{P}(\breve{\Omega}'^{i+d-1})$, we obtain r degree-1 polynomials in terms of the entries of $\mathbf{A}_i$ but some of the entries of CP decomposition are elements of the $\mathbf{Q}_i$ matrices (in the statement of Lemma 3), $i = 1, \ldots, d-1$. In order to complete the proof, we need to show that condition (ii) with the above procedure using $\{p_1, \ldots, p_t\}$ results in obtaining all variables uniquely. In particular, we show that repeating the described procedure for any of the polynomials in $\mathcal{P}(\breve{\Omega}'^i)$ and $\mathcal{P}(\breve{\Omega}'^{i+d-1})$ result in obtaining all variables of the i-th element of CP decomposition uniquely.

According to Lemma 3, we have the following two scenarios for any $i \in \{1, \ldots, d-1\}$:

(i) $\mathbf{Q}_i \in \mathbb{R}^{r \times r}$: In this case, condition (ii) for $\breve{\Omega}'^{i+d-1}$ and for any $t'_{i+d-1} \in \{1, \ldots, n_i - r\}$ and any subtensor $\breve{\Omega}''^{i+d-1} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t'_{i+d-1}}$ of the tensor $\breve{\Omega}'^{i+d-1}$ results

$$rm_i(\breve{\Omega}''^{i+d-1}) - r^2 \geq rt'_{i+d-1}. \tag{33}$$

Note that rt'_{i+d-1} is the number of polynomials from the above mentioned procedure corresponding to $\breve{\Omega}''^{i+d-1}$ and $rm_i(\breve{\Omega}''^{i+d-1})$ denotes the number of involved entries of $\mathbf{A}_i$ in these polynomials, and therefore $rm_i(\breve{\Omega}''^{i+d-1}) - r^2$ is the number of involved variables of $\mathbf{A}_i$ in these polynomials. As a result, according to Fact 2, given $\mathcal{P}(\breve{\Omega}'^{i+d-1})$ and $\{p_1, \ldots, p_t\}$, the mentioned procedure results in $rn_i - r^2$ algebraically independent degree-1 polynomials in terms of the unknown entries of $\mathbf{A}_i$. Therefore, $\mathbf{A}_i$ can be determined uniquely.

(ii) $\mathbf{Q}_i \in \mathbb{R}^{1 \times r}$: Similar to scenario (i), condition (ii) for $\breve{\Omega}''^i$ and for any $t'_i \in \{1, \ldots, n_i\}$ and any subtensor $\breve{\Omega}''^i \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_{d-1} \times t'_i}$ of the tensor $\breve{\Omega}'^i$ results

$$rm_i(\breve{\Omega}''^i) - r \geq rt'_i, \tag{34}$$

and therefore similar to the previous scenario, $\mathbf{A}_i$ can be determined uniquely. ■

In Theorem 26, we obtained the deterministic condition on the sampling pattern for unique completability. Note that Condition (i) in Theorem 26 is the same condition for finite completability.

In the remainder of this section, we are interested in characterizing the probabilistic conditions on the number of samples to ensure unique completability with high probability. For the sake

of simplicity, as in Section 4 we consider the sampled tensor $\mathcal{U} \in \mathbb{R}^{\overbrace{n \times \ldots \times n}^{d}}$. Recall that for the general values of $n_1, \ldots, n_d$ the same proof will still work, but instead of assumption $n > \max\{200, (r+2)(d-2)\}$, we need another assumption in terms of $n_1, \ldots, n_d$.

Theorem 27 *Assume that $d > 2$, $n > \max\{200, (r+2)(d-2)\}$, $r \leq \frac{n}{6}$ and $I = \{1, 2, \ldots, d-2\}$. Assume that each column of $\widetilde{\mathbf{\Omega}}_{(I)}$ includes at least l nonzero entries, where*

$$l > \max\left\{27 \log\left(\frac{2n}{\epsilon}\right) + 9 \log\left(\frac{8r(d-2)}{\epsilon}\right) + 18, 6r\right\}. \tag{35}$$

Then, with probability at least $1-\epsilon$, for almost every $\mathcal{U} \in \mathbb{R}^{\overbrace{n \times \ldots \times n}^{d}}$, there exist only one completion of the sampled tensor $\mathcal{U}$ with CP rank r.

Proof Similar to the proof of Theorem 20, define the $(d-1)$-way tensor $\mathcal{U}' \in \mathbb{R}^{\overbrace{n \times \ldots \times n}^{d-2} \times n^2}$ which is obtained through merging the $(d-1)$-th and d-th dimensions of the tensor $\mathcal{U}$ and recall that the finiteness of the number of completions of the tensor $\mathcal{U}'$ of rank r ensures the finiteness of the number of completions of the tensor $\mathcal{U}$ with rank r. Similarly, for simplicity, assume that Ω and $\breve{\Omega}$ denote the $(d-1)$-way sampling pattern and constraint tensors corresponding to $\mathcal{U}'$, respectively. Note that since $n > (r+2)(d-2)$, we conclude $n^2 > r(n-r) + (d-3)r(n-1) + 2(d-2)$, and therefore $\widetilde{\mathbf{\Omega}}_{(I)}$ includes more than $r(n-r) + (d-3)r(n-1) + 2n(d-2)$ columns. According to the proof of Theorem 20, considering $r(n-r) + (d-3)r(n-1)$ arbitrary columns of $\widetilde{\mathbf{\Omega}}_{(I)}$ results in existence of $\breve{\Omega}'$ such that condition (i) holds with probability at least $1 - \frac{\epsilon}{2}$. Also, there exist at least $2n(d-2)$ columns other than these $r(n-r) + (d-3)r(n-1)$ columns.

Consider $n-1$ arbitrary columns of $\widetilde{\mathbf{\Omega}}_{(I)}$. By setting $r = 1$ in the statement of Lemma 19, these $n-1$ columns result in $\breve{\Omega}''^{i}$ with $n-1$ columns such that with probability at least $1 - \frac{\epsilon}{4r(d-2)}$, (31) holds. Similarly, consider $n-r$ arbitrary columns of $\widetilde{\mathbf{\Omega}}_{(I)}$. Then, there exists $\breve{\Omega}''^{i+d-2}$ such that with probability at least $1 - \frac{\epsilon}{4r(d-2)}$, (32) holds. Hence, condition (ii) holds (for all $i \in \{1, \ldots, 2d-4\}$) with probability at least $1 - \frac{\epsilon}{2r}$. Therefore, conditions (i) and (ii) hold with probability at least $1 - \left(\frac{\epsilon}{2r} + \frac{\epsilon}{2}\right) \geq 1 - \epsilon$. ■

Remark 28 *A tensor $\mathcal{U}$ that satisfies the properties in the statement of Theorem 27 requires*

$$n^2 \max\left\{27 \log\left(\frac{2n}{\epsilon}\right) + 9 \log\left(\frac{8r(d-2)}{\epsilon}\right) + 18, 6r\right\} \tag{36}$$

samples to be uniquely completable with probability at least $1-\epsilon$, which is orders-of-magnitude lower than the number of samples required by the unfolding approach given in (14)*. Note that the number of samples given in Theorem 3 of (Pimentel-Alarcón et al., 2016d) results in both finite and unique completability, and therefore the number of samples required by the unfolding approach given in Remark 15 is for both finite and unique completability.*

Lemma 29 *Assume that $d > 2$, $n > \max\{200, (r+2)(d-2)\}$ and $r \leq \frac{n}{6}$. Moreover, assume that the sampling probability satisfies*

$$p > \frac{1}{n^{d-2}} \max\left\{27 \log\left(\frac{2n}{\epsilon}\right) + 9 \log\left(\frac{8r(d-2)}{\epsilon}\right) + 18, 6r\right\} + \frac{1}{\sqrt[4]{n^{d-2}}} \tag{37}$$

Then, with probability at least $(1-\epsilon)\left(1 - \exp(-\frac{\sqrt{n^{d-2}}}{2})\right)^{n^2}$, $\mathcal{U}$ is finitely completable.

Proof The proof is similar to the proof of Lemma 24. ■

Remark 30 *Theorem 27 ensures that there exist enough number of algebraically independent polynomials and according to Theorem 26, these algebraically independent polynomials can be distinguished among all the polynomials, which has a unique solution that leads to the unique completion of the sampled tensor. Hence, Theorem 27 characterizes an information-theoretic sampling rate for completing the sampled data. However, one direction for future research is to propose an algorithm that solves this particular system of polynomials obtained from Theorem 26 in order to achieve the bound in Theorem 27 and close the gap. Note that there are some fast methods to approximate the zero of the system polynomials including generalizations of Newton-Raphson method, which is independent from this work as in this work we find the information-theoretic rate of sampling to ensure the proposed combinatorial deterministic conditions hold.*

6. Numerical Comparisons

In order to show the advantage of our proposed CP approach over the unfolding approach, we compare the lower bound on the total number of samples that is required for finite completability using an example. Since the bound on the number of samples for finiteness and uniqueness are the same for the unfolding approach and they are almost the same for the CP approach, we only consider finiteness bounds for this example. In particular, we consider a 7-way tensor $\mathcal{U}$ ($d = 7$) such that each dimension size is $n = 10^3$. We also consider the CP rank r which varies from 1 to 150. Figure 1 plots the bounds given in (14) (unfolding approach) and in (29) (CP approach) for the corresponding rank value, where $\epsilon = 0.001$. It is seen that the number of samples required by the proposed CP approach is substantially lower than that is required by the unfolding approach.

Observe that the unfolding approach does not take advantage of the structural property of the underlying CP model properly, and therefore our proposed multi-way analysis on CP manifold significantly outperforms the simple two-way analysis.

7. Conclusions

This paper is concerned with the low CP rank tensor completion problem and aims to derive fundamental conditions on the sampling pattern for finite and unique completability of a sampled tensor given its CP rank. In order to do so, a novel algebraic geometry analysis on the CP manifold is proposed. In particular, each sampled entry can be treated as a polynomials in terms of the entries of the components of the CP decomposition. We have defined a geometric pattern which classifies all CP decompositions such that each class includes only one decomposition of any tensor. We have

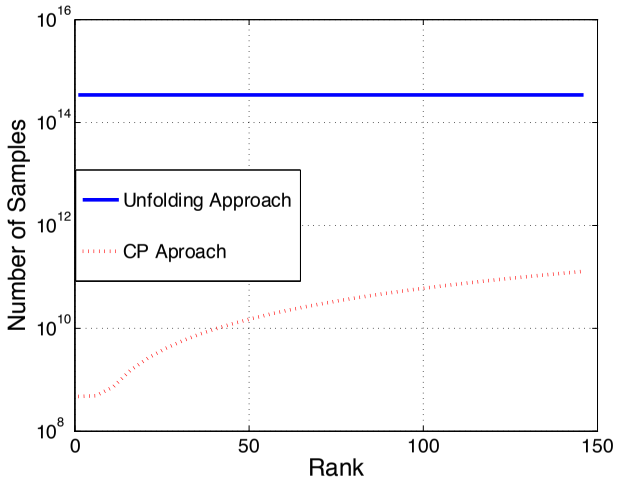


Figure 1: Lower bounds on the number of samples for a 7-way tensor.

shown that finite completability is equivalent to having a certain number of algebraically independent polynomials among all the defined polynomials based on the sampled entries. Furthermore, using the proposed classification, we can characterize the maximum number of algebraically independent polynomials in terms of a simple function of the sampling pattern. Moreover, we have developed several combinatorial tools that are used to bound the number of samples to ensure finite completability with high probability. Using these developed tools, we have treated three problems in this paper: (i) Characterizing the deterministic necessary and sufficient conditions on the sampling pattern, under which there are only finitely many completions given the CP rank, (ii) Characterizing deterministic sufficient conditions on the sampling pattern, under which there exists exactly one completion given the CP rank, (iii) Deriving lower bounds on the sampling probability or the number of samples such that the obtained deterministic conditions in Problems (i) and (ii) are satisfied with high probability. In addition, it is seen that our proposed CP analysis leads to an orders-of-magnitude lower number of samples than the unfolding approach that is based on analysis on the Grassmannian manifold.

Acknowledgments

This work was supported in part by the U.S. National Science Foundation (NSF) under grant CIF1064575, and in part by the U.S. Office of Naval Research (ONR) under grant N000141410667.

References

Ralph Abraham, Jerrold E Marsden, and Tudor Ratiu. *Manifolds, Tensor Analysis, and Applications*, volume 75. Springer Science & Business Media, 2012.

Morteza Ashraphijuo and Xiaodong Wang. Characterization of deterministic and probabilistic sampling patterns for finite completability of low tensor-train rank tensor. *arXiv preprint:1703.07698*, 2017.

Morteza Ashraphijuo, Ramtin Madani, and Javad Lavaei. Inverse function theorem for polynomial equations using semidefinite programming. In *IEEE 54th Annual Conference on Decision and Control (CDC)*, pages 6589–6596, 2015.

Morteza Ashraphijuo, Vaneet Aggarwal, and Xiaodong Wang. Deterministic and probabilistic conditions for finite completability of low rank tensor. *arXiv preprint:1612.01597*, 2016a.

Morteza Ashraphijuo, Ramtin Madani, and Javad Lavaei. Characterization of rank-constrained feasibility problems via a finite number of convex programs. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6544–6550, 2016b.

Morteza Ashraphijuo, Vaneet Aggarwal, and Xiaodong Wang. A characterization of sampling patterns for low-Tucker-rank tensor completion problem. In *IEEE International Symposium on Information Theory (ISIT)*, pages 531–535, 2017a.

Morteza Ashraphijuo, Xiaodong Wang, and Vaneet Aggarwal. A characterization of sampling patterns for low-rank multi-view data completion problem. In *IEEE International Symposium on Information Theory (ISIT)*, pages 1147–1151, 2017b.

Morteza Ashraphijuo, Xiaodong Wang, and Vaneet Aggarwal. Deterministic and probabilistic conditions for finite completability of low-rank multi-view data. *arXiv preprint:1701.00737*, 2017c.

Morteza Ashraphijuo, Xiaodong Wang, and Vaneet Aggarwal. Rank determination for low-rank data completion. *arXiv preprint:1707.00622*, 2017d.

Jian Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.

Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

Emmanuel J Candès, Yonina C Eldar, Thomas Strohmer, and Vladislav Voroninski. Phase retrieval via matrix completion. *SIAM Journal on Imaging Sciences*, 6(1):199–225, 2013.

J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an N-way generalization of Eckart-Young decomposition. *Psychometrika*, 35(3):283–319, 1970.

Lieven De Lathauwer. A survey of tensor methods. In *IEEE International Symposium on Circuits and Systems*, pages 2773–2776, 2009.

Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

Gregory Ely, Shuchin Aeron, Ning Hao, Misha E Kilmer, et al. 5d and 4d pre-stack seismic data completion using tensor nuclear norm (tnn). In *Society of Exploration Geophysicists*, 2013.

Silvia Gandy, Benjamin Recht, and Isao Yamada. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Problems*, 27(2):1–19, 2011.

Lars Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM Journal on Matrix Analysis and Applications*, 31(4):2029–2054, 2010.

Lars Grasedyck, Daniel Kressner, and Christine Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM-Mitteilungen*, 36(1):53–78, 2013.

Richard A Harshman. Foundations of the PARAFAC procedure: Models and conditions for an explanatory multi-modal factor analysis. *University of California at Los Angeles*, 1970.

Christopher J Hillar and Lek-Heng Lim. Most tensor problems are NP-hard. *Journal of the ACM (JACM)*, 60(6):45:1–45:39, 2013.

Sebastian Holtz, Thorsten Rohwedder, and Reinhold Schneider. On manifolds of tensors of fixed TT-rank. *Numerische Mathematik*, 120(4):701–731, 2012.

Prateek Jain and Sewoong Oh. Provable tensor factorization with missing data. In *Advances in Neural Information Processing Systems*, pages 1431–1439, 2014.

Misha E Kilmer, Karen Braman, Ning Hao, and Randy C Hoover. Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging. *SIAM Journal on Matrix Analysis and Applications*, 34(1):148–172, 2013.

Tamara G Kolda. Orthogonal tensor decompositions. *SIAM Journal on Matrix Analysis and Applications*, 23(1):243–255, 2001.

Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM Review*, 51(3): 455–500, 2009.

Nadia Kreimer, Aaron Stanton, and Mauricio D Sacchi. Tensor completion based on nuclear norm minimization for 5D seismic data reconstruction. *Geophysics*, 78(6):V273–V284, 2013.

Daniel Kressner, Michael Steinlechner, and Bart Vandereycken. Low-rank tensor completion by Riemannian optimization. *BIT Numerical Mathematics*, 54(2):447–468, 2014.

Akshay Krishnamurthy and Aarti Singh. Low-rank matrix and tensor completion via adaptive sampling. In *Advances in Neural Information Processing Systems*, pages 836–844, 2013.

Lek-Heng Lim and Pierre Comon. Multiarray signal processing: Tensor decomposition meets compressed sensing. *Comptes Rendus Mecanique*, 338(6):311–320, 2010.

Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):208–220, 2013.

Xiao-Yang Liu, Shuchin Aeron, Vaneet Aggarwal, and Xiaodong Wang. Low-tubal-rank tensor completion using alternating minimization. *arXiv preprint:1610.01690*, 2016.

Damien Muti and Salah Bourennane. Survey on tensor signal algebraic filtering. *Signal Processing*, 87(2):237–249, 2007.

Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

Evangelos E Papalexakis, Christos Faloutsos, and Nicholas D Sidiropoulos. ParCube: Sparse parallelizable tensor decompositions. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 521–536, 2012.

D Pimentel-Alarcón, Nigel Boston, and Robert D Nowak. Deterministic conditions for subspace identifiability from incomplete sampling. In *IEEE International Symposium on Information Theory (ISIT)*, pages 2191–2195, 2015.

D Pimentel-Alarcón, L Balzano, R Marcia, R Nowak, and R Willett. Group-sparse subspace clustering with missing data. In *IEEE Workshop on Statistical Signal Processing (SSP)*, pages 1–5, 2016a.

D Pimentel-Alarcón, EDU Robert D Nowak, and WISC EDU. The information-theoretic requirements of subspace clustering with missing data. In *International Conference on Machine Learning*, 2016b.

Daniel Pimentel-Alarcón, Laura Balzano, and Robert Nowak. Necessary and sufficient conditions for sketched subspace clustering. In *Allerton Conference on Communication, Control, and Computing*, 2016c.

Daniel Pimentel-Alarcón, Nigel Boston, and Robert Nowak. A characterization of deterministic sampling patterns for low-rank matrix completion. *IEEE Journal of Selected Topics in Signal Processing*, 10(4):623–636, 2016d.

Bernardino Romera-Paredes and Massimiliano Pontil. A new convex relaxation for tensor completion. In *Advances in Neural Information Processing Systems*, pages 2967–2975, 2013.

Nicholas D Sidiropoulos and Anastasios Kyrillidis. Multi-way compressed sensing for sparse low-rank tensors. *IEEE Signal Processing Letters*, 19(11):757–760, 2012.

Marco Signoretto, Raf Van de Plas, Bart De Moor, and Johan AK Suykens. Tensor versus matrix completion: a comparison with application to spectral data. *IEEE Signal Processing Letters*, 18 (7):403–406, 2011.

Marco Signoretto, Quoc Tran Dinh, Lieven De Lathauwer, and Johan AK Suykens. Learning with tensors: a framework based on convex optimization and spectral regularization. *Machine Learning*, 94(3):303–351, 2014.

Alwin Stegeman and Nicholas D Sidiropoulos. On Kruskals uniqueness condition for the CANDECOMP/PARAFAC decomposition. *Linear Algebra and its Applications*, 420(2):540–552, 2007.

Bernd Sturmfels. *Solving Systems of Polynomial Equations*. Number 97. American Mathematical Society, 2002.

Jos MF ten Berge and Nikolaos D Sidiropoulos. On uniqueness in CANDECOMP/PARAFAC. *Psychometrika*, 67(3):399–409, 2002.

Ryota Tomioka, Kohei Hayashi, and Hisashi Kashima. Estimation of low-rank tensors via convex optimization. *arXiv preprint:1010.0789*, 2010.

Wenqi Wang, Vaneet Aggarwal, and Shuchin Aeron. Tensor completion by alternating minimization under the tensor train (TT) model. *arXiv preprint:1609.05587*, Sep. 2016.

Journal of Machine Learning Research 18 (2017) 1-34 Submitted 8/14; Revised 6/17; Published 8/17

Parallel Symmetric Class Expression Learning

An C. Tran TCAN@CIT.CTU.EDU.VN
College of Information and Communication Technology
Can Tho University, Vietnam

Jens Dietrich J.B.DIETRICH@MASSEY.AC.NZ
Hans W. Guesgen H.W.GUESGEN@MASSEY.AC.NZ
Stephen Marsland S.R.MARSLAND@MASSEY.AC.NZ
School of Engineering and Advanced Technology
Massey University, New Zealand.

Editor: Luc De Raedt

Abstract

In machine learning, one often encounters data sets where a general pattern is violated by a relatively small number of exceptions (for example, a rule that says that all birds can fly is violated by examples such as penguins). This complicates the concept learning process and may lead to the rejection of some simple and expressive rules that cover many cases. In this paper we present an approach to this problem in description logic learning by computing partial descriptions (which are not necessarily entirely complete) of both positive and negative examples and combining them. Our *Symmetric Parallel Class Expression Learning* approach enables the generation of general rules with exception patterns included. We demonstrate that this algorithm provides significantly better results (in terms of metrics such as accuracy, search space covered, and learning time) than standard approaches on some typical data sets. Further, the approach has the added benefit that it can be parallelised relatively simply, leading to much faster exploration of the search tree on modern computers.

Keywords: description logic learning, parallel, symmetric, exception

1. Introduction

In machine learning we seek general rules that describe data. However, it is not uncommon for there to be exceptions to general rules, i.e., particular situations where those rules do not apply. In this situation, non-monotonic reasoning can be used, and a default rule that covers most of the cases is retracted if further evidence is provided that overrides the general case. Non-monotonic reason will be discussed more in Section 4.1.

For example, in the classic *Tweety* demonstration of logical proof, the first assertion is that *'Birds fly'*. This is *generally* true, and so the follow-up statement that *'Tweety is a bird'* has the implication that *'Tweety can fly'*. However, there are several species of bird that cannot fly, such as penguins. This type of example can be covered by a set of extra statements: *'Penguins are birds. Penguins do not fly. Tweety is a penguin.'* Non-monotonic reasoning is able to retract the earlier inference that *'Tweety can fly'* given the extra information that *'Tweety is a penguin.'*

©2017 A.C. Tran, J. Dietrich, H.W. Guesgen and S. Marsland.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/14-317.html.

An alternative approach to this is to make the general rules correct by listing the exceptions: *'Birds fly **except** penguins'*. Providing that the complete rule is still 'simpler' (e.g., shorter) than other rules that describe the same situation (i.e., enumerations, or disjunctions of types of birds that can fly), this can provide a net gain in learning. Thus, a short general rule can be augmented by a set of exception rules that make the general rule correct. In the Tweety case this could be a list of counter-examples, possibly by species and genus, such as *'All birds can fly except ratites and penguins'*.

In this paper, we enable such use of exception for description logics by explicitly considering negative examples and building rules for them. As we will show, this leads to a parallel algorithm that treats positive and negative examples in exactly the same way (thus meaning that the learning would be invariant to a switch between positive and negative in the data). Experiments on a variety of standard learning problems demonstrate that this algorithm is statistically significantly faster and more accurate than standard approaches from the literature.

Most existing description logic (DL) learning algorithms, e.g. $\mathcal{AL}$ − CELOE (Lisi and Malerba, 2003), DL-FOIL (Fanizzi et al., 2008), CELOE (Lehmann and Hitzler, 2010) and ParCEL (Tran et al., 2012), only take the definitions of negative examples into account implicitly, by incorporating them into the definitions of positive examples. Learning starts from a very general concept, usually 'top' in DL, and subclasses and sub-properties, conjunction (intersection) and the combination of conjunction and negation (subtraction) are used to remove the negative examples from the candidate concepts.

While this approach is suitable for many concept learning problems and has been used successfully by Hellmann (2008); Lehmann and Hitzler (2010); Tran et al. (2012), it does not use descriptions in the search space efficiently. Consider a top-down approach for learning such as CELOE, which constructs the search tree using a downward refinement operator until it finds a node (expression) that correctly and completely defines the positive examples. Given a concept learning problem with a set of positive (+) and negative (−) examples, the concepts D_1, D_2, N_1, N_2 and their coverage[1] as described in Figure 1(a), CELOE will find the single concept $C = (D_1 \sqcap \neg N_1) \sqcup (D_2 \sqcap \neg N_2)$ (shown in Figure 1(b)).

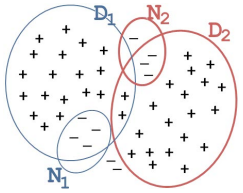
Algorithms that combine both top-down and bottom-up approaches (such as ParCEL and DL-FOIL) find two simpler concepts: $P_1 = D_1 \sqcap \neg N_1$ and $P_2 = D_2 \sqcap \neg N_2$. These concepts are visualised in Figure 1(c). If all of the concepts D_1, D_2, N_1 and N_2 have the same length of 3, then the length of the longest concepts generated by these two approaches are 16 (4 concepts of length 3 plus 4 operators) and 8 (2 concepts of length 3 plus 2 operators) respectively. If a concept is of length 16 then it must occur at a depth of at least 16 in the search tree, and so CELOE will only be able to identify this concept after searching the previous 15 levels of the search tree. As the expansion of the search tree in these approaches is directed by a heuristic that is based upon accuracy, not all branches of the search tree are fully expanded; this prevents 'dummy' breadth-first expansion.

The difference between these two methods is the way that the search tree is constructed: CELOE refines a general concept, while ParCEL searches for partial solutions that provide a correct (true) description of a subset of the data, and combines them to generate a complete solution. This idea of partial solutions can be extended to define exceptions as

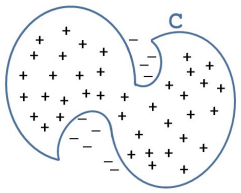
1. Coverage of a concept is the set of its instances, see Definition 1

Figure 1: Exception patterns in learning. Pluses (+) denote positive examples, minuses (–) denote negative examples. The ellipses represent coverage of the corresponding class expressions.

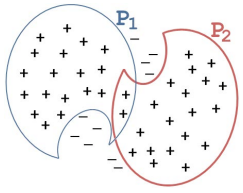
(a) The target concept can be constituted from simple descriptions D_1, D_2, N_1 and N_2: $(D_1 \sqcap \neg N_1) \sqcup (D_2 \sqcap \neg N_2)$. The description would be given by the SPaCEL learning algorithm described in this paper.



(b) CELOE learnt concept: a single complex description $C = (D_1 \sqcap \neg N_1) \sqcup (D_2 \sqcap \neg N_2)$ is required



(c) ParCEL learnt concepts: constructed from more complex descriptions $P_1 = D_1 \sqcap \neg N_1$ and $P_2 = D_2 \sqcap \neg N_2$ describe the data set concisely



partial descriptions of negative examples, and this is what we consider in this paper. It enables a learning algorithm to find the solution suggested in Figure 1(a), which is at depth three in the search tree that combines descriptions of positive and negative examples separately. This is the main idea of the algorithm that we propose in this paper, which we call SPaCEL for reasons that will be made clear later, and which treats positive and negative examples equally and can search for their definition in parallel. Definitions of negative examples can then be combined with other concepts in the search tree to construct new partial solutions. In the case of the example in Figure 1, definitions N_1 and N_2 of negative examples, which are ignored by CELOE and ParCEL, are employed by SPaCEL and are combined with concepts D_1 and D_2 to to constitute partial solutions $D_1 \sqcap \neg N_1$ (equivalent to P_1 in CELOE) and $D_2 \sqcap \neg N_2$ (equivalent to P_2 in CELOE). In some cases, definitions of negative examples can be very simple and thus they can be found very early in the search process. In addition, each negative example can be combined with a variety of concepts to constitute several partial solutions. This will result in smaller search trees and faster learning time. Our evaluation shows that this approach works well not only in cases where exceptions exist, but also in normal cases.

Figure 2, which extends the *Tweety* example using the knowledge base shown in Figure 2(a), provides another simple example of the benefits of exceptions. For the instances given in Figure 2(b) (where b_i are instances of Bird, p_i instances of Penguin, c_k instances of Mammal and t_m instances of Bat), the p_i can be considered as exceptions to a rule that says that birds fly, and likewise t_m for a rule that says that mammals do not. Figure 2(c) represents the search tree for the three learning strategies described above. Finding a complete definition (triple-line node, e.g., CELOE) requires a deeper search tree than finding several shorter partial definitions (double-line nodes, e.g. ParCEL, DL-FOIL). However, using definitions of exceptions (dotted-line nodes) can produce even shallower search trees.

Further, since some 'useful' definitions (such as `Bat` in this example) are based on negative examples, they may be ignored by the first two approaches.

Another, more complex example that demonstrates differences between the three approaches is in the uncle relationship for the Forte Uncle data set. The knowledge base of this data set contains concepts and relations that are related to family relationships, such as *Person, Male, Female, married, has parent, has sibling*, etc. It also includes some instances of people and their properties. For example, *Alfred* is a male and is the father of David and Elisa. *Alice* is a female, she is married to Art and is the mother of F14, M13, M15. *Art* is a male and he is the father of F14, M13 and M15. He also has two siblings, Umo and Wendy. *Wendy* is a female and is married to Walt. She has two children, F12 and M11. Her siblings are Art and Umo. In this set, Art is a positive example of an uncle, and Alfred, Alice and Wendy are negative examples (Alfred does not have siblings, Alice and Wendy are female). There are a total of 186 instances in the data set. A detailed description of this data set is provided in Section 3.1.

Written in Manchester syntax (Horridge and Patel-Schneider, 2009) and also in English, definitions of the uncle relationship learnt by the three different approaches are :

- **CELOE:** generates a single complex solution of length 15:

  male AND ((married SOME hasSibling SOME Person) OR
  (married SOME hasSibling SOME hasChild SOME Thing))

  (An uncle is a male who has married someone with a sibling; or a man who married someone's sibling and the sibling has some children)

- **ParCEL:** generates two partial solutions of length 7 and 9:

  1. male AND hasSibling SOME hasChild SOME Thing

     (An uncle is a man who has at least sibling and a sibling has children)

  2. male AND married SOME hasSibling SOME hasChild SOME Thing

     (An uncle is a man who married someone with at least one sibling, and that sibling has a child)

- **SPaCEL:** generates two partial solutions that are created by combining two concepts "hasSibling SOME hasChild SOME Thing" of length 5 and "married SOME hasSibling SOME hasChild SOME Thing" of length 7 with a partial solution "female" of negative examples:

  1. hasSibling SOME hasChild SOME Thing AND (NOT female)

     (An uncle is a person who has a sibling with children, and who is not female)

  2. married SOME hasSibling SOME hasChild SOME Thing AND (NOT female)

     (An uncle is a person who married someone with a sibling who has children and who is not female)

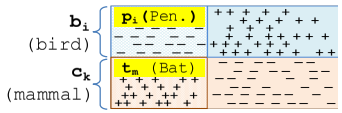
Figure 2: Different approaches to learning the definition for the *extended Tweety* concept learning problem.

(a) The knowledge base of the learning problem.

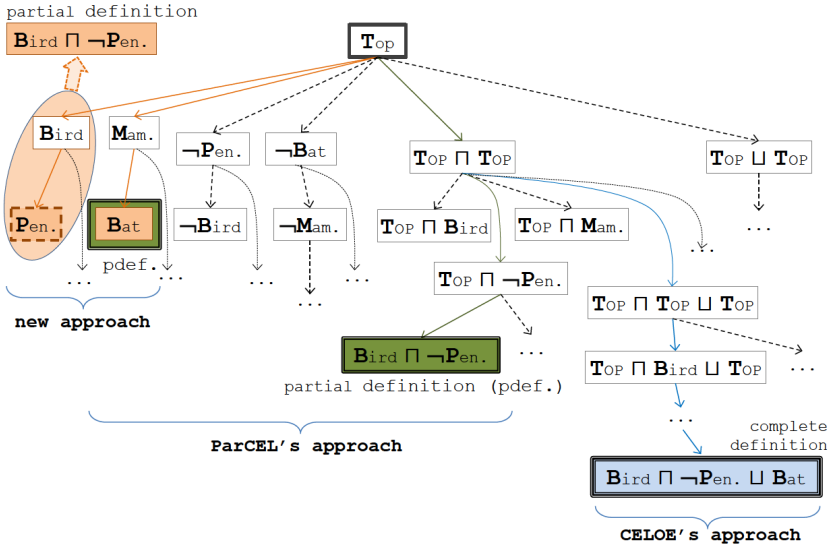
Concepts	Instances
Top	
Bird	$\{b_i\}$
Penguin	$\{p_j\}$
Mammal	$\{c_k\}$
Bat	$\{t_m\}$
Fly	$\{b_i, t_m\}$
¬Fly	$\{p_j, c_k\}$

observations

(b) Visualisation of the knowledge base.



(c) The search tree for learning the *extended Tweety* example using three different approaches. *Triple-line nodes* represent complete definitions (e.g., CELOE). *Double-line nodes* represent partial definitions (e.g., ParCEL). *Dashed-line nodes* represent definitions of some negative examples. The *ellipses* represent the combinations of an (incorrect and incomplete) expression with a definition of negative examples that constitutes a partial definition. *Solid arrows* represent the paths to the definitions for positive or negative examples.



The partial solution of negative examples can be found in an early stage of the learning progress. However, it is ignored by CELOE and ParCEL, whereas SPaCEL employs it for later combinations with concepts in the search tree to generate partial solutions if possible. All three solutions are valid, but the SPaCEL one can be found more quickly.

Using exceptions could increase the risk of selective over-fitting, since sub-cases are given high weight, but the emphasis on short descriptions for the general rules, and the fact that the exception cases are also based on short descriptions, ensures that this is unlikely to happen. We present experimental evidence that this method often gives higher predictive accuracy on test data. It is true that logical formulae based on exceptions can quickly become torturous, as a short general description gets progressively more complex caveats added to it, and possibly even exceptions to the exceptions. However, the constraints on search that are implicit in logic programming (such as search tree size, definition length, and learning time) means that these complex cases are very unlikely to ever be found.

One key part of our approach is that positive and negative examples are treated in the same way; this 'symmetry' gives us the name of our algorithm: *Symmetric Parallel Class Expression Learning* (SPaCEL). A secondary point (referred to in the 'parallel' part of the name) is that the algorithm is inherently parallelisable in the first phase of learning, where separate definitions of positive and negative examples are sought. These definitions are combined in the second phase to construct a sufficiently accurate definition. By using rules about negative examples to remove them from the concepts in the search tree it is possible to combine (potentially very short) concepts and so define relatively complex concepts based on much smaller depth traversals. Consequently, concepts in the search tree are used more effectively and thus the learning time can be reduced.

We present a formal definition of our learning algorithm in Section 2, together with a discussion of various strategies for combining partial definitions of positive and negative examples. In addition, we demonstrate the parallelisation strategy that is one part of what makes our algorithm so effective computationally. We then present the data sets that we use for evaluation in Section 3.1, followed by a comparison of the three strategies for combining definitions that we have considered. Following this, in Section 3.3 we report search tree size, predictive accuracy, learning time, and definition length for 15 data sets of varying complexity for CELOE, ParCEL and SPaCEL. In Section 4.1, we summarise some related work before concluding with a summary and discussion of future work in Section 5.

2. Symmetric Parallel Class Expression Learning

Our learning approach constructs definitions of both positive and negative examples and gives them equal weight in order to find the final definition. This is performed in three main steps:

1. Find partial definitions for positive and negative examples separately such that the set of definitions, together, cover all positive and negative examples.
2. Consolidate the partial definitions to remove redundancies (i.e., the partial definitions whose coverage is a subset of the union of other partial definitions) and select the best candidates for constructing a complete definition.
3. Aggregate the best candidates to form the complete definition using disjunction.

In the first step, partial definitions can be produced directly by specialisation using a downward refinement operator, as in algorithms such as DL-FOIL and ParCEL. Definitions for negative examples are constructed side-by-side with definitions for positive examples and are used in combination with existing expressions in the search tree to create further partial definitions.

In the next section we introduce the concepts necessary for our algorithms. The algorithms themselves will be presented in Section 2.2. We follow the definitions used by Lehmann and Hitzler (2010) closely, although some of the concepts have to be extended in order to include negative examples.

2.1 Symmetric class expression learning

A *concept learning problem* in description logics can be described as a structure $\langle \mathcal{K}, (\mathcal{E}^+, \mathcal{E}^-) \rangle$ where $\mathcal{K}$ is a knowledge base (ontology), $\mathcal{E}^+$ is a set of positive examples and $\mathcal{E}^-$ is a set of negative examples such that $\mathcal{E}^+ \cap \mathcal{E}^- = \emptyset$. A learnt concept is also called a *hypothesis* or *definition*. $\mathcal{K} \models C(a)$ denotes that a is an instance of class C with respect to the knowledge base $\mathcal{K}$. If $\mathcal{K} \models C(a)$ is satisfied, C is said to *cover* a with respect to $\mathcal{K}$. The aim of *description logic learning* is to find a concept C such that $\mathcal{K} \models C(e)$ for all $e \in \mathcal{E}^+$ (*completeness*) and $\mathcal{K} \not\models C(e)$ for all $e \in \mathcal{E}^-$ (*correctness*). A learnt concept is called *complete* if it covers all positive examples, *correct* if it does not cover any negative examples, *accurate* if it is both complete and correct, *overly general* if it is complete but incorrect, and *overly specific* if it is correct but incomplete.

In order to define negative examples, we extend the concept *cover* for a set of instances in the form of a function, as follows:

Definition 1 (Cover) *Let $\mathcal{K}$ be a logic knowledge base, $\mathcal{X}$ be a set of instances and* C *be a concept. Then,* $\mathrm{cover}(\mathcal{K}, \mathrm{C}, \mathcal{X})$ *is a function that computes a set of examples in $\mathcal{X}$ covered by* C *with respect to $\mathcal{K}$:*

$$\mathrm{cover}(\mathcal{K}, \mathrm{C}, \mathcal{X}) = \{\mathrm{e} \in \mathcal{X} \mid \mathcal{K} \models C(e) (\text{that is:e } \textit{is covered by } \mathrm{C} \textit{ with respect to } \mathcal{K})\}$$

This can be generalised to a set of definitions as:

$$cover(\mathcal{K}, \mathcal{Q}, \mathcal{X}) = \bigcup_{C \in \mathcal{Q}} cover(\mathcal{K}, \mathrm{C}, \mathcal{X})$$

where $\mathcal{K}$ be a knowledge base, $\mathcal{X}$ be a set of instances and $\mathcal{Q}$ be a set of definitions.

Definition 1 enables us to give definitions of positive examples (which are called *partial definitions*) and negative examples (which are called *counter-partial definitions*):

Definition 2 (Partial definition) *For a concept learning problem $\langle \mathcal{K}, (\mathcal{E}^+, \mathcal{E}^-) \rangle$, a concept* C *is called a* partial definition *if* $\mathrm{cover}(\mathcal{K}, \mathrm{C}, \mathcal{E}^+) \neq \emptyset$ *and* $\mathrm{cover}(\mathcal{K}, \mathrm{C}, \mathcal{E}^-) = \emptyset$.

Definition 3 (Counter-partial definition) *For a concept learning problem $\langle \mathcal{K}, (\mathcal{E}^+, \mathcal{E}^-) \rangle$, a concept* C *is called a* counter-partial definition *if* $\mathrm{cover}(\mathcal{K}, \mathrm{C}, \mathcal{E}^+) = \emptyset$ *and* $\mathrm{cover}(\mathcal{K}, \mathrm{C}, \mathcal{E}^-) \neq \emptyset$.

Definitions 2 and 3 are 'ideal' in that they insist on the cover of the opposite set being the empty set. In practice, it is possible to allow a few negative examples to be covered by a partial definition, and a few positive examples by a counter-partial definition; this could potentially help with noisy data.

The first (top-down) step of our algorithm consists of finding partial and counter-partial definitions separately using a refinement operator and a combination strategy to compute further partial definitions from the counter-partial definitions and existing expressions in the search tree. As these searches are performed separately and simultaneously, our approach avoids being over-specific for concept learning problems that contain exceptions.

These sets are then combined, which enables the algorithm to cope with exceptions to general rules through the opposing definitions (so rules made of partial definitions can have counter-partial definitions added, and vice versa). This is simply the use of conjunction and negation, The combination acts as an extra step to deal with the exceptions with the support of counter-partial definitions. In our approach, only one downward refinement operator is used in the specialisation step to generate both partial and counter-partial definitions. The combination strategy essentially consists of checking for the possibility of creating new partial definitions from an expression and counter-partial definitions, using conjunction and negation: a combination of an expression C with a set of counter-partial definitions $\mathcal{X}$ has the form $C \sqcap \neg(\sqcup_{D \in \mathcal{X}} D)$.

Definition 4 (Combinability) *Given a concept learning problem* $\mathrm{LP} = \langle \mathcal{K}, (\mathcal{E}^+, \mathcal{E}^-) \rangle$, *a set of counter-partial definitions $\mathcal{Q}$, and an expression* C *such that* $\mathrm{cover}(\mathcal{K}, \mathrm{C}, \mathcal{E}^+) \neq \emptyset$ *and* $\mathrm{cover}(\mathcal{K}, \mathrm{C}, \mathcal{E}^-) \neq \emptyset$, C *is said to be* combinable *with $\mathcal{Q}$ iff:*

$$\mathrm{cover}(\mathcal{K}, \mathrm{C}, \mathcal{E}^-) \subseteq \mathrm{cover}(\mathcal{K}, \mathcal{Q}, \mathcal{E}^-),$$

which means that C *can be "corrected" by $\mathcal{Q}$.*

The main objective of the combination step is to check for the combinability of descriptions in the search tree. Combining definitions can be performed at several stages of the learning algorithm; this is discussed further in Section 2.3.

In this paper, we employ the refinement operator used by Lehmann and Hitzler (2010). However, the combination step helps us avoid the usage of negation in the refinement without any loss of generality, and the use of *partial models* and disjunction in the reduction steps means there is no need for disjunction in the refinement. Therefore, we use only 5 of the 7 basic rules in the original refinement operator described by Lehmann and Hitzler (2010). The two rules which refine the negation and disjunction are not used, thus our refinement operator is defined as:

Definition 5 (SPaCEL refinement operator $\rho_\sqcap$) *Given an expression* C, *a set of concept names* $\mathrm{N_C}$ *and a set of property (role) names* $\mathrm{N_R}$, *then $\rho_\sqcap$ is defined as follows:*

1. *if $C \in N_C$ (C is an atomic concept):*
   $\rho_\sqcap(C) = \{C' \mid C' \sqsubset C \text{ and } \nexists C'' \in N_C : C' \sqsubset C'' \sqsubset C\} \cup \{C \sqcap C' \mid C' \in \rho_\sqcap(\top)\}$
   (if C is an atomic concept, it is specialised by using its proper sub-concepts or creating a conjunction with the refinements of the `TOP` *concept $\rho_\sqcap(\top)$).*

2. *if* $C = \top$ *(*C *is the* TOP *concept):*
$\rho_\sqcap(C) = \{C' \mid C' \in N_C, \nexists C'' \in N_C : C' \sqsubset C'' \sqsubset C\}$
$\cup\ \{\exists r.\top \mid r \in N_R\}\ \cup\ \{\forall r.\top \mid r \in N_R\}$
(if C *is the* TOP *concept, specialise it by using its proper sub-concepts or property restrictions)*

3. *if* $C = C_1 \sqcap ... \sqcap C_n$ *(*C *is a conjunctive of descriptions):*
$\rho_\sqcap(C) = \{C_1 \sqcap ... \sqcap C' \sqcap ... \sqcap C_n \mid C' \in \rho_\sqcap(C_i), 1 \leq i \leq n\}$
(if C *is a conjunction, it is specialised by refinements of its conjuncts)*

4. *if* $C = \forall r.D, r \in N_R$*:* $\rho_\sqcap(C) = \{\forall r.D' \mid D' \in \rho_\sqcap(D)\}$
$\cup\ \{\forall r.\bot \mid D = A \in N_C$ and $\nexists A' \in N_C$ *s.t.* $A' \sqsubset A\}$
(if C *is a 'for all' property restriction, it is specialised by refinements of its range)*

5. *if* $C = \exists r.D, r \in N_R$*:* $\rho_\sqcap(C) = \{\exists r.D' \mid D' \in \rho_\sqcap(D)\}$
(this rule is for the 'there exists' property restriction, and is similar to the 4^{th} *rule.)*

Note that in the expression for $\rho_\sqcap$ in the second rule, negation and disjunction are not used, since they can be generated by *combination* and *aggregation* (steps 2 and 3 of our algorithm) respectively.

The top-down step finishes when either the partial definitions cover all positive examples or the counter-partial definitions cover all negative examples (up to allowable noise), which produces this formal definition (where the first line of this describes as the union of correct descriptions of the positive data and correctable descriptions of the same data):

Definition 6 (SPaCEL top-down learning) *Given a concept learning problem* $LP = \langle \mathcal{K}, (\mathcal{E}^+, \mathcal{E}^-)\rangle$*, the top-down learning step in this approach aims to find a set of partial definitions* $\mathcal{P}$*, a set of counter-partial definitions* $\mathcal{Q}$ *and a set of expressions* $\mathcal{X}$ *such that:*

$$cover(\mathcal{K}, \mathcal{P}, \mathcal{E}^+) \cup cover(\mathcal{K}, \mathcal{X}, \mathcal{E}^+) = \mathcal{E}^+,$$
$$\text{such that } \forall C \in \mathcal{X} \mid C \text{ is combinable with } \mathcal{Q}, \text{ or}$$
$$cover(\mathcal{K}, \mathcal{Q}, \mathcal{E}^-) = \mathcal{E}^-.$$

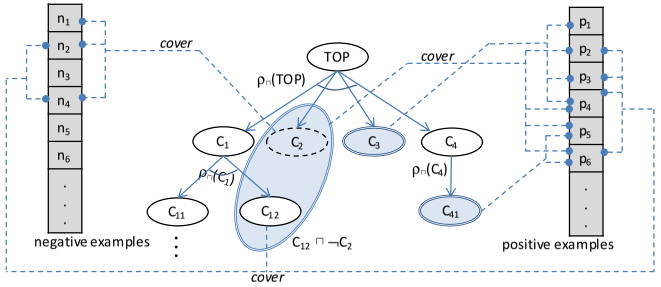
Figure 3 demonstrates the top-down step in our approach that uses the refinement operator $\rho_\sqcap$ to produce both partial and counter-partial definitions. In this example, the combination of the counter-partial definition C_2 and the expression C_{12} creates a partial definition $C_{12} \sqcap \neg C_2$.

As in the ParCEL algorithm, the selection of expressions in the search tree for refinement is controlled by node scores that are computed by a learning heuristic. The particular learning heuristic that we use is adapted from Lehmann and Hitzler (2010). However, the factors involving in the scoring function and their weights are redefined accordingly to our learning strategy:

Definition 7 (SPaCEL score) *Let* $LP = \langle \mathcal{K}, (\mathcal{E}^+, \mathcal{E}^-)\rangle$ *be a concept learning problem,* C *be a class expression and* C' *the parent expression of* C*. The score of* C *is computed as:*

$$score(C) = correctness(C, LP) + \alpha \times gain(C, LP) + \beta \times completeness(C, LP) - \gamma \times length(C, LP)$$
$$(\alpha \geq 0, \beta \geq 0, \gamma \geq 0)$$

Figure 3: The top-down learning step aims to find both partial definitions and counter-partial definitions. *Double-line* nodes are *partial definitions*, *dashed-line nodes* are *counter-partial definitions*. $\rho_\sqcap$ is the refinement operator defined in Definition 5. *Connections* from nodes to examples represent the *coverage* of the expressions.



where $gain(C, LP) = accuracy(C, LP) - accuracy(C', LP)$.

In this definition, correctness, completeness and accuracy have their standard meanings, i.e., (Lehmann and Hitzler, 2010):

$$correctness(C, LP) = \frac{|\mathcal{E}^- \backslash cover(|\mathcal{K}, C, \mathcal{E}^-)|}{|\mathcal{E}^-|},$$
$$completeness(C, LP) = \frac{|cover(\mathcal{K}, C, \mathcal{E}^+)|}{|\mathcal{E}^+|},$$
$$accuracy(C, LP) = \frac{|cover(\mathcal{K}, C, \mathcal{E}^+)| + |\mathcal{E}^- \backslash cover(\mathcal{K}, C, \mathcal{E}^-)|}{|\mathcal{E}^+ \cup \mathcal{E}^-|}.$$

The length of an expression is taken to be the sum of the number of concept names, role names, quantifiers, and connective symbols occurring in the expression, which for expressions in the $\mathcal{ALC}$ language can be defined as:

Definition 8 (Length of an $\mathcal{ALC}$ expression) *The length of an expression* D *or* E *(denoted by* $|D|$*) is defined inductively as follow (where* A *denotes an atomic concept):*

$$|A| = |\top| = |\bot| = 1$$
$$|\neg D| = |D| + 1$$
$$|D \cup E| = |D \cap E| = 1 + |D| + |E|$$
$$|\exists r.D| = |\forall r.D| = 2 + |D|$$

The learning heuristic is mainly based on the correctness of the class expression (in contrast to that of Lehmann and Hitzler (2010), which is based on accuracy). A penalty is applied for long definitions to avoid infinitely deep search (necessary because the refinement operator used in our learning algorithm is infinite). A bonus for accuracy gained by an

expression is also applied, as accurate expressions are more likely to be close to the solution. A bonus is also given for the relatively complete expressions.

Default values for the weightings of the above factors were chosen based on experimental investigations and the work by Lehmann and Hitzler (2010). We chose $\alpha = 0.2, \beta = 0.01, \gamma = 0.05$. These values can be adjusted based on the characteristics of the learning problem, such as decreasing the penalty for learning problems that need long definitions, thus biasing the search to look for deeper solutions.

The last step of our algorithm, the aggregation of the best candidates, can be performed using disjunction.

One optimisation step that can be applied to our learning is to notice *irrelevant concepts* and remove them from the search tree. Irrelevant concepts are those from which no partial or counter-partial definition can be produced, hence they cannot be expanded. An irrelevant concept in SPaCEL is defined as follows:

Definition 9 (Irrelevant concept) *Given a concept learning problem* $LP = \langle \mathcal{K}, (\mathcal{E}^+, \mathcal{E}^-) \rangle$*, a concept* C *is* irrelevant *if* $cover(\mathcal{K}, C, \mathcal{E}^+) = \emptyset$ *and* $cover(\mathcal{K}, C, \mathcal{E}^-) = \emptyset$*, i.e., it covers no positive and no negative examples.*

2.2 The algorithms

The SPaCEL learning algorithm is essentially top-down learning combined with a reduction task. The top-down step is used to solve the sub-problems of the given concept learning problem and the reduction is used to reduce and combine the sub-solutions into an overall solution. The former is performed by the downward refinement operator and a combination strategy, while the latter uses a set coverage algorithm to choose the best partial definitions and disjunction to form the overall solution.

Algorithm 1 describes the main loop of our learning algorithm, the *reducer*. It chooses the best concepts (i.e., those with highest score, based on the search heuristic described in Definition 7) from the search tree and uses the SPECIALISE algorithm (see Algorithm 2) for refinement and evaluation until the completeness of the partial definitions is sufficient. Concepts are scored using an expansion heuristic that is mainly based on the correctness of the concepts. Our reducer is the analogue of the cover removal step in ILP in that it tries to remove redundancies.

The *specialisation*, which performs the refinement and evaluation of the concepts assigned by the learning algorithm, is described in Algorithm 2. It refines the given concept ($\rho_\sqcap(C)$) and evaluates the result ($cover(\mathcal{K}, C, \mathcal{E}^+)$ and $cover(\mathcal{K}, C, \mathcal{E}^-)$). Irrelevant concepts are removed, as no partial definition or counter-partial definition can be computed from them. Then, the specialisation finds new partial definitions, counter-partial definitions and descriptions from the refinements. In practice, new descriptions are checked for redundancy before being evaluated and added to the search tree, as the same description can be generated from different branches of the search tree.

The sets of new descriptions, partial definitions and counter-partial definitions produced by the SPECIALISE algorithm are used to update the corresponding data structures and the set of covered positive and negative examples in the learning algorithm, and the new descriptions are combined with the counter-partial definitions to create new partial definitions

Algorithm 1: Symmetric Class Expression Learning Algorithm– SPACEL($\mathcal{K}, \mathcal{E}^+, \mathcal{E}^-, \varepsilon$)

Input: background knowledge $\mathcal{K}$, a set of positive $\mathcal{E}^+$ and negative $\mathcal{E}^-$ examples, and a noise value $\varepsilon \in [0, 1]$ (0 means no noise)
Output: a definition C such that $|cover(\mathcal{K}, C, \mathcal{E}^+)| \geq (|\mathcal{E}^+| \times (1 - \varepsilon))$ and $cover(\mathcal{K}, C, \mathcal{E}^-) = \emptyset$ (*empty set*)

```
1  begin
2      initialise the search tree ST = {⊤}            /* ⊤: TOP concept in DL */
3      cum_pdefs = ∅            /* set of cumulative partial definitions */
4      cum_cpdefs = ∅  /* set of cumulative counter-partial definitions */
5      cum_cp = ∅        /* set of cumulative covered positive examples */
6      cum_cn = ∅        /* set of cumulative covered negative examples */
       /* cf.  Def.  6 for the meaning of the OR operator */
7      while |cum_cp| < (|E+| × (1 − ε)) or |cum_cn| < |E−| do
8          get and remove the 'best' concept B from ST              /* see text */
9          (pdefs, cpdefs, descriptions) = SPECIALISE(B, E+, E−)      /* cf.Alg.2 */
10         cum_pdefs = cum_pdefs ∪ pdefs
11         cum_cpdefs = cum_cpdefs ∪ cpdefs
12         cum_cp = cum_cp ∪ {e | e ∈ cover(K, P, E+), P ∈ pdefs}
13         cum_cn = cum_cn ∪ {e | e ∈ cover(K, P, E−), P ∈ cpdefs}
           /* online combination, see Sec.2.3 */
14         foreach D ∈ descriptions do
               /* if D can be ''corrected'' by existing cpdefs */
15             if (cover(K, D, E−) \ cum_cn) = ∅ then
                   /* combine D with cpdefs if possible */
16                 candidates = COMBINE(D, cum_cpdefs, E−)          /* cf.Alg.3 */
17                 new_pdef = D ⊓ ¬(⊔_{A∈candidates}(A))      /* new partial def.  */
18                 cum_pdefs = cum_pdefs ∪ new_pdef
19                 cum_cp = cum_cp ∪ cover(K, new_pdef, E+)
20             else
21                 ST = ST ∪ {D}
       /* explore more partial definitions to satisfy completeness */
22     if |cum_cp| < (|E+| × (1 − ε)) then
23         foreach D ∈ ST do
24             candidates = COMBINE(D, cum_cpdefs, E−)
25             new_pdef = D ⊓ ¬(⊔_{A∈candidates}(A))
26             cum_pdefs = cum_pdefs ∪ new_pdef
27     return REDUCE(cum_pdefs)                  /* for description, see text */
```

where possible, a form of theory learning. Concepts that have been refined can be scheduled for further refinements.

The refinement operator given in Definition 5 is infinite (i.e., it can continue to expand the tree without limit), but in practice each refinement step is monotonic and bounded, as it is only allowed to generate descriptions of at most a pre-defined length (see Definition 8). For example, a concept of length N will first be refined to concepts of length $(N+1)$, and later, when it is revisited, to concepts of length $(N+2)$, etc. For the sake of simplicity, we use $\rho_\sqcap$ in the algorithms to refer to one refinement step rather than the entire refinement. This technique is used in DL-Learner and discussed in detail by Lehmann and Hitzler (2010).

When the learning algorithm reaches a sufficient degree of completeness, it stops and reduces the partial definitions to remove the redundancies using the REDUCE function, which is essentially a set coverage algorithm: given a set of partial definitions $\mathcal{X}$ and a set of positive examples $\mathcal{E}^+$, it finds a subset $\mathcal{X}' \subseteq \mathcal{X}$ such that $\mathcal{E}^+ \subseteq \bigcup_{D \in \mathcal{X}'}(cover(\mathcal{K}, D, \mathcal{E}^+))$. The solution returned by the algorithm is a disjunction of the reduced partial definitions. As an alternative, a set of partial definitions could be returned instead. This may be useful in some contexts, such as making the result more readable. The reduction algorithm may be tailored to meet particular requirements, such as the shortest definition or the smallest number of partial definitions. The combination of descriptions and counter-partial definitions that is given in Algorithm 1 is one of the combination strategies implemented in our evaluation. This strategy is called the *on-the-fly* combination strategy; it gave the best performance in our evaluation (see Section 3.2). A description of the various combination strategies we tried is given in Section 2.3.

Algorithm 2: Specialisation algorithm – SPECIALISE($C, \mathcal{K}, \mathcal{E}^+, \mathcal{E}^-$)

Input: a description C and a concept learning problem $LP = \langle \mathcal{K}, (\mathcal{E}^+, \mathcal{E}^-) \rangle$.
Output: a triple consisting of a set of partial definitions $pdefs \subseteq \rho_\sqcap(C)$; a set of counter-partial definitions $cpdefs \subseteq \rho_\sqcap(C)$; and a set of new descriptions $descriptions \subseteq \rho_\sqcap(C)$ such that
$\forall D \in descriptions : D$ is not irrelevant and $D \notin (pdefs \cup cpdefs)$, in which $\rho_\sqcap$ is the refinement operator defined in Definition 5.

begin
  $pdefs = \{D \in \rho_\sqcap(C) \mid cover(\mathcal{K}, D, \mathcal{E}^+) \neq \emptyset \ \wedge \ cover(\mathcal{K}, D, \mathcal{E}^-) = \emptyset\}$
  $cpdefs = \{D \in \rho_\sqcap(C) \mid cover(\mathcal{K}, D, \mathcal{E}^+) = \emptyset \ \wedge \ cover(\mathcal{K}, D, \mathcal{E}^-) \neq \emptyset\}$
  $descriptions = \{D \in \rho_\sqcap(C) \mid cover(\mathcal{K}, D, \mathcal{E}^+) \neq \emptyset \ \wedge \ cover(\mathcal{K}, D, \mathcal{E}^-) \neq \emptyset\}$
  return *(pdefs, cpdefs, descriptions)*

Algorithm 3 describes the *combination algorithm* that is used to combine the descriptions and counter-partial definitions to construct new partial definitions. This is basically a set coverage algorithm. One of the smallest sets of counter-partial definitions that together cover all negative examples covered by the given expression will be returned. This set of counter-partial definitions is then used to correct the given expression.

Algorithm 3: Combination algorithm – COMBINE(C, $cpdefs, \mathcal{E}^-$)

Input: a description C, a set of counter-partial definitions *cpdefs* and a set of negative examples $\mathcal{E}^-$
Output: a set $candidates \subseteq cpdefs$ such that
$cover(\mathcal{K}, C, \mathcal{E}^-) \subseteq \bigcup_{P \in candidates}(cover(\mathcal{K}, P, \mathcal{E}^-))$

begin
  $candidates = \emptyset$ `/* candidate counter-partial definitions */`
  $cn_c = cover(\mathcal{K}, C, \mathcal{E}^-)$ `/* negative examples covered by C */`
  sort *cpdefs* by descending coverage of negative examples
  while $cpdefs \neq \emptyset$ *and* $cn_c \neq \emptyset$ **do**
    get and remove the top counter-partial definition D from *cpdefs*
    if $(cover(\mathcal{K}, D, \mathcal{E}^-) \cap cn_c) \neq \emptyset$ **then**
      $candidates = candidates \cup D$
      $cn_c = cn_c \setminus cover(\mathcal{K}, D, \mathcal{E}^-))$
  if $cn_c \neq \emptyset$ **then**
    return $\emptyset$ `/* return the empty set */`
  else
    return *candidates*

COMPLETENESS OF THE ALGORITHM

The refinement operator used in our algorithm is based on the refinement operator described by Lehmann (2010). In that paper a proof of completeness over the $\mathcal{ALC}$ language is given, i.e., any concepts in the $\mathcal{ALC}$ language can be produced by this operator after a finite number of refinement steps.

Due to our learning strategy, our refinement operator does not generate negation and disjunction. As a result, our refinement operator itself is not complete. However, negation and disjunction can be created by the reduction and combination steps of the algorithm. The combination step uses negation and conjunction to correct an expression with counter-partial definitions (see Algorithm 3 and Definition 4) whereas the reduction step can generate disjunctions. Clearly, the completeness of our algorithm depends upon the combination strategy that produces the negations. Three combination strategies are proposed in Section 2.3, and our learning algorithm is complete when the *on-the-fly* and *delayed* combination strategies are used; completeness is not guaranteed for the *late* combination strategy.

2.3 Counter-partial definitions combination strategies

Counter-partial definitions are combined with expressions in the search tree to create new partial definitions where possible, producing expressions of the form $C \sqcap \neg(\sqcup_{D \in \mathcal{X}} D)$ (for expression C and a set of counter-partial definitions $\mathcal{X}$). The combinability of an expression with respect to a set of counter-partial definitions is given in Definition 4. The combination step can be performed at several stages of the learning algorithm, and different choices have different effects on the learning result. We describe three combination strategies. An evaluation of these strategies is given in Section 3.2.

2.3.1 Late combination

In this strategy, the algorithm maintains sets of partial definitions and counter-partial definitions separately. When all positive or negative examples are covered (or another termination condition is reached, such as maximum time allowed, or a sufficient percentage of examples covered), it will combine descriptions from the search tree and the set of counter-partial definitions.

Since the combination is performed after the learning stops, this strategy may provide a better combination, i.e., it may have better choices of counter-partial definitions to use for the combination. However, for concept learning problems in which both positive and negative examples use negation, the algorithm may not be able to find the definition because the refinement operator designed for this algorithm does not use negation. If this is the case and a maximum time timeout is set, then the combination will be made when the timeout is reached in order to find the definition. Otherwise, the algorithm will not terminate.

For example, the concept learning problem shown in Figure 2 may cause this strategy to run out of memory without finding an accurate concept if there is no timeout or no noise is allowed. To cover all positive examples, we need a class expression with negation as follows:

$$\text{Bat} \sqcup (\text{Bird} \sqcap \neg\text{Penguin})$$

and to completely cover all negative examples, the following class expression is needed:

$$\text{Penguin} \sqcup (\text{Mammal} \sqcap \neg\text{Bat})$$

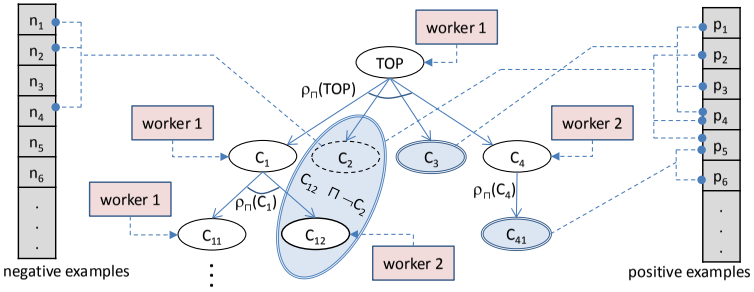
To produce the definition for positive examples, the counter-partial definition `Penguin` needs to be combined with the expression `Bird`, which cannot be performed directly due to the removal of negation from the refinement operator. However, in this strategy, the combination is only called when all positive examples or negative examples are covered, and in the example this condition is never met, as the expression $\neg$`Bat` cannot be produced.

2.3.2 On-the-fly combination

This strategy is used in Algorithm 1. When a new description is generated from the refinement or an existing description is revisited, it is combined with the existing counter-partial definitions if possible. This strategy can avoid the termination problem discussed in the *late combination* strategy because negation is used in the combination. The evaluation suggests that this strategy, overall, gives the best performance and the smallest search tree. However, the final results can be optimised since a counter-partial definition may be combined with many expressions and thus the final definition is unnecessarily long. For example, in the concept learning problem described in Figure 2, when the description `Bird` is generated, there is no counter-partial definition to combine it with. However, when it is revisited for refinement, it will be combined with the new counter-partial definition `Penguin` to produce a partial definition `Bird` $\sqcap$ $\neg$`Penguin`.

This strategy seems to produce similar results to the original refinement by Lehmann and Hitzler (2010), which can generate negation. However, they are essentially different. In our approach, the negation is only used for counter-partial definitions to remove negative examples from the description in the search tree, while negation may be applied for any description in the original refinement operator. In addition, in our approach, the counter-

Figure 4: Top-down learning in SPaCEL with multiple workers.



partial-definitions are stored to be reused to "correct" the new descriptions in the search tree, while the original refinement may regenerate the same negated description.

2.3.3 Delayed combination

This strategy lies inbetween the previous two, and checks the possibility of combinations when a new description is generated. However, even if the combination is possible, only the set of cumulative covered positive examples is updated (by removing the elements that are covered by the new description), while the new description is put into a *potential partial definitions* set. The combination is executed when the termination condition is reached, i.e., either all positive examples or all negative examples are covered, or the algorithms times out.

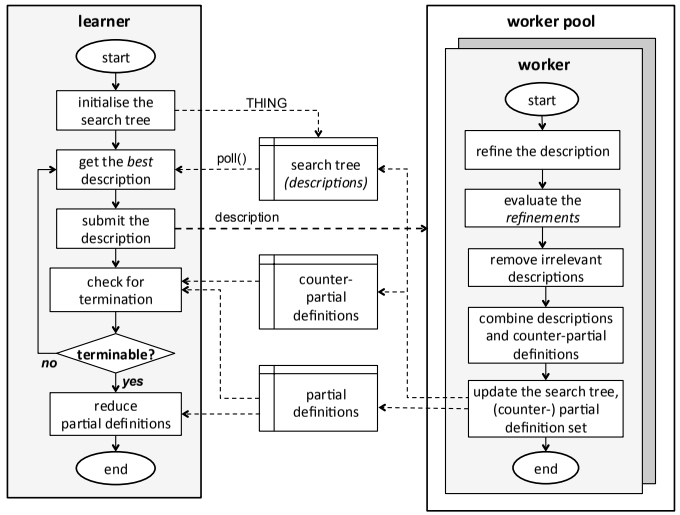
This strategy may help to prevent the problem of the late combination strategy, and it may return better combinations in comparison with the *on-the-fly* strategy because it inherits the advantage of the *late combination* strategy. However, as the combinable descriptions in this strategy are not combined until the learning terminates, the search tree is likely to be larger than that of the on-the-fly strategy.

2.4 Parallelisation of the learning algorithm

One benefit of the symmetric approach is that it is inherently parallel: several expressions (branches) in the search tree can be processed (refined and evaluated) in parallel using multiple *workers* to find the partial definitions. A central *reducer*, which is part of the learner, controls the management of partial and counter-partial definitions produced by the workers until the termination condition described in Definition 6 is met. Then it performs the reduction and aggregation steps to create a final definition. The parallel exploration of the search tree by multiple workers is illustrated in Figure 4.

Our organisation of learning tasks described above follows the idea of the map-reduce architecture (Dean and Ghemawat, 2008), which enables the *parallel divide and conquer* strategy. Here, we organise our algorithm architecture into two parts as shown in Figure 5. The computationally heavy parts, including refinement and evaluation of concepts, are

Figure 5: Implementation architecture. Note that the combination step may be performed by the workers or the reducer, depending upon the combination strategy (see Section 2.3).



done by the multiple workers. The combining of descriptions and counter-partial definitions is also performed on the worker side.

3. Evaluation

In order to evaluate the Symmetric Class Expression Learning approach of SPaCEL, we compared it to CELOE and ParCEL using 10-fold cross validation with 15 data sets. These are summarised in Section 3.1, using the search tree size, predictive accuracy and learning time, and the length of the definitions learnt as the metrics for comparison. CELOE is one of the description logic learning algorithms in the DL-Learner framework. It is well-evaluated and has been compared with many other learning algorithms. Therefore, using it for comparison enables an implicit comparison with other learning algorithms; the evaluations by Hellmann (2008); Lehmann (2010) suggest that, overall, CELOE produced better results than the comparators. The implementation of CELOE that we used is publicly available at `https://github.com/tcanvn/SParCEL/`.

Before performing the comparison, we used the same evaluation method to compare the three evaluation strategies described in Section 2.3, meaning that only one of them (the on-the-fly strategy) was used in the comparison experiments.

3.1 Experimental data sets

To ensure the evaluation covers as much as possible of the space of cases, which helps to archive a thorough assessment of the learning algorithms, we chose data sets that vary in: i) size of knowledge base and examples, ii) the length of definitions, and iii) the amount of noise. In some cases, (i) was achieved by using subsets of a particular data set.

The data sets are divided into three groups. The first group includes 7 learning problems that have low to medium complexity and on which all algorithms in the experiment can find solutions on the training set without timing out in the 10 minutes we allowed. The second group contains high to very high complexity learning problems; all algorithms in the experiment can also find accurate solutions on the training set without timing out. The third contains leaning problems for which one of the learning algorithms could not find accurate solution on the training set (i.e. timeout occurred). The complexity of the learning problems is approximately estimated based on the definition length produced by CELOE: i) low: (0, 8]; ii) medium: (8, 15]; iii) high: (15, 20]; and very high: longer than 20. The reason for treating problems learnt with and without timeout separately is that some metrics cannot be compared if the learning algorithm cannot find the solution (e.g. learning time, search space size).

The list of data sets used and their properties are summarised in Table 1, and an overview of them is given next. We use Manchester OWL Syntax (Horridge and Patel-Schneider, 2009) to describe the learning problems' expected definitions.

Moral This data set was first introduced by Wogulis (1994) and is available at the University of California Irvine (UCI) machine learning repository[2] (Frank and Asuncion, 2010). It is intended to be used to learn definitions of harm-doing activities and contains concepts and observations related to the classification of activities as 'guilty' and 'not guilty' through a set of sub-categories such as 'blameworthy' and 'justified'.

Poker A description of 10 classes of poker hands (5 cards drawn from a standard deck) based on 10 predictive attributes (suit and rank of each card). This data set can be customised into binary-class learning and is available at the UCI repository[3].

Family A set of data sets containing observations about family relations such as aunt, brother, cousin, daughter, father, grandson and uncle (Richards and Mooney, 1995). This data set can be found in many repositories. In our evaluation, we use the data set distributed with the DL-Learner package (which can be found in the DL-Learner repository[4]). Each family relation can be treated as a separate learning problem, and we use several of them. We also use the `Forte Uncle` problem used by Lehmann (2007) and available in the DL-Leaner repository.

CarcinoGenesis This data set is used to learn the structure of chemical compounds and bioassays that may cause cancer based on data from the US National Toxicology Program. It was transformed into logic programming and used for evaluation of ILP algorithms by many authors such as Bahler (1993); Srinivasan et al. (1997) and was

2. http://archive.ics.uci.edu/ml/datasets/Moral+Reasoner
3. http://archive.ics.uci.edu/ml/datasets/Poker+Hand
4. https://github.com/AKSW/DL-Learner

transformed into the OWL ontology format and used to evaluate the description logic learning algorithms in the DL-Learner framework (Lehmann, 2010)[5]. It is particularly noisy, which makes it a challenging problem in machine learning for both symbolic and sub-symbolic learning approaches.

MUSE Dataset We employ use case 1 of a set of context-aware smart home simulations by Lyons et al. (2010) for normal and abnormal living behaviours. The data set is available in the SPaCEL repository[6].

ILPD Dataset A data set based on the liver function test records of patients collected from the North East of Andhra Pradesh, India (Indian Liver Patient Dataset). Each record comprises patient information such as age and gender and results of liver function tests including total Bilirubin, Albumin, and total protein, and the target is an expert-assigned label of cancerous or non-cancerous. This data set is also available in the UCI repository[7].

MUBus Dataset A data set developed to demonstrate the SPaCEL algorithm, it is based on a bus timetable that varies according to conditions such as weekday/weekend, university holidays, and school holidays. Data was generated by sampling at 5 minute intervals. Sub-datasets can be created by restricting the sampling times, such as "MUBus12 [07:00 - 09:20]" (shortened as MUBus-1), "MUBus12 [07:00 - 12:00]" (shortened as MUBus-2) and "MUBus12 [07:00 - 21:30]" (shortened as MUBus-3). This data set is available in the SPaCEL repository; for more information about it, see (Tran, 2013).

The timetable makes surprising complicated rules, such as the following description for a weekend bus:

```
(NOT Holiday1) AND ((hasMinute VALUE 0 and hasHour VALUE 9) OR
  (hasMinute VALUE 20 AND
    (hasHour VALUE 10 OR hasHour VALUE 14 OR hasHour VALUE 16)) OR
  (hasMinute VALUE 40 AND
    (hasHour VALUE 11 OR hasHour VALUE 13 OR hasHour VALUE 15)))
```

3.2 Comparison of Combination Strategies

As was described in Section 2.3, three combination strategies were implemented: *late*, *delayed* and *on-the-fly*. We chose to compare these strategies in order to choose one as the most useful for SPaCEL. Table 2 shows the experimental results. It can be seen that the accuracies achieved are similar across all three, but the learning times and search space sizes differ. For example, the late combination strategy did not terminate in some cases: for the MUBus-2 data set the learner was interrupted by timeout after 10 minutes. However, by applying the algorithm after the learner was interrupted, a definition with 100% accuracy on the training data set was produced. This means the solution existed implicitly, but the

5. http://svn.code.sf.net/p/dl-learner/code/trunk/examples/carcinogenesis/
6. https://github.com/tcanvn/SParCEL/
7. http://archive.ics.uci.edu/ml/datasets/ILPD+(Indian+Liver+Patient+Dataset)

Table 1: Properties of the evaluation data sets. OPs and DPs are short for Object Properties and Data Properties respectively. Number of examples is given in the form positive/negative examples.

No	Dataset	Classes	OPs	DPs	Assertions			Examples
					Class	OP	DP	
1.	Moral	43	0	0	4646	0	0	102/100
2.	Forte uncle	3	3	0	86	251	0	23/163
3.	Poker	4	6	0	374	1080	0	4/151
4.	Carcino-Genesis	142	4	15	22,372	40,666	11,185	182/155
5.	ILPD	4	0	10	976	0	1952	323/165
6.	UCA1 (MUSE)	30	4	11	300	200	200	73/77
	Family data set	4	4	0	606	728	0	-
7.	Aunt	-	-	-	-	-	-	41/41
8.	Brother	-	-	-	-	-	-	43/30
9.	Cousin	-	-	-	-	-	-	71/71
10.	Daughter	-	-	-	-	-	-	52/52
11.	Father	-	-	-	-	-	-	60/60
12.	Grandson	-	-	-	-	-	-	30/30
13.	Uncle	-	-	-	-	-	-	38/38
	MUBus12 Dataset							
14.	[07:00-09:10]	25	0	12	2675	0	32110	383/2292
15.	[07:00-12:00]	25	0	12	6314	0	75766	670/5643
16.	[07:00-21:30]	25	0	12	17128	0	205534	1250/15877

learner was not able to compute it. To make sure that the learner was not terminated too early, the experiment was rerun for 3 hours, but the learner was still not able to find the solution on the training data set. This demonstrates the disadvantage of this strategy.

Table 2: Combination strategies experimental result (*means ± standard deviations of 10 folds*).

Metric	**Late**		**Delayed**		**On-the-fly**	
UCA1						
Learning time (s)	1.01	± 0.33	0.54	± 0.23	0.72	± 0.17
Accuracy (%)	100.00	± 0	100.00	± 0	100.00	± 0
Definition length	20.20	± 0.63	58.60	± 21.31	66.60	± 1.90
No of descriptions	11,137.30	± 2,711.99	5,065.30	± 1,867.64	6,998.70	± 859.74
No of pdef.[8]	1.00	± 0.00	3.40	± 1.27	4.00	± 0.00
Avg.[9] pdef. length	20.20	± 0.63	17.65	± 2.02	16.65	± 0.47
MUBus1						
Learning time (s)	47.59	± 17.66	23.02	± 28.33	7.52	± 2.37
Accuracy (%)	100.00	± 0	99.81	± 0.36	99.74	± 0.36
Definition length	297.30	± 22.67	383.00	± 170.47	393.30	± 73.51
No of descriptions	39,096.80	± 12,210.86	14,463.90	± 20,232.28	2,130.10	± 1,179.48
No of pdef.	1.00	± 0.00	4.20	± 1.69	6.30	± 1.25
Avg. pdef. length	297.30	± 22.67	145.32	± 155.66	63.40	± 10.54
MUBus2						
Learning time (s)	int.[10] @600s		90.97	± 73.43	48.46	± 16.56
Accuracy (%)	99.89	± 0.15	99.84	± 0.20	99.78	± 0.23
Definition length	1,419.20	± 328.20	1,372.30	± 539.13	1,179.50	± 209.80
No of descriptions	187,279.60	± 2,584.59	23,843.40	± 21,430.22	8,575.10	± 4,184.27
No of pdef.	8.00	± 0	8.60	± 0.10	12.00	± 2.79
Avg. pdef. length	177.40	± 41.03	161.52	± 67.40	101.29	± 45.37
MUBus3						
Learning time (s)	int. @900s		566.89	± 224.52	495.94	± 267.67
Accuracy (%)	99.83	± 0.10	97.96	± 1.90	99.72	± 0.40
Definition length	4,120.10	± 1,045.59	3,740.50	± 1,228.56	3,728.00	± 1,598.00
No of descriptions	141,243.00	± 2,185	49,980.00	± 21,778	23,477.00	± 13,897
No of pdef.	18.20	± 0.79	15.70	± 2.83	18.10	± 1.85
Avg. pdef. length	225.38	± 51.33	249.16	± 100.58	205.90	± 85.49

In addition, as the late combination strategy performed the combination at the end, only after all positive or negative examples are covered, its learning times were always longer than other strategies. In theory this might produce more concise solutions than the on-the-fly strategy, as can be seen in the following partial definitions produced by the on-the-fly strategy with the UCA1 data set:

1. activityHasDuration SOME hasDurationValue ≤ 15.5 AND
   (NOT (Activity AND activityHasDuration SOME hasDurationValue ≤ 4.5))

8. Partial definitions
9. Average
10. Interrupted

2. activityHasStarttime SOME Autumn AND
   (NOT (Activity AND activityHasDuration SOME hasDurationValue ≤ 4.5)) AND
   (NOT (Activity AND activityHasDuration SOME hasDurationValue ≥ 19.5))

3. activityHasStarttime SOME Summer AND
   (NOT (Activity AND activityHasDuration SOME hasDurationValue ≤ 4.5)) AND
   (NOT (Activity AND activityHasDuration SOME hasDurationValue ≥ 19.5))

4. activityHasDuration SOME hasDurationValue ≥ 4.5 AND
   activityHasStarttime SOME Spring AND
   (NOT (Activity AND activityHasDuration SOME hasDurationValue ≥ 19.5))

In contrast, the delayed strategy found:

activityHasDuration SOME (hasDurationValue ≥ 4.5 AND
hasDurationValue ≤ 19.5) AND
(NOT (activityHasStarttime SOME Winter AND
activityHasDuration SOME hasDurationValue ≥ 15.5))

The difference is between a large number of short expressions and a smaller number of long ones; usually the delayed strategy produced shorter descriptions overall. However, the on-the-fly strategy dominated in most aspects, particularly the learning time and search space covered. Our hope that the delayed strategy would combine the advantages of both of the others was not bourne out, and we therefore chose to use the on-the-fly strategy for the experimental comparison of SPaCEL with other algorithms.

3.3 Comparison Experiment

We measured search tree size (Table 3), predictive accuracy (Table 4), learning time (also Table 4), and definition length (Table 6) for all 15 data sets using CELOE, PaRCEL and SPaCEL. This section reports and discusses the results of this experiment.

The search tree size reported in this experiment is the total number of all descriptions that are inserted into the search tree, including the irrelevant descriptions (the partial and counter-partial definitions which are removed later by the algorithm). If an algorithm can find the solution, the result reported is the search tree size after the learning algorithm has terminated. Otherwise, it is the search tree size at the time when the timeout has occurred. In this case, the comparison should be treated with caution as it depends upon the timeout assigned for the learning algorithm. Only learning problems on which at least one of the algorithms found an accurate definition on the training set were considered. Therefore, the results for the CarcinoGenesis and ILDP learning problems are not reported, since all three algorithms timed out.

In the first group of learning problems, SPaCEL produced smaller search trees than both CELOE and ParCEL for 4 out of 7 learning problems. However, the definitions of these learning problems were short and therefore the search tree sizes were usually small. Consequently, they were sensitive to the parallelisation: as the search tree in our algorithm is expanded by multiple workers, the solution may be found by one of the workers while the other workers are still processing the search tree. The search tree may be therefore

Table 3: Search tree size (*means ± standard deviations of 10 folds*). The underlined values are the search tree size after the *timeout* has occurred. The statistical significance was tested using t-tests: bold values are statistically significantly higher (at the 95% confidence level) than other values; unformatted values are statistically significantly lower than other values; bold italic values are statistically significantly lower than the bold values, and statistically significantly higher than the unformatted values.

Problem	CELOE		ParCEL		SPaCEL	
Low to medium complexity learning problems without timeout						
Moral	540.5	± 16.9	**33.3**	± **11.0**	**223.4**	± **364.2**
Forte	64,707.9	± 33,641.9	***859.5***	± ***251.2***	**174.1**	± **108.2**
PokerStraight	**1,090.8**	± **12.5**	14,204.8	± 2,847	***2,105.0***	± ***1,217.7***
Brother	***37.0***	± ***0***	104.4	± 92.6	**18.4**	± **9.1**
Daughter	***21.0***	± ***0***	111.3	± 110.2	**18.1**	± **5.9**
Father	***29.0***	± ***0***	82.9	± 63.8	**23.0**	± **7.0**
Grandson	**80.5**	± **3.0**	1,867.5	± 1,519.3	***125.3***	± ***25.3***
High to very high complexity learning problems without timeout						
Aunt	85,883.8	± 67,328.8	***7,023.4***	± ***2,912.1***	**2,127.8**	± **1,160.9**
Cousin	***20,331.6***	± ***233.6***	35,484.4	± 39,055.4	**6,761.1**	± **722.9**
Uncle	541,081.8	± 0	***6,332.5***	± ***3,247.9***	**2,400.0**	± **551.2**
Learning problems with at least 1 timeout						
UCA1	1,465,263.2	± 11,515.5	28,676.0	± 14,935.2	**6,998.70**	± **859.7**
MUBus-1	161,832.2	± 3,054.5	643,401.5	± 139,303	**2,130.1**	± **1,179.5**
MUBus-2	79.959.2	± 118.5	879,866.1	± 34,000.4	**8,575.1**	± **4,184.3**
MUBus-3	55,130.4	± 73.7	453,605.1	± 7,565.3	**23,477.0**	± **13,897**

expanded redundantly and thus the reported search tree sizes might be unnecessarily larger than the minimal search tree size required to learn the problem.

In the second and the third groups, SPaCEL always produced the smallest search trees for all learning problems. The search tree sizes generated by the three algorithms for the same learning problem in this group were extremely different. For example, SPaCEL only needed to explore about 2,400 expressions to find the solution for the Uncle learning problem while CELOE had to explore more than 541,081 expressions. Similarly, SPaCEL found a solution after exploring about 2,130 expressions, while ParCEL needed to explore 643,401 expressions on the MUBus-1 learning problem. CELOE could not find an accurate solution for this learning problem and timed out after 10 minutes. The average search tree size at the time of timeout was about 161,832 expressions.

A t-test rejected the null hypothesis for similarity of search tree size of the algorithms on all learning problems at the 99% confidence level, meaning that the search trees generated by SPaCEL were statistically significantly smaller than CELOE in 12 out of 14 learning problems and than ParCEL for 13 out of 14 learning problems.

With regard to predictive accuracy and learning time, which are reported in Table 4, in general, SPaCEL achieved better predictive accuracy in most learning problems. In the first two groups (10 learning problems) all three learning algorithms achieved very high accuracy, including 100% accuracy in 5 of them. In the remaining 5 learning problems, SPaCEL was statistically significantly more accurate than CELOE for 3 out of 5 and ParCEL for 1 out of 5 problems. There were no learning problems in this group where SPaCEL was statistically significantly less accurate than either of the other algorithms.

Table 4: Learning time and predictive accuracy experimental results summary (*means ± standard deviations of 10 folds*). Result of the statistical significance t-test (at the 95% confidence level) is also included: the bold and highlighted values are statistically significantly *better* than other values; the unformatted values are statistically significantly *worse* than other values; the bold and italic and highlighted values are statistically significantly *worse* than the bold and highlighted values, and statistically significantly *better* than the unformatted values; the underlined values represent the values that are not statistically significantly different from the other values.

Problem	Predictive accuracy (%)			Learning time (s)		
	CELOE	ParCEL	SPaCEL	CELOE	ParCEL	SPaCEL
Low to medium complexity learning problems without timeout.						
Moral	100.00 ±0.00	100.00 ±0.00	100.00 ±0.00	0.15 ±0.03	**0.02** ±**0.01**	**0.03** ±**0.02**
Forte	98.86 ±2.27	100.00 ±0.00	100.00 ±0.00	2.60 ±1.64	***0.23*** ±***0.17***	**0.05** ±**0.02**
Poker-Straight	100.00 ±0.00	96.43 ±4.12	98.21 ±3.57	0.36 ±0.71	0.59 ±0.08	**0.32** ±**0.18**
Brother	100.00 ±0.00	100.00 ±0.00	100.00 ±0.00	0.19 ±0.16	***0.03*** ±***0.02***	**0.02** ±**0.01**
Daughter	100.00 ±0.00	100.00 ±0.00	100.00 ±0.00	0.2 ±0.02	**0.03** ±**0.03**	**0.02** ±**0.01**

Continued on next page

Table 4 – continued

Problem	Predictive accuracy (%)			Learning time (s)		
	CELOE	ParCEL	SPaCEL	CELOE	ParCEL	SPaCEL
Father	100.00 ±0.00	100.00 ±0.00	100.00 ±0.00	0.02 ±0.10	0.03 ±0.03	0.02 ±0.01
Grandson	100.00 ±0.00	100.00 ±0.00	100.00 ±0.00	0.08 ±0.07	0.19 ±0.79	0.05 ±0.02
High to very high complexity learning problems without timeout.						
Aunt	96.5 ±0.00	**100.00** ±0.00	**100.00** ±0.00	30.01 ±0.02	**0.26** ±0.15	**0.22** ±0.15
Cousin	99.29 ±0.00	99.29 ±2.26	**100.00** ±0.00	3.79 ±0.54	**0.54** ±0.20	***0.80*** ±*0.28*
Uncle	95.83 ±6.80	98.75 ±3.95	95.42 ±10.84	34.13 ±14.94	***0.29*** ±*0.18*	**0.16** ±0.11
Learning problems with timeout.						
CarcinoGenesis	53.73 ±4.79	56.05 ±4.30	**60.52** ±6.06	int.* @2000s	int.* @2000s	int.* @2000s
UCA1	91.42 ±7.01	**100.00** ±0.00	**100.00** ±0.00	int.* @2000s	***29.75*** ±*5.77*	**0.72** ±0.17
MUBus-1	53.61 ±2.45	**99.63** ±0.31	**99.74** ± 0.36	int.* @600s	***395.33*** ±*11.83*	**7.52** ±2.37
MUBus-2	14.35 ±1.10	***97.91*** ±*0.50*	**99.78** ±0.23	int.* @600s	int.* @600s	**48.46** ±16.56
MUBus-3	11.34 ±0.06	***95.85*** ±*0.31*	**99.72** ±0.40	int.* @900s	int.* @900s	**495.94** ±267.67
ILPD	**76.02** ±2.61	71.12 ±5.36	72.67 ±8.12	int.* @120s	int.* @120s	int.* @120s

Note: *: *Interrupted by timeout*

In the last group of problems SPaCEL outperformed CELOE on 5 out of 6 and ParCEL on 3 out of 6 learning problems. The data set MU-Bus is a complex learning problem in which the target definition is very long, as the bus operation time depends upon many conditions. For this data set, SPaCEL outperformed both ParCEL and CELOE. It always found the complete definition on the training set and the accuracy on the test set was always over 99.7%, while CELOE could not find accurate definitions on the training set and the accuracy on the test set was very low, from 11.34% to 53.61%. ParCEL performed better than CELOE and the accuracy was also very high but it was still statistically significantly less accurate than SPaCEL.

For the ILPD data set it appears that CELOE had higher predictive accuracy that SPaCEL, although the difference was not statistically significant. However, as the ILPD and MuBus data sets are unbalanced, it is more appropriate to use balanced accuracy [11] for these learning problems. Table 5 shows the balanced predictive accuracy for ILPD and some other unbalanced learning problems. The balanced accuracy of SPaCEL and ParCEL on the ILDP learning problem were statistically significantly higher than CELOE. The outcome of the statistical significance test on the balanced accuracy of other learning problems did not change.

11. *balanced accuracy* $= \frac{1}{2} \cdot \left(\frac{|true\ positive|}{|positive\ examples|} + \frac{|true\ negative|}{|negative\ examples|} \right)$

Table 5: Balanced predictive accuracy of unbalanced data sets in Table 4. Conventions of the results' representation are similar to that of in Table 4.

Problem	Balanced predictive accuracy (%)					
	CELOE		ParCEL		SPaCEL	
MUBus-1	71.12	± 0.65	**99.68**	± 0.60	**99.74**	± 0.62
MUBus-2	52.09	± 0.06	***91.86***	± *3.02*	**99.78**	± 0.59
MUBus-3	52.18	± 0.03	***73.07***	± *1.91*	**99.02**	± 2.67
ILDP	64.62	± 4.83	**70.94**	± 10.87	**71.73**	± 12.29

Our symmetric approach to class expression learning not only increased the predictive accuracy, but also decreased the learning time, although this effect can only be observed for the medium and high complexity data sets. The t-test result on learning time shows that SPaCEL was statistically significantly faster than CELOE in 13 out of 14 problems and than ParCEL for 10 out of 14 learning problems. It was statistically significantly slower than ParCEL for 2 out of 14 problems, both in the first and second groups. The definition lengths in this group were short and the learning times were very small.

The last metric that we evaluated was the length of the definitions learnt; the results can be seen in Table 6. The definition lengths of ParCEL and SPaCEL are reported by the number of partial definitions and the average length of the partial definitions, the total average length of their definitions is the product of those numbers.

Table 6: Definition length of the learning problems (*means* ± *standard deviations of 10 folds*). The bold values are the definition lengths after the timeout occurred.

Problem	(Partial) definition length			No of partial definitions	
	CELOE	ParCEL	SPaCEL	ParCEL	SPaCEL
Low to medium complexity learning problems without timeout					
Moral	3.00 ±0.00	1.52 ±0.05	1.50 ±0.00	2.10 ±0.32	2.00 ±0.00
Forte	13.50 ±1.00	7.75 ±0.50	8.50 ±0.00	2.00 ±0.00	2.00 ±0.00
PokerStraight	11.70 ±0.68	10.90 ±1.31	19.75 ±2.50	1.70 ±0.68	1.00 ±0.00
Brother	6.00 ±0.00	6.00 ±1.83	6.40 ±0.84	1.00 ±0.00	1.00 ±0.00
Daughter	5.00 ±0.00	5.25 ±1.09	6.20 ±0.63	1.10 ±0.32	1.00 ±0.00
Father	5.00 ±0.00	5.50 ±0.53	5.90 ±0.98	1.00 ±0.00	1.00 ±1.00
Grandson	7.25 ±0.50	7.25 ±0.50	8.00 ±0.00	1.00 ±0.00	1.00 ±0.00
High to very high complexity learning problems without timeout					
Aunt	19.00 ±0.00	8.80 ±0.48	10.10 ±0.97	2.00 ±0.00	2.00 ±0.00
Cousin	23.40 ±2.59	8.50 ±0.00	8.50 ±0.00	2.00 ±0.00	2.00 ±0.00

Continued on next page

Table 6 – continued

Problem	(Partial) definition length			No of partial definitions	
	CELOE	**ParCEL**	**SPaCEL**	**ParCEL**	**SPaCEL**
Uncle	19.00 ±14.94	8.40 ±0.38	10.15 ±1.67	2.00 ±0.00	2.00 ±0.00
Learning problems with timeout					
CarcinoGenesis	4.80 ±0.42	55.87 ±9.52	138.00 ±51.46	72.70 ±3.43	17.00 ±5.74
UCA1	**9.00** ±0.00	12.75 ±0.00	16.65 ±0.47	4.00 ±0.00	4.00 ±0.00
MUBus-1	**12.70** ±0.48	16.99 ±0.42	63.40 ±10.54	15.00 ±1.56	6.30 ±1.25
MUBus-2	**2.00** ±0.00	**16.07** ±0.19	101.29 ±45.37	24.80 ±3.52	12.00 ±2.79
MUBus-3	**2.00** ±0.00	**14.64** ±0.13	**205.90** ±85.49	25.40 ±1.58	18.10 ±1.85
ILPD	**5.80** ±1.69	**8.34** ±0.12	**13.30** ±1.40	42.80 ±1.69	37.00 ±2.06

The experimental results show that ParCEL and SPaCEL produced longer definitions than CELOE for most learning problems. By manually inspecting the results we identified that this was because partial definitions tend to overlap, and that the creation of sub-solutions leads to more specific definitions. It may also be that since CELOE timed out more often, it had not yet managed to produce completely accurate (and necessarily longer) solutions before the learning was terminated.

For the learning problems in the first two groups, where all three algorithms can find the definition for the training sets without timeout, the differences in definition length between the three algorithms were small, and any difference was caused by the overlap between partial definitions. In some cases, this can be shortened using an optimisation strategy. For example, the definitions produced by CELOE and SPaCEL for the Forte Uncle data set are:

- **CELOE:**

  male AND ((married SOME sibling SOME Person) OR
  (married SOME hasSibling SOME hasChild SOME Thing))

- **SPaCEL** (two partial definitions):

  1. hasSibling SOME hasChild SOME Thing AND (NOT female)

  2. married SOME hasSibling SOME hasChild SOME Thing AND (NOT female)

The length of the definition produced by CELOE is 15 and SPaCEL is 19 (length of two partial definitions plus 1 for disjunction). However, at least 3 axioms in the SPaCEL final definition can be reduced by removing the common part among partial definitions, i.e. `AND NOT female`. That means that if the same normal form (Lloyd, 1984; Gabbay

et al., 1998) is applied for both CELOE and SPaCEL, the difference between their lengths can be reduced. Moreover, `NOT female` can be replaced by `Male` if `male` and `female` are declared as disjoint and jointly exhaustive properties. This is the idea of *optimisation* and *simplification* in description logic (Horrocks and Patel-Schneider, 1999; Baader et al., 2010). Currently, this idea has not yet been implemented in our algorithm. However, rather than applying optimisation and simplication to the definition constructed by combining multiple partial definitions using conjunction, breaking down a long definition into several smaller partial definitions (as in SPaCEL) may help to make the definition more readable, particularly when the definition is long.

For the learning problems on which at least one of the learning algorithms cannot find an accurate definition on the training set, i.e. timeout occurred in the experiments, definitions produced by SPaCEL are significant longer than those of CELOE. An interesting comparison is seen in the definitions produced for the UCA1 data set, which is complex, but noiseless:

- **CELOE**:

  activityHasDuration SOME (hasDurationValue $\geq$ 4.5 AND
  hasDurationValue $\leq$ 21.5)

- **ParCEL**:

  1. activityHasDuration SOME
     (hasDurationValue $\geq$ 4.5 AND hasDurationValue $\leq$ 15.5)

  2. activityHasDuration SOME
     (hasDurationValue $\geq$ 15.5 AND hasDurationValue $\leq$ 19.5) AND
     activityHasStarttime SOME Spring

  3. activityHasDuration SOME
     (hasDurationValue $\geq$ 15.5 AND hasDurationValue $\leq$ 19.5) AND
     activityHasStarttime SOME Summer

  4. activityHasStarttime SOME Autumn AND activityHasDuration SOME
     (hasDurationValue >= 4.5 AND hasDurationValue <= 19.5)

- **SPaCEL** (late combination):

  activityHasDuration SOME (hasDurationValue $\geq$ 4.5 AND
  hasDurationValue $\leq$ 19.5) AND (NOT (activityHasDuration SOME
  hasDurationValue $\geq$ 15.5 AND activityHasStarttime SOME Winter))

Obviously, the short definition (length 9) produced by CELOE does not fully define the positive examples (both training accuracy and predictive accuracy were below 100%). Meanwhile, ParCEL produced a far longer definition (length 51) that describes the positive examples accurately (both training and predictive accuracy were 100%). The definition produced by SPaCEL for this data set better demonstrates the idea of using the symmetric learning approach and exceptions in learning, since it is a combination of an expression:

activityHasDuration SOME
(hasDurationValue $\geq$ 4.5 AND hasDurationValue $\leq$ 19.5)

and a counter-partial definition:

activityHasDuration SOME
(hasDurationValue $\geq$ 15.5 AND activityHasStarttime SOME Winter)

This definition is shorter than the definition produced by ParCEL and it still describes the positive examples accurately.

4. Related Work

4.1 Description logic learning

Learning in description logic is basically a search problem in which the search tree is often dynamically generated by a refinement operator. In contrast to other algorithms, SPaCEL uses a top-down approach to learning both positive and negative patterns. Only YinYang (Iannone et al., 2007) has particularly considered negative patterns previously, and they took a bottom-up approach to them, despite learning top-down for the positive patterns. The bottom-up approach creates concepts by joining most specific concepts created for positive examples using disjunction, which can create large concept definitions that are not truly intentional.

Top-down learning has been used by several algorithms reported in the literature, such as Badea and Nienhuys-Cheng (2000), which uses a refinement operator designed for the $\mathcal{ALER}$ description logic. However, as discussed by Lehmann and Hitzler (2010), this refinement operator cannot be extended to handle more expressive description logic language such as $\mathcal{SRIOQ}(\mathcal{D})$, which is the language on which OWL2 is based.

Lisi and Malerba (2003) proposed an alternative based on the hybrid $\mathcal{AL}$-log language, which combines the $\mathcal{ALC}$ description logic language and Datalog. In Nienhuys-Cheng and De Wolf (1997) an ideal downward refinement operator for the $\mathcal{AL}$-log language was proposed that is based on the notation of $\mathcal{B}$-subsumption. This is a generalisation of the generalised subsumption in the $\mathcal{AL}$-log language and is suitable for the hybrid knowledge representation systems that are constituted by the relational and structural subsystems. It was implemented in the $\mathcal{AL}$-QuIn ($\mathcal{AL}$-log Query Induction) system.

In Esposito et al. (2004); Iannone et al. (2007), a top-down refinement operator for the $\mathcal{ALC}$ language is used in combination with a bottom-up most specific concept (MSC) operator. The essential idea of these studies lies in the concept of *counterfactuals*, that is *errors* within candidate hypotheses. Therefore, for each candidate hypothesis, the learning algorithm uses an MSC operator to find the concept(s) representing errors and remove them from the candidate hypothesis. However, this approach has two disadvantages: i) finding of concept(s) for counterfactuals may be repeated for the same set of errors, and ii) as a result, it tends to generate unnecessarily long concepts.

Some description logic learning algorithms also have been proposed by Lehmann (2010), where two top-down refinement operators have been proposed. One is for the $\mathcal{EL}$ language and the other is for $\mathcal{ALC}$ (although it can be extended to more expressive languages). The latter is the most expressive refinement operator proposed so far. In addition, several learning approaches based on the proposed refinement operators were also developed. Two

interesting algorithms in this framework are Class Expression Learner for Ontology Engineering (CELOE) and OWL Class Expression Learner (OCEL). These algorithms perform very well when compared with other learning approaches (Hellmann, 2008). However, they are sequential algorithms and thus they cannot take advantages of parallel systems such as multi-core processors or cloud computing platforms. In addition, these algorithms focus on generating short descriptions and thus they have another disadvantage: they cannot handle complex learning problems (see Hellmann (2008) and Tran et al. (2012)). Recently, an extension of CELOE has been developed and integrated into the DLLearner framework. This algorithm is the parallelisation of CELOE in which the search tree is expanded by multiple threads. Basically, the learning approach is the same as CELOE, i.e., it looks for a complete definition. Therefore, its performance (learning speed) is improved, but not the capability for learning problems requiring complex (long) concepts.

DL-FOIL (Fanizzi et al., 2008) combines both top-down and bottom-up learning. The top-down step finds a set of *correct concepts* such that each of them correctly defines a subset of the positive examples. Then, the bottom-up step computes a complete concept that defines all positive examples. This approach handles complex concepts better than the top-down or bottom-up approach alone. However, it produces longer concepts in comparison with the approaches proposed by Lehmann (2010). The unnecessarily long learnt concepts are caused by the lack of optimisation in the aggregation step. In addition, like other existing description logic learning algorithms, this approach is serial by nature and thus it cannot take advantage of concurrency.

During the early 90s, a related idea to the overly general descriptions that we use is this paper was proposed by Bain and Muggleton (1990) for first-order theories. Called non-monotonic learning, this approach adopted a specialisation schema based on an existing non-monotonic logic formalism for incremental specialising of over-general beliefs. Although this learning method also employs the negative examples to *correct* the learned beliefs, the use of negative examples in this approach is different from our approach. In Bain and Muggleton's approach, each negative example is intentionally and incrementally used to specialise the learned beliefs. On the other hand, negative examples in our algorithm are mainly used to verify the correctness of the learned beliefs. Their definitions, which are occasionally found in the process of finding the definition for positive examples, are used to correct (specialise) the over-general concepts.

Recent research in description logic learning has focused on the induction instance retrieval cast as a classification problem (Rizzo et al., 2015). They extend the decision tree to describe logic representations (terminological decision trees) by using the nodes to represent conjunctive description logics and left and right branches of the binary tree corresponding to the result of checking some instance against the parent node. In essence, this is a top-down learning approach in which the child nodes are generated by a downward refinement operator.

4.2 Parallelisation in description logic learning

Parallel computing has a history of development since the late 1950s, from multi-processor computer systems to multi-core processors. Parallelisation helps to use computer resources more effectively in order to create fast, efficient scalable algorithms. Currently, parallelisa-

tion can be found in every area of computing, with many parallel frameworks developed, such as Parallel Virtual Machine (PVM) (Sunderam, 2006) and Apache Hadoop (Bhandarkar, 2010).

In logics, parallelisation has been used to develop parallel logic programming languages such as PARLOG (Gregory, 1987) and PEPSys (Ratcliffe and Syre, 1987), allowing parallelised logic deduction. A parallel inductive logic programming approach was proposed by Dehaspe and De Raedt (1995), based on partitioning positive examples into several sets depending on the parallel level of the system and combining the concepts for the partitioned sets at the end. In Matsui et al. (1998), several parallelisation strategies were implemented in FOIL, an inductive logic learning algorithm (Quinlan, 1990). These strategies differ mainly in the dividing strategies, either dividing space among processors (i.e. multiple refinements work in parallel) or partitioning the data and learning the partitions independently, similarly to Dehaspe and De Raedt (1995), although using background knowledge to perform the partition. Our parallel learning approach is similar to the first strategy: we explore the search space in parallel. However, we use a different learning strategy, employing concepts for a subset of positive examples and use a reduction strategy to construct the final concept.

Parallelisation is also used in description logics, but mainly not for reasoning, with only the Deslog reasoner (Wu and Haarslev, 2012), and parallel inferencing algorithms for OWL (Liebig and Müller, 2007; Soma and Prasanna, 2008) being reported to date.

5. Conclusion and Future Work

A symmetric approach to class expression learning has been proposed, where we learn from both positive and negative examples simultaneously. This is motivated by learning scenarios where negative examples can be classified using simple patterns. This is common in practice and our empirical experiments suggest that our approach deals well with this kind of scenario. This approach to class expression learning can deal with other kinds of concept learning problems, as shown in Table 4. For example, the Forte learning problem can be solved by a top-down approach (e.g., CELOE and ParCEL) without using negation. However, this learning problem is solved faster by SPaCEL, which uses the definitions of negative examples, without decreasing the predictive accuracy.

Some current learning algorithms, e.g. CELOE and ParCEL, which were used in our evaluations, can also solve this category of problem by specialising the concepts or using negation and conjunction to remove negative examples from candidate concepts. However, for some data sets with regular exception patterns such as MUBus and UCA1, these algorithms had difficulties in finding the right concept: their learning times were very long in comparison with SPaCEL, which sometimes caused the system to run out of memory before the definition could be found. The most impressive improvements were in the search tree size and learning time. Although SPaCEL often generated longer definitions than other algorithms, there was no over-fitting for the data sets used.

However, the definitions generated by SPaCEL are not optimised. Normalisation and simplification can be used to produce better definitions, i.e. shorter and more readable. This, together with investigations on more data sets, will require further research.

Acknowledgments

Part of this article was written while the first author worked as a researcher at the Vietnam Institute for Advanced Study in Mathematics (VIASM). We gratefully acknowledge the financial support of VIASM for the first author during that time. We also appreciate the constructive comments and suggestions for improving this article made by the anonymous reviewers.

References

Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider. *The description logic handbook: Theory, implementation and applications*. Cambridge University Press, 2010.

Liviu Badea and Shan-Hwei Nienhuys-Cheng. A refinement operator for description logics. *Inductive Logic Programming*, pages 40–59, 2000.

Dennis Bahler. The induction of rules for predicting chemical carcinogenesis in rodents. In *Intelligent Systems for Molecular Biology*, pages 29–37. AAAI/MIT Press, 1993.

Michael Bain and Stephen Muggleton. *Non-monotonic learning*. Turing Institute, 1990.

Milind Bhandarkar. MapReduce programming with apache Hadoop. In *Parallel & Distributed Processing (IPDPS) Symposium*, pages 1–1. IEEE, 2010.

Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

Luc Dehaspe and Luc De Raedt. Parallel inductive logic programming. In *Proceedings of the MLnet Familiarization Workshop on Statistics, Machine Learning and Knowledge Discovery in Databases*, page 5, 1995.

Floriana Esposito, Nicola Fanizzi, Luigi Iannone, Ignazio Palmisano, and Giovanni Semeraro. Knowledge-intensive induction of terminologies from metadata. *The Semantic Web–ISWC 2004*, pages 441–455, 2004.

N. Fanizzi, C. d'Amato, and F. Esposito. DL-FOIL concept learning in description logics. *Inductive Logic Programming*, pages 107–121, 2008.

Andrew Frank and Arthur Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

Dov M. Gabbay, Christopher John Hogger, John Alan Robinson, J. Siekmann, Donald Nute, and Anthony Galton. *Handbook of Logic in Artificial Intelligence and Logic Programming*. Clarendon Press, 1998.

Steve Gregory. Parallel logic programming in PARLOG: The language and its implementation. 1987.

S. Hellmann. Comparison of concept learning algorithms. *Master's Thesis, Leipzig University*, 2008.

Matthew Horridge and Peter F. Patel-Schneider. OWL 2 web ontology language Manchester syntax. *W3C Working Group Note*, 2009.

Ian Horrocks and Peter F. Patel-Schneider. Optimizing description logic subsumption. *Journal of Logic and Computation*, 9(3):267–293, 1999.

L. Iannone, I. Palmisano, and N. Fanizzi. An algorithm based on counterfactuals for concept learning in the semantic web. *Applied Intelligence*, 26(2):139–159, 2007.

J. Lehmann. *Learning OWL Class Expressions*. AKA Akademische Verlagesellschaft, 2010.

J. Lehmann and P. Hitzler. Concept learning in description logics using refinement operators. *Machine Learning*, 78(1):203–250, 2010.

Jens Lehmann. Hybrid learning of ontology classes. *Machine Learning and Data Mining in Pattern Recognition*, pages 883–898, 2007.

Thorsten Liebig and Felix Müller. Parallelizing tableaux-based description logic reasoning. In *On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops*, pages 1135–1144. Springer, 2007.

F. Lisi and D. Malerba. Ideal refinement of descriptions in AL-Log. *Inductive Logic Programming*, pages 215–232, 2003.

John Wylie Lloyd. *Foundations of logic programming*. Springer-verlag Berlin, 1984.

Paul Lyons, An C. Tran, H. Joe Steinhauer, Stephen Marsland, Jens Dietrich, and Hans W. Guesgen. Exploring the responsibilities of single-inhabitant smart homes with use cases. *Journal of Ambient Intelligence and Smart Environments*, 2(3):211–232, 2010.

Tohgoroh Matsui, Nobuhiro Inuzuka, Hirohisa Seki, and Hidenori Itoh. Comparison of three parallel implementations of an induction algorithm. In *8th Int. Parallel Computing Workshop*, pages 181–188, 1998.

Shan-Hwei Nienhuys-Cheng and Ronald De Wolf. *Foundations of Inductive logic programming*. Springer, 1997.

J. Ross Quinlan. Learning logical definitions from relations. *Machine learning*, 5(3):239–266, 1990.

Michael Ratcliffe and Jean-Claude Syre. A parallel logic programming language for PEPSys. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, volume 1, pages 48–55. Morgan Kaufmann Publishers Inc., 1987.

Bradley L. Richards and Raymond J. Mooney. Automated refinement of first-order horn-clause domain theories. *Machine Learning*, 19(2):95–131, 1995.

Giuseppe Rizzo, Claudia dAmato, Nicola Fanizzi, and Floriana Esposito. Terminological tree-based models for inductive classification in description logics. *Semantic Web*, 2015.

Ramakrishna Soma and Viktor K. Prasanna. Parallel inferencing for OWL knowledge bases. In *37th International Conference on Parallel Processing (ICPP'08)*, pages 75–82. IEEE, 2008.

Ashwin Srinivasan, Ross D. King, Stephen Muggleton, and Michael JE. Sternberg. Carcinogenesis predictions using ILP. *Inductive Logic Programming*, pages 273–287, 1997.

Vaidy S. Sunderam. PVM: A framework for parallel distributed computing. *Concurrency: practice and experience*, 2(4):315–339, 2006.

A.C. Tran, J. Dietrich, H. Guesgen, and S. Marsland. An approach to parallel class expression learning. *Rules on the Web: Research and Applications*, pages 302–316, 2012.

An C. Tran. *Symmetric Parallel Class Expression Learning*. PhD thesis, Massey University, 2013.

James Lee Wogulis. *An approach to repairing and evaluating first-order theories containing multiple concepts and negation*. PhD thesis, University of California at Irvine, Irvine, CA, USA, 1994. UMI Order No. GAX94-12189.

Kejia Wu and Volker Haarslev. A parallel reasoner for the Description Logic ALC. In *Proceedings of the 2012 International Workshop on Description Logics (DL 2012)*, 2012.

Journal of Machine Learning Research 18 (2017) 1-35 Submitted 5/15; Revised 10/16; Published 8/17

Learning Partial Policies to Speedup MDP Tree Search via Reduction to I.I.D. Learning

Jervis Pinto
JERVIS.PINTO@GMAIL.COM
School of EECS, Oregon State University, Corvallis OR 97331 USA

Alan Fern
AFERN@EECS.OREGONSTATE.EDU
School of EECS, Oregon State University, Corvallis OR 97331 USA

Editor: Michael Bowling

Abstract

A popular approach for online decision-making in large MDPs is time-bounded tree search. The effectiveness of tree search, however, is largely influenced by the action branching factor, which limits the search depth given a time bound. An obvious way to reduce action branching is to consider only a subset of potentially good actions at each state as specified by a provided partial policy. In this work, we consider offline learning of such partial policies with the goal of speeding up search without significantly reducing decision-making quality. Our first contribution consists of reducing the learning problem to set learning. We give a reduction-style analysis of three such algorithms, each making different assumptions, which relates the set learning objectives to the sub-optimality of search using the learned partial policies. Our second contribution is to describe concrete implementations of the algorithms within the popular framework of Monte-Carlo tree search. Finally, the third contribution is to evaluate the learning algorithms on two challenging MDPs with large action branching factors. The results show that the learned partial policies can significantly improve the anytime performance of Monte-Carlo tree search.

Keywords: online sequential decision-making, partial policy, partial policy learning, imitation learning, Monte-Carlo tree search, reductions

1. Introduction

Online sequential decision-making involves selecting actions in order to optimize a long-term performance objective while meeting practical decision-time constraints. In many environments, including Chess, Go, computer networks, automobiles and aircraft, it is possible to obtain simulators or learn an accurate model of the environment, whereas in others, no such simulator or model is available. A common approach in both cases is to use learning techniques such as reinforcement learning (RL) (Sutton and Barto, 1998) or imitation learning (Pomerleau, 1989; Sammut et al., 1992) in order to learn reactive policies, which can be used to quickly map any state to an action. While reactive policies support fast decision making, for many complex problems, it can be difficult to represent and learn high-quality reactive policies. For example, in games such as Chess, Go, and other complex problems, the best learned reactive policies are significantly outperformed by the best approaches based on look-ahead search (Gelly and Silver, 2007; Veness et al., 2009; Silver et al., 2016). As another example, recent work has shown that effective reactive policies can be learned for playing a variety of Atari video games using both RL (Mnih et al., 2013) and imitation learning (Guo et al., 2014). However, the performance of these reactive policies are easily outperformed or equaled by a vanilla lookahead planning agent (Guo et al., 2014).

Faced with the choice of a planning agent versus a (model-free) learning agent, a practitioner must consider two key factors. The first is the non-trivial expense of obtaining a simulator or learning a model. The second is that the planning agent requires significantly more time per decision. That is, even when a simulator is available, the computational expense may not be practical for certain applications, including real-time Atari game play.

This paper attempts to make progress in the region between fast reactive decision making and slow lookahead search, when a simulator or model of the world is available. In particular, we study algorithms for learning to improve the anytime performance of lookahead search, allowing for more effective deliberative decisions to be made within practical time bounds. Lookahead tree search constructs a finite-horizon lookahead tree using a (possibly stochastic) model or simulator in order to estimate action values at the current state. A variety of algorithms are available for building such trees, including instances of Monte-Carlo Tree Search (MCTS) such as UCT (Kocsis and Szepesvári, 2006), Sparse Sampling (Kearns et al., 2002), and FSSS (Walsh et al., 2010), along with model-based search approaches such as RTDP (Barto et al., 1995) and AO* (Blai and Hector, 2012). However, under practical time constraints (e.g., one second per root decision), the performance of these approaches strongly depends on the action branching factor. In non-trivial MDPs, the number of possible actions at each state is often considerable, which greatly limits the feasible search depth. An obvious way to address this problem is to provide prior knowledge for explicitly pruning bad actions from consideration. In this paper, we consider the offline learning of such prior knowledge in the form of partial policies [1].

A partial policy is a function that quickly returns an action subset for each state and can be integrated into search by pruning away actions not included in the subsets. Thus, a partial policy can significantly speedup search if it returns small action subsets, provided that the overhead for applying the partial policy is small enough. If those subsets typically include high-quality actions, then we might expect little decrease in decision-making quality. Although learning partial policies to guide tree search is a natural idea, it has received surprisingly little attention, both in theory and practice. In this paper we formalize this learning problem from the perspective of "speedup learning". We are provided with a distribution over search problems in the form of a root state distribution and a search depth bound. The goal is to learn partial policies that significantly speedup depth D search, while bounding the expected regret of selecting actions using the pruned search versus the original search method that does not prune.

In order to solve this learning problem, there are at least two key choices that must be made: 1) Selecting a training distribution over states arising in lookahead trees, and 2) Selecting a loss function that the partial policy is trained to minimize with respect to the chosen distribution. The key contribution of our work is to consider a family of reduction-style algorithms that answer these questions in different ways. In particular, we consider three algorithms that reduce partial policy learning to set learning problems, characterized by choices for (1) and (2). The set learning problems are further reduced to cost-sensitive binary classification. Our main results bound the sub-optimality of tree search using the learned partial policies in terms of the expected loss of the supervised learning problems. Interestingly, these results for learning partial policies mirror similar

1. A preliminary version of this work appeared in UAI 2014 (Pinto and Fern, 2014).

©2017 Jervis Pinto, and Alan Fern.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-251.html.

reduction-style results for learning (complete) policies via imitation (Ross and Bagnell, 2010; Syed and Schapire, 2010; Ross et al., 2011).

We empirically evaluate our algorithms in the context of learning partial policies to speedup MCTS in two challenging domains with large action branching factors: 1) a real-time strategy game, Galcon and 2) a classic dice game, Yahtzee. The results show that using the learned partial policies to guide MCTS leads to significantly improved anytime performance in both domains. Furthermore, we show that several other existing approaches for injecting knowledge into MCTS are not as effective as using partial policies for action pruning and can often hurt search performance rather than help.

2. Problem Setup

We consider sequential decision-making in the framework of Markov Decision Processes (MDPs). An MDP is a tuple (S, A, P, R), where S is a finite set of states, A is a finite set of actions, $P(s'|s,a)$ is the transition probability of arriving at state s' after executing action a in state s, and $R(s,a) \in [0,1]$ is the reward function giving the reward of taking action a in state s. In environments where the reward is non-deterministic (but still stationary), $R(s,a)$ is taken to be the expected value. The typical goal in MDP planning and learning is to compute a policy for selecting an action in any state, such that following the policy (approximately) maximizes some measure of long-term expected reward. For example, two popular choices are maximizing the expected finite-horizon total reward or expected infinite-horizon discounted reward.

In practice, regardless of the long-term reward measure, for large MDPs, the offline computation of high-quality policies over all environment states is impractical. In such cases, a popular action-selection approach is online tree search, where at each encountered environment state, a time-bounded search is carried out in order to estimate action values. Note that this approach requires the availability of either an MDP model or a simulator in order to construct search trees. In this paper, we assume that a model or simulator is available and that online tree search has been chosen as the action selection mechanism. Next, we formally describe the paradigm of online tree search, introduce the notion of partial policies for pruning tree search, and then formulate the problem of offline learning of such partial policies.

2.1 Online Tree Search

Throughout this paper, we will focus on search that has a depth bound D. Doing so bounds the length of future action sequences to be considered. Given a state s, we denote by $T(s)$ the depth D expectimax tree rooted at s. $T(s)$ alternates between layers of state nodes and action nodes[2], labeled by states and actions respectively. The children of each state node are action nodes for each action in A. The children of an action node a with parent labeled by s are all states s' such that $P(s'|s,a) > 0$. Figure 1(a) shows an example of a depth two expectimax tree. The depth of a state node is the number of action nodes traversed from the root to reach it. Note that leaves of $T(s)$ will always be action nodes.

The optimal value of a state node s at depth d, denoted by $V_d^*(s)$, is equal to the maximum value of its child action nodes, which we denote by $Q_d^*(s,a)$ for child a. $V_d^*(s)$ and $Q_d^*(s,a)$ are formally defined next.

2. Also known as "chance" nodes in the context of expectimax trees.



(a) Unpruned expectimax tree $T(s)$ with $D = 2$

(b) Pruning T with **partial** policy ψ gives T_ψ where a third of the actions have been pruned.

Figure 1: Unpruned and pruned expectimax trees with depth $D = 2$ for an MDP with $|A| = 3$ and two possible next states. The distributions over the transitions to the next state are uniform. The numeric values are the optimal action values Q^*. Nodes for which no value is shown have $Q^*(s,a) = 0$. Nodes marked with an asterisk indicate that the node corresponds to the optimal action choice. In (b), pruning can change the value at the root if nodes on the optimal paths are incorrectly pruned.

$$V_d^*(s) = \max_a Q_d^*(s,a)$$

$$Q_d^*(s,a) = R(s,a) \quad \text{if} \quad d = D-1 \tag{1}$$

$$= R(s,a) + E_{s' \sim P(\cdot \mid s,a)}\left[V_{d+1}^*(s')\right] \quad \text{otherwise} \tag{2}$$

Equation 1 corresponds to leaf action nodes. In Equation 2, $s' \sim P(\cdot|s,a)$ ranges over the children of a. Given these value functions, the optimal action policy for state s at depth d is

$$\pi_d^*(s) = \arg\max_a Q_d^*(s,a)$$

Given an environment state s, online search algorithms such as UCT or RTDP attempt to completely or partially search the tree $T(s)$ in order to approximate the root action values $Q_0^*(s,a)$ well enough to identify the optimal action $\pi_0^*(s)$. It is important to note that optimality in our context is with respect to the specified search depth D, which may be significantly smaller than the number of actions that will be taken in the actual environment (e.g., the length of a full game). This is a practical necessity that is often referred to as receding-horizon control. Here we simply assume that an appropriate search depth D has been specified and our goal is to speedup planning within that depth.

2.2 Search with a Partial Policy

One way to speedup depth D search is to prune actions from $T(s)$. In particular, if a fixed fraction σ of actions are removed from each state node, then the size of the tree would decrease by a factor of $(1-\sigma)^D$, potentially resulting in significant computational savings. In this paper, we will utilize partial policies for pruning actions. A depth D (non-stationary) *partial policy* ψ is a sequence $(\psi_0, \ldots, \psi_{D-1})$ where each ψ_d maps a state to an action subset. In this work, we focus exclusively on non-stationary partial policies, rather than also considering stationary partial policies where $\psi_i = \psi_j$ for all i and j. One reason for this is that optimal solutions to depth-bounded search trees are

non-stationary in general. In addition, considering non-stationary policies allows for defining and analyzing relatively simple training algorithms as described in section 3. In some applications, however, where the planning depth is not known in advance, it may be desirable to learn stationary partial policies. Extending our work to the stationary scenario is a point of future work, which we expect can draw on ideas for learning stationary policies in imitation learning (Ross and Bagnell, 2010; Ross et al., 2011).

Given a partial policy ψ and root state s, we can define a pruned tree $T_\psi(s)$ that is identical to $T(s)$, except that each state s at depth d only has subtrees for actions in $\psi_d(s)$, pruning away subtrees for any child $a \notin \psi_d(s)$. The phrase "root state" is not to be confused with the starting state of a sequential problem (e.g., the initial board in Chess). Rather, "root state" refers to *any* state in which the search agent is required to act and must therefore place the state at the root of the search tree. Figure 1(b) shows a pruned tree $T_\psi(s)$, where ψ prunes away one action at each state. It is straightforward to incorporate ψ into a search algorithm by only expanding actions at state nodes that are consistent with ψ.

We define the state and action values relative to $T_\psi(s)$ the same as before and let $V_d^\psi(s)$ and $Q_d^\psi(s,a)$ denote the depth d state and action value functions, as follows.

$$\begin{aligned} V_d^\psi(s) &= \max_{a \in \psi_d(s)} Q_d^\psi(s,a) \\ Q_d^\psi(s,a) &= R(s,a) && \text{if} \quad d = D-1 \\ &= R(s,a) + E_{s' \sim P(\cdot \mid s,a)}\left[V_{d+1}^\psi(s')\right] && \text{otherwise} \end{aligned}$$

We will denote the highest-valued, or greedy, root action of $T_\psi(s)$ as

$$\pi^\psi(s) = \arg\max_{a \in \psi_0(s)} Q_0^\psi(s,a)$$

This is the root action that a depth D search procedure would attempt to return in the context of ψ. We say that a partial policy ψ subsumes a partial policy ψ' if for each s and d, $\psi'_d(s) \subseteq \psi_d(s)$. In this case, it is straightforward to show that for any s and d, $V_d^{\psi'}(s) \leq V_d^\psi(s)$. This is simply because planning under ψ allows for strictly more action choices than planning under ψ'. Note that a special case of partial policies is when $|\psi_d(s)| = 1$ for all s and d, which means that ψ defines a traditional (complete) deterministic MDP policy. In this case, V_d^ψ and Q_d^ψ represent the traditional depth d state value and action value functions for policies. The second special case occurs at the other extreme where $\psi_d = A$ for all d. This corresponds to full-width search and we have $V^* = V^\psi$ and $Q^* = Q^\psi$.

Clearly, a partial policy can reduce the complexity of search by eliminating some actions. However, we are also concerned with the quality of decision-making using $T_\psi(s)$ versus $T(s)$, which we will quantify in terms of expected regret. The regret of selecting action a at state s relative to $T(s)$ is equal to $V_0^*(s) - Q_0^*(s,a)$. Note that the regret of the optimal action $\pi_0^*(s)$ is zero. We prefer partial policies that result in root decisions with small regret over the root state distribution that we expect to encounter, while also supporting significant pruning. For this purpose, if μ_0 is a distribution over root states, we define the expected regret of ψ with respect to μ_0 as

$$\text{REG}(\mu_0, \psi) = E\left[V_0^*(s_0) - Q_0^*(s_0, \pi^\psi(s_0))\right] \quad \text{where } s_0 \sim \mu_0 \tag{3}$$

2.3 Learning Problem

We consider an offline learning setting where we are provided with a model or simulator of the MDP in order to train a partial policy that will be used later for online decision-making. As an illustrative example, consider the domain of Chess, where we wish to learn a partial policy for pruning the search tree. Our approach is to use expensive search offline to play many full games of Chess and to record the sequence of states encountered during those games. Each state encountered during actual play is called a *root state* of a search tree. The collected root states and associated trees are then used to learn a partial policy that can be used to speedup future online search.

More formally, the learning problem provides us with a distribution μ_0 over root states (or a representative sample from μ_0) and a depth bound D. In our Chess example, μ_0 would be a distribution over the states encountered along games played using the expensive search. The intention is for μ_0 to be representative of the states that will be encountered during online use. In this work, we are agnostic about how μ_0 is defined for an application. A typical choice of μ_0 is the distribution of states encountered along trajectories of a receding-horizon controller that makes decisions based on unpruned depth D search. We use this definition of μ_0 in our experiments. This setup is closely related to imitation learning (Syed and Schapire, 2010; Ross and Bagnell, 2010; Ross et al., 2011) since we are essentially treating unpruned, expensive search as the expert to be imitated by a pruned search. Other choices of the initial state distribution μ_0 are the state distributions that arise from more complex imitation learning algorithms such as DAGGER (Ross et al., 2011).

Given μ_0 and D, our "speedup learning" goal is to learn a partial policy ψ with small expected regret $\text{REG}(\mu_0, \psi)$, while providing significant pruning. That is, we want to imitate the decisions of depth D unpruned search via a much less expensive depth D pruned search. In general, there will be a tradeoff between the potential speedup and expected regret. At one extreme, it is always possible to achieve zero expected regret by selecting a partial policy that does not prune. At the other extreme, we can remove the need for any search by learning a partial policy that always returns a single action, which corresponds to a complete policy. However, for many complex MDPs, it can be difficult to learn computationally efficient, or reactive, policies that achieve small regret. Rather, it may be much easier to learn partial policies that prune away many, but not all actions, yet still retain high-quality actions. While such partial policies lead to more search than a reactive policy, the regret may be much less.

In practice, we seek a good tradeoff between the two extremes. The tradeoff between a reactive policy versus a full depth D search-based policy is application-specific. Instead of specifying a particular trade-off point as our learning objective, we develop learning algorithms in the next section that provide some ability to explore different points. In particular, the algorithms are associated with regret bounds in terms of supervised learning objectives that measurably vary with different amounts of pruning.

3. Learning Partial Policies

Given μ_0 and D, we now develop reduction-style algorithms for learning partial policies. The algorithms reduce partial policy learning to a sequence of D i.i.d. supervised *set learning* problems, each producing one partial policy component ψ_d. Informally, a supervised set learning problem is similar to a traditional classification problem, except that the learned functions (partial policies, in this case) return sets of labels rather than a single label. In our work, labels will correspond to actions and the designer will specify the size of the label/action sets that the learned function

should return. The quality of a learned set function will be evaluated in terms of a cost function that provides a measure of whether a set returned by the function contains "good" labels/actions.

More formally, the set learning problem for partial policy component ψ_d will be characterized by a pair (μ_d, C_d), where μ_d is a distribution over states, and C_d is a cost function that, for any state s and action subset $A' \subseteq A$ assigns a prediction cost $C_d(s, A')$. The cost function is intended to measure the quality of A' with respect to including actions that are high quality for s. Typically, the cost function will be a monotone decreasing set function of A' with zero pruning having zero cost, $C_d(s, A) = 0$. Note that a trivial solution to the learning problem, which achieves zero cost, would be to learn a function that always returns the complete action set A. However, such solutions will be avoided by having the designer specify the amount of pruning that they desire, or equivalently the size of the action sets returned. Note that constraining the set to contain exactly one action would correspond to learning complete policies.

We begin by assuming the availability of a set learning algorithm called SETLEARN that takes three inputs, namely, a set of states drawn from μ_d, cost function C_d, and a pruning percentage $0 < \sigma_d < 1$. The output of SETLEARN is a partial policy component ψ_d that returns action sets that contain at most a fraction $1 - \sigma_d$ of the available actions while attempting to minimize the expected cost of ψ_d on μ_d. That is, the goal is to return a ψ_d that minimizes $E[C_d(s, \psi_d(s))]$ for $s \sim \mu_d$. The designer's choice of pruning percentage σ_d (e.g., prune 75% of the actions in each state) is related to the time constraints of the decision problem, with smaller time constraints requiring more pruning.

Given access to SETLEARN we first present our generic reduction algorithm for learning partial policies in Algorithm 1. This algorithm template simply calls SETLEARN on a sequence of

Algorithm 1 A template for learning a partial policy $\psi = (\psi_0, \ldots, \psi_{D-1})$. The template is instantiated by specifying the pairs of distributions and cost functions (μ_d, C_d) for $d \in \{0, \ldots, D-1\}$. SETLEARN is a set learning algorithm that aims to minimize the expected cost of each ψ_d relative to C_d and μ_d. σ_d is a pruning fraction such that SETLEARN returns at most $1 - \sigma_d$ actions. Each partial policy ψ_d is a set function that maps states to action sets of size $(1 - \sigma_d)|A|$.

```
procedure PARTIALPOLICYLEARNER({(μ_d, C_d)}, σ_d)
    for d = 0, 1, ..., D − 1 do
        Sample a training set of states S_d from μ_d
        ψ_d ← SETLEARN(S_d, C_d, σ_d)
    end for
    return ψ = (ψ_0, ψ_1, ..., ψ_{D−1})
end procedure
```

set learning problems and returns a list of the learned partial policies. In order to instantiate this template, it is necessary to specify SETLEARN and the individual set learning problems (μ_d, C_d). We defer details of our implementation of SETLEARN until section 4. Rather, we proceed with the definition of our learning reductions. Each reduction is specified by a particular choice of (μ_d, C_d), such that we can bound the expected regret of ψ (when used for search) by the expected (i.i.d.) costs of the ψ_d returned by SETLEARN.

We begin with the state distributions μ_d. These are specified in terms of distributions induced by (complete) policies. In particular, given a policy π, we let $\mu_d(\pi)$ denote the state distribution produced by the following procedure: Draw an initial state from μ_0, execute π for d steps and return the final state. Since we have assumed an MDP model or simulator, it is straightforward to sample

from $\mu_d(\pi)$ for any provided π. Before proceeding, we state two simple lemmas that will be used to prove our regret bounds.

Lemma 1 *If a complete policy π is subsumed by partial policy ψ, then for any initial state distribution μ_0, REG$(\mu_0, \psi) \leq E[V_0^*(s_0)] - E[V_0^\pi(s_0)]$, for $s_0 \sim \mu_0$.*

Proof Since π is subsumed by ψ, we know that $Q_0^\psi(s, \pi^\psi(s)) = V_0^\psi(s) \geq V_0^\pi(s)$. Since for any a, $Q_0^*(s, a) \geq Q_0^\psi(s, a)$, we have for any state s, $Q_0^*(s, \pi^\psi(s)) \geq V_0^\pi(s)$. The result follows by negating each side of the inequality, followed by adding $V_0^*(s)$, and taking expectations. ■

Thus, we can bound the regret of a learned ψ if we can guarantee that it subsumes a policy whose expected value has bounded sub-optimality. Our three reduction algorithms, presented below, provide different approaches for making such a guarantee.

The second lemma will be used to bound the sub-optimality of complete polices that are subsumed by our learned partial policies, which then allows the application of Lemma 1. Versions of this "performance difference lemma" have been used in prior work (Kakade and Langford, 2002; Bagnell et al., 2003; Ross and Bagnell, 2014).

Lemma 2 *For any two complete policies π and π' and any initial state distribution μ_0,*

$$E\left[V_0^{\pi'}(s_0) - V_0^{\pi}(s_0)\right] = \sum_{d=0}^{D-1} E\left[V_d^{\pi'}(s_d) - Q_d^{\pi'}(s_d, \pi(s_d))\right], \text{where } s_d \sim \mu_d(\pi_{0:d-1})$$

.

Proof Following a prior proof (Ross and Bagnell, 2014), define π_d to be a policy that follows π for d steps and then at step $d+1$ switches to following π' until the maximum depth D. Using the observations that $\pi_0 = \pi'$ and $\pi_D = \pi$, we can derive the following.

$$\begin{aligned}
E\left[V_0^{\pi'}(s_0) - V_0^{\pi}(s_0)\right] &= E\left[\sum_{d=0}^{D-1} V_0^{\pi_d}(s_0) - V_0^{\pi_{d+1}}(s_0)\right] \\
&= \sum_{d=0}^{D-1} E\left[V_0^{\pi_d}(s_0) - V_0^{\pi_{d+1}}(s_0)\right] \\
&= \sum_{d=0}^{D-1} E\left[V_d^{\pi'}(s_d) - Q_d^{\pi'}(s_d, \pi(s_d))\right], \text{where } s_d \sim \mu_d(\pi_{0:d-1})
\end{aligned}$$

The first equality is simply a telescoping sum. The third equality follows due the fact that π_d and π_{d+1} both follow π for the first d steps, which yields the same distribution over s_d. ■

We will use this lemma when the reference policy π' is optimal, that is, $\pi' = \pi^*$. Thus, the sub-optimality of π can be bounded by accumulating the expected sub-optimality of its action choices along trajectories generated by π. Next, we present our three reduction algorithms, which are summarized in Table 1.

3.1 OPI : Optimal Partial Imitation

Perhaps the most straightforward idea for learning a partial policy is to attempt to find a partial policy that is usually consistent with trajectories of the optimal policy π^*. That is, each ψ_d should

OPI	$\mu_d = \mu_d(\pi^*)$ $C_d(s, \psi_d(s)) = 0$ if $\pi^*_d(s) \in \psi_d(s)$, otherwise 1
FT-OPI	$\mu_d = \mu_d(\psi^+_{0:d-1})$ $C_d(s, \psi_d(s)) = 0$ if $\pi^*_d(s) \in \psi_d(s)$, otherwise 1
FT-QCM	$\mu_d = \mu_d(\psi^*)$ $C_d(s, \psi_d(s)) = Q(s, \pi^*_d(s)) - Q(s, \psi^*_d(s))$

Table 1: Instantiations for OPI, FT-OPI and FT-QCM in terms of the template in Algorithm 1. Note that ψ_d is a partial policy while π^*_d, ψ^+_d and ψ^*_d are complete policies.

be learned so as to maximize the probability of containing actions selected by π^*_d with respect to the optimal state distribution $\mu_d(\pi^*)$. This approach is followed by our first algorithm called Optimal Partial Imitation (OPI). In particular, Algorithm 1 is instantiated with $\mu_d = \mu_d(\pi^*)$ (noting that $\mu_0(\pi^*)$ is equal to μ_0 as specified by the learning problem) and C_d equal to zero-one cost. Here $C_d(s, A') = 0$ if $\pi^*_d(s) \in A'$ and $C_d(s, A') = 1$ otherwise. Note that the expected cost of ψ_d in this case is equal to the probability that ψ_d does not contain the optimal action. We refer to this as the pruning error and denote it by

$$e^*_d(\psi) = \Pr_{s \sim \mu_d(\pi^*)}(\pi^*_d(s) \notin \psi_d(s)) \tag{4}$$

A naive implementation of OPI is straightforward. We can generate length D trajectories by drawing an initial state from μ_0 and then selecting actions (approximately) according to π^*_d using standard unpruned search. Defined like this, OPI has the nice property that it only requires the ability to reliably compute actions of π^*_d, rather than requiring that we also estimate action values accurately. This allows us to exploit the fact that search algorithms such as UCT often quickly identify optimal actions, or sets of near-optimal actions, well before the action values have converged. This is an important point since we will approximate π^* using long, computationally expensive runs of UCT, described in section 4.

Intuitively, if the expected cost e^*_d is small for all d, then the regret of ψ should be bounded, since the pruned search trees will generally contain optimal actions for state nodes. The following proof clarifies this dependence. For the proof, given a partial policy ψ, it is useful to define a corresponding complete policy ψ^+ such that $\psi^+_d(s) = \pi^*_d(s)$ whenever $\pi^*_d(s) \in \psi_d(s)$ and otherwise $\psi^+_d(s)$ is the lexicographically least action in $\psi_d(s)$. Note that ψ^+ is subsumed by ψ.

Theorem 3 *For any initial state distribution μ_0 and partial policy ψ, if for each $d \in \{0, \ldots, D-1\}$, $e^*_d(\psi) \leq \epsilon$, then REG$(\mu_0, \psi) \leq \epsilon D^2$.*

Proof Given the assumption that $e^*_d(\psi) \leq \epsilon$ and that ψ^+ selects the optimal action whenever ψ contains it, we know that $e^*_d(\psi^+) \leq \epsilon$ for each $d \in \{0, \ldots, D-1\}$. Given this constraint on ψ^+, we can apply Lemma 3 from (Syed and Schapire, 2010)[3], which implies $E[V_0^{\psi^+}(s_0)] \geq E[V^*_0(s_0)] - \epsilon D^2$, where $s_0 \sim \mu_0$. The result follows by combining this with Lemma 1. ■

This result mirrors work on reducing imitation learning to supervised classification (Ross and Bagnell, 2010; Syed and Schapire, 2010), showing the same dependence on the planning horizon. Borrowing again from imitation learning, it is straightforward to construct an example problem where the above regret bound is shown to be tight. This result motivates a learning approach where each ψ_d returned by SETLEARN attempts to maximize pruning (returns small action sets) while maintaining a small expected cost.

3.2 FT-OPI : Forward Training OPI

OPI has a potential weakness, similar in nature to issues identified in prior work on imitation learning (Ross and Bagnell, 2010; Ross et al., 2011). In short, OPI does not train ψ to recover from its own pruning mistakes. Consider a node n in the optimal subtree of a tree $T(s_0)$. Now suppose that the learned ψ erroneously prunes the optimal child action of n. This means that the optimal subtree under n will be pruned from $T_\psi(s)$, increasing the potential regret. Ideally, we would like the pruned search in $T_\psi(s)$ to recover from the error gracefully and return an answer based on the best remaining subtree under n. Unfortunately, the distribution used to train ψ by OPI was not necessarily representative of this alternate subtree under n, since it was not an optimal subtree of $T(s)$. Thus, no guarantees about the pruning accuracy of ψ can be made under node n.

In imitation learning, this type of problem has been dealt with via "forward training" of non-stationary policies (Ross et al., 2011). We employ a similar idea in the Forward Training OPI (FT-OPI) algorithm. FT-OPI differs from OPI only in the state distributions used for training. The key idea is to learn the partial policy components ψ_d in sequential order from $d = 0$ to $d = D - 1$. Each ψ_d is trained on a distribution induced by $\psi_{0:d-1} = (\psi_0, \ldots, \psi_{d-1})$, which will account for pruning errors made by $\psi_{0:d-1}$. Specifically, recall that for a partial policy ψ, we defined ψ^+ to be a complete policy that selects the optimal action if it is consistent with ψ and otherwise the lexicographically least action. The state distributions used to instantiate FT-OPI in Algorithm 1 are $\mu_d = \mu_d(\psi^+_{0:d-1})$ and the cost function remains zero-one cost as for OPI. Thus, the expected cost of ψ_d is $e^+_d(\psi) = \Pr_{s \sim \mu_d(\psi^+_{0:d-1})}(\pi^*_d(s) \notin \psi_d(s))$, which gives the probability of pruning the optimal action with respect to the state distribution of $\psi^+_{0:d-1}$.

Note that as for OPI, we only require the ability to compute π^* in order to sample from $\mu_d(\psi^+_{0:d-1})$. In particular, note that when learning ψ_d, we have $\psi_{0:d-1}$ available. Hence, we can sample a state from μ_d by executing a trajectory of $\psi^+_{0:d-1}$. Actions for ψ^+_d can be selected by first computing π^*_d and selecting it if it is in ψ_d and otherwise selecting the lexicographically least action.

As shown for the forward training algorithm for imitation learning (Ross et al., 2011), we give below an improved regret bound for FT-OPI under an assumption on the maximum sub-optimality

3. The main result of (Syed and Schapire, 2010) holds for stochastic policies and requires a more complicated analysis that results in a looser bound. Lemma 3 is strong enough for deterministic policies.

of any action. The intuition is that if it is possible to discover high-quality subtrees, even under sub-optimal action choices, then FT-OPI can learn on those trees and recover from errors.

Theorem 4 *Assume that for any state s, depth d, and action a, we have $V_d^*(s) - Q_d^*(s,a) \leq \Delta$. For any initial state distribution μ_0 and partial policy ψ, if for each $d \in \{0,\ldots,D-1\}$, $e_d^+(\psi) \leq \epsilon$, then REG$(\mu_0,\psi) \leq \epsilon\Delta D$.*

Proof We first bound the sub-optimality of ψ^+. By applying Lemma 2 with $\pi' = \pi^*$ and $\pi = \psi'$ we can infer

$$E\left[V_0^*(s_0) - V_0^{\psi^+}(s_0)\right] = \sum_{d=0}^{D-1} E\left[V_d^*(s_d) - Q_d^*(s_d, \psi^+(s_d))\right], \text{where } s_d \sim \mu_d(\psi^+_{0:d-1}). \quad (5)$$

Since we have assumed that $e_d^+(\psi) \leq \epsilon$, we know that for each d the probability that ψ^+ does not select an optimal action for states $s_d \sim \mu_d(\psi^+_{0:d-1})$ is no more than ϵ. In addition, by assumption, the worst case regret of such erroneous action choices is bounded by Δ. Thus, each expectation term of the right-hand-side of Equation 5 can be bounded by $\epsilon\Delta$. Summing these D terms then implies that

$$E\left[V_0^*(s_0) - V_0^{\psi^+}(s_0)\right] \leq \epsilon\Delta D.$$

Since ψ^+ is subsumed by ψ, we can apply Lemma 1 to yield the result. ■

This result implies that if Δ is significantly smaller than D, then FT-OPI has the potential to outperform OPI given the same bound on zero-one cost. In the worst case $\Delta = D$ and the bound will equal to that of OPI.

3.3 FT-QCM: Forward Training Q-Cost Minimization

While FT-OPI addressed one potential problem with OPI, they are both based on zero-one cost, which raises other potential issues. The primary weakness of zero-one cost is its inability to distinguish between large pruning errors and small pruning errors. It was for this reason that FT-OPI required the assumption that *all* action values had sub-optimality bounded by Δ. However, in many problems, including those in our experiments, that assumption is unrealistic, since there can be many highly sub-optimal actions. For example, in Chess, some actions might lead to unavoidable defeat. This motivates using a cost function that is sensitive to the sub-optimality of pruning decisions.

In addition, it can often be difficult to learn a ψ that has small zero-one cost while also providing significant pruning. For example, in many domains, in some states there will often be many near-optimal actions that are difficult to distinguish from the slightly better optimal action. In such cases, achieving low zero-one cost may require producing large action sets. However, learning a ψ that provides significant pruning while reliably retaining at least one near-optimal action may be easily accomplished. This again motivates using a cost function that is sensitive to the sub-optimality of pruning decisions, which is accomplished via our third algorithm, Forward Training Q-Cost Minimization (FT-QCM)

The cost function of FT-QCM is the minimum sub-optimality, or Q-cost, over unpruned actions. In particular, we use $C_d(s,A') = V_d^*(s) - \max_{a\in A'} Q_d^*(s,a)$. Our state distribution will be defined similarly to that of FT-OPI, only we will use a different reference policy. Given a partial policy ψ, define a new complete policy $\psi^* = (\psi_0^*,\ldots,\psi_{D-1}^*)$ where $\psi_d^*(s) = \arg\max_{a\in\psi_d(s)} Q_d^*(s,a)$, so

that ψ^* always selects the best unpruned action. We define the state distributions for FT-QCM as the state distribution induced by ψ^*, i.e. $\mu_d = \mu_d(\psi^*_{0:d-1})$. We will denote the expected Q-cost of ψ at depth d to be $\Delta_d(\psi) = E\left[V_d^*(s_d) - \max_{a\in\psi_d(s_d)} Q_d^*(s_d,a)\right]$, where $s_d \sim \mu_d(\psi^*_{0:d-1})$.

Unlike OPI and FT-OPI, this algorithm requires the ability to estimate action values of sub-optimal actions in order to sample from μ_d. That is, sampling from μ_d requires generating trajectories of ψ_d^*, which means we must be able to accurately detect the action in $\psi_d(s)$ that has maximum value, even if it is a sub-optimal action. The additional overhead for doing this during training depends on the search algorithm being used. For many algorithms, near-optimal actions will tend to receive more attention than clearly sub-optimal actions. In those cases, as long as $\psi_d(s)$ includes reasonably good actions, there may be little additional regret. We use long runs of UCT to approximate ψ_d^* and use UCT's Q estimates in the cost function. The full implementation is described in section 4.

There is a significant theoretical benefit to using expected Q-cost for learning compared to zero-one cost. The following bound, which motivates the FT-QCM algorithm, shows that the dependence on D decreases from worst-case quadratic (for OPI and FT-OPI) to linear.

Theorem 5 *For any initial state distribution μ_0 and partial policy ψ, if for each $d \in \{0,\ldots,D-1\}$, $\Delta_d(\psi) \leq \Delta$, then REG$(\mu_0,\psi) \leq \Delta D$.*

Proof Applying Lemma 2 with $\pi' = \pi^*$ and $\pi = \psi^*$ we get that

$$E\left[V_0^*(s_0) - V_0^{\psi^*}(s_0)\right] = \sum_{d=0}^{D-1} E\left[V_d^*(s_d) - Q_d^*(s_d, \psi^*(s_d))\right], \text{where } s_d \sim \mu_d(\psi^*_{0:d-1}).$$

By our assumption that $\Delta_d(\psi) \leq \Delta$ for all d, we can combine this with the above to obtain

$$E\left[V_0^*(s_0) - V_0^{\psi^*}(s_0)\right] \leq \Delta D.$$

Since ψ^* is subsumed by ψ we can apply Lemma 1 to get the result. ■

FT-QCM tries to minimize this regret bound by minimizing $\Delta_d(\psi)$ via supervised learning at each depth. As we will show in our experiments, it is possible to maintain small expected Q-cost with significant pruning, while the same amount of pruning would result in a much larger zero-one cost. When this is true, the benefits of FT-QCM over OPI and FT-OPI can be significant, which is shown in our experiments.

4. Implementation Details

This section first describes the UCT algorithm, which is the base planner used in all the experiments. We then describe the partial policy representation and the learning algorithm. Finally, we specify how we generate the training data.

4.1 UCT

UCT (Kocsis and Szepesvári, 2006) is an online planning algorithm. Given the current state s, UCT selects an action by building a sparse lookahead tree over the state space reachable from s. Thus, s is at the root of the tree. Edges from s correspond to actions and their outcomes so that the tree

consists of alternating layers of state and action nodes. Finally, leaf nodes correspond to terminal states. Each state node in the tree stores Q estimates for each of the available actions. During search, the Q values are used to select the next action to be executed. The algorithm is anytime, which means that a decision can be obtained by querying the root at any time. A common practice is to run the algorithm for a fixed number of trajectories and then select the root action which has the largest Q value.

UCT became famous for advancing the state-of-the-art in Computer Go (Gelly et al., 2006; Gelly and Silver, 2007). Since then, however, many additional successful applications have been reported, including but not limited to the IPCC planning competitions (Keller and Helmert, 2013), general game playing (Méhat and Cazenave, 2010; Finnsson, 2012), Klondike Solitaire (Bjarnason et al., 2009), tactical battles in real-time strategy games (Balla and Fern, 2009) and feature selection (Gaudel and Sebag, 2010). See Browne et al. (2012) for a comprehensive survey.

The algorithm is unique in the way that it constructs the tree and estimates action values. Unlike standard minimax search or sparse sampling (Kearns et al., 2002), which build depth-bounded trees and apply evaluation functions at the leaves, UCT neither imposes a depth bound nor does it require an evaluation function. Rather, UCT incrementally constructs a tree and updates action values by carrying out a sequence of Monte-Carlo rollouts of entire trajectories from the root to the terminal state. The key idea in UCT is to bias the rollout trajectories towards promising ones, indicated by prior trajectories, while continuing to explore. The outcome is that the most promising parts of the tree are grown first while still guaranteeing that an optimal decision will be made given sufficient time and memory. Tree construction is incremental since each rollout introduces only a small number of additional nodes to the tree, typically one state node and its immediate children (action nodes).

There are two key algorithmic choices in UCT. The first is the policy used to conduct each rollout. We employ the popular method of random action selection which is simple, fast, and requires no domain knowledge. The second choice is the method for updating the value estimates in the tree in response to the outcome produced by the rollout. A commonly used technique requires that each node store: a) the number of times the state (or action) node has been visited in previous rollouts $n(s)$ (or $n(s,a)$), b) the current estimate of each action value $Q(s,a)$. Given this information at each state node, UCT performs a rollout starting at the root state of the tree. If there exist actions that have yet to be tried, then a random choice is made over these untried actions. Otherwise, when all actions have been tried at least once, the algorithm selects the action that maximizes an upper confidence bound given by

$$Q_{UCB}(s,a) = Q(s,a) + c\sqrt{\frac{\log n(s)}{n(s,a)}} \tag{6}$$

where c is a positive constant that controls the exploration bonus. In practice, c is often tuned separately on a per-domain basis. The selected action is then executed and the resulting state is added to the tree if it is not already present. This action selection rule is based on the UCB rule (Auer et al., 2002) for multi-armed bandits, which seeks to balance exploration and exploitation. The first term $Q(s,a)$ is large for actions with high Q value estimates (exploitation), while the second term is large for actions that have been visited less often (exploration), relative to the number of visits to the parent state. The exploration bonus decreases as the action is visited more often.

Having described a procedure to generate trajectories, we now describe the method used to update the tree. After the trajectory reaches a terminal state and receives a reward R, the action values and counts of each state along the trajectory are updated. In particular, for any state action pair (s,a), the update rules are,

$$n(s) \leftarrow n(s) + 1 \tag{7}$$

$$n(s,a) \leftarrow n(s,a) + 1 \tag{8}$$

$$Q(s,a) \leftarrow Q(s,a) + \frac{1}{n(s,a)}(R - Q(s,a)) \tag{9}$$

The equation computes the average reward of rollout trajectories that pass through the node (s,a). As previously mentioned, the algorithm is run for some fixed number of simulations before selecting the best action at the root, typically, the action with the largest Q value, although selecting the action with the most visits is also popular. This particular algorithm has two parameters, the exploration constant c, and the number of rollout trajectories.

The above version is the most basic form of UCT, which has received a very large amount of attention in the game-playing and planning community. See Browne et al. (2012) for a comprehensive survey of the MCTS family of algorithms. We defer the discussion of the MCTS family and relevant UCT variants to section 5. Next, we describe the partial policy representation and learning framework used in our experiments.

4.2 Partial Policy Representation and Learning

We begin by reducing partial policy learning to set learning. As mentioned before, the designer chooses a fixed fraction of actions to prune in each state. The pruning fraction σ_d can vary from 0 to 1, where $\sigma_d = 1$ indicates maximum pruning. The choice of σ_d allows the designer to tradeoff the time constraints of the decision problem with the expected cost achievable by the SETLEARN algorithm. In this work, we perform a post-hoc analysis of the expected cost of ψ_d for a range of pruning values and select values of σ_d that yield reasonably small costs. In section 6, we give details of the selections used in our experiments.

Once σ_d is fixed, we have induced a set learning problem. Next, we convert the set learning problem to a ranking problem. That is, for states at depth d, we seek to learn a scoring function over actions that will allow us to prune away the bottom σ_d fraction of actions. In this work, we use linear ranking functions for their computational efficiency and simplicity, although non-linear approaches (e.g., deep neural networks) may also be used. Formally, we have a n-dimensional weight vector w_d and a user-provided feature vector $\phi(s,a)$ over state-action pairs. The linear ranking function $f_d(s,a) = w_d^T\phi(s,a)$ allows us to define a total order over actions, breaking ties lexicographically. Thus, w_d along with the pruning fraction σ_d fully parameterizes the partial policy component ψ_d, which is defined as the set of $\lceil(1-\sigma_d)\times|A|\rceil$ highest ranked actions.

Next, we specify how we implement the SETLEARN procedure. Each training set will consist of pairs $\{(s_i,c_i)\}$, where s_i is a state and c_i is a vector that assigns a cost to each action. We first describe the reduction to cost-sensitive binary classification and then specify the method by which the training data is generated from long runs of UCT. For OPI and FT-OPI, the cost vector assigns 0 to the optimal action and 1 to all other actions. For FT-QCM, the c_i give the Q-costs of each action, obtained from the Q estimates during offline UCT. We learn the partial policy by first learning the

ranking function in a way that attempts to rank the optimal action as highly as possible and then select an appropriate pruning fraction based on the learned ranker.

For rank learning, we follow a common approach of converting the problem to cost-sensitive binary classification. In particular, for a given example (s, c) with optimal action a^*, we create a cost-sensitive classification example for each action $a_j \neq a^*$ of the form,

$$(s, c) \quad \longrightarrow \quad \{(\phi(s, a^*) - \phi(s, a_j)\ ,\ c(a^*) - c(a_j)) \quad | \quad a_j \neq a^*\}$$

Learning a linear classifier for such an example will attempt to rank a^* above a_j according to the cost difference. This produces $m - 1$ training examples for a state with m actions. We apply an existing cost-sensitive learner (VW, (Langford, 2011)) to learn a weight vector based on the pairwise data. The learning algorithm is standard binary classification with squared loss and L_2 regularization, set to 0.01. We also randomly flip the ranking constraints with uniform probability in order to balance the binary class labels. In VW format, each training example consists of a cost, binary label, and feature difference vector. Finally, we experimented with other methods of generating ranking constraints for the reduction from set learning to ranking (e.g., all pairs, unpruned vs pruned). The simple technique described above performed best in this experimental setup. Note that this implies that the training of f_d is independent of σ_d.

4.3 Generating Training States

Each of our algorithms requires sampling training states from trajectories of particular policies. OPI and FT-OPI require approximately computing π^*_d. FT-QCM has the additional requirement of approximating the action values for sub-optimal actions. Our implementation of this is to first generate a set of trees using substantial search, which provides us with the required policy or action values. We then sample trajectories from those trees.

More specifically, our learning algorithm is provided with a set of root states by using expensive runs of UCT to select actions along complete trajectories, starting from a state drawn from the initial state distribution of a domain. We let S_0 denote the set of states encountered along these trajectories, noting that we can view S_0 as being drawn from the target distribution μ_0. For each $s_0 \in S_0$, we store the tree produced by UCT, noting that the resulting trees will typically have a large number of nodes on the tree fringe that have been infrequently visited. Since such states are not useful for learning, we select a depth bound D such that nodes at depth $d < D$ have been sufficiently explored and have meaningful action values. The trees are then pruned to depth D.

Given this set of depth D trees, we can now generate execution trajectories using the action selection rule for each algorithm (Table 1) and the MDP simulator to sample subsequent states. For example, OPI simply requires running trajectories through the trees based on selecting actions according to the optimal action estimates at each tree node. The state at depth d along each trajectory is added to the data set for training ψ_d. FT-QCM samples states for training ψ_d by generating length d trajectories of $\psi^*_{0:d-1}$.

Each such action selection requires referring to the estimated action values and returning the highest-valued action that is not pruned. The final state on the trajectory is then added to the training set for ψ_d. Note that since our approach assumes i.i.d. training sets for each ψ_d, we only sample a single trajectory from each tree. However, our experiments indicate that multiple trajectories can be sampled from the tree without significant performance loss (although the guarantee is lost). Doing so allows many additional examples to be generated per tree, avoiding the need for additional expensive tree construction.

Algorithm	Action cost vector	Sample trajectory
OPI	$[..., I_{a^*=a_i}, ...]$	$s_0 \xrightarrow{\pi^*_0(s_0)} s_1 \xrightarrow{\pi^*_1(s_1)} s_2 \ ...$
FT-OPI	$[..., I_{a^*=a_i}, ...]$	$s_0 \xrightarrow{\psi^+_0(s_0)} s_1 \xrightarrow{\psi^+_1(s_1)} s_2 \ ...$
FT-QCM	$[... , Q^*_d(s, a^*) - Q^*_d(s, a_i),\ ...]$	$s_0 \xrightarrow{\psi^*_0(s_0)} s_1 \xrightarrow{\psi^*_1(s_1)} s_2 \ ...$

Figure 2: Training data generation and ranking reduction for each algorithm. Start by sampling s_0 from the same initial state distribution μ_0. OPI uses the optimal policy at each depth whereas FT-OPI and FT-QCM use the most recently learned complete policy (at depth $d - 1$). OPI and FT-OPI use 0-1 costs. Only FT-QCM uses the Q cost function, which turns out to be critical for good performance.

5. Related Work

While there is a large body of work on integrating learning and planning, we do not know of any work on learning partial policies for speeding up online MDP planning.

There are a number of efforts that study model-based reinforcement learning (RL) for large MDPs that utilize tree search methods for planning with the learned model. Examples include RL using FSSS (Walsh et al., 2010), Monte-Carlo AIXI (Veness et al., 2011), and TEXPLORE (Hester and Stone, 2013). However, these methods focus on model/simulator learning and do not attempt to learn to speedup tree search, which is the focus of our work.

Work on learning search control knowledge in deterministic planning and games is more related. One research direction has been on learning knowledge for STRIPS-style deterministic planners. Examples of this approach are learning heuristics and policies for guiding best-first search (Yoon et al., 2008) or state ranking functions (Xu et al., 2009). The problem of learning leaf evaluation heuristics has also been studied in the context of deterministic real-time heuristic search (Bulitko and Lee, 2006). As another example, evaluation functions for game tree search have been learned from the "principle variations" of deep searches (Veness et al., 2009). A related body of work involves iterative deepening heuristic search and its variants (Korf, 1985; Reinefeld and Marsland, 1994). Control knowledge is typically leveraged using heuristic scoring functions, transposition tables and principal variations. These methods often improve search by using memory-intensive control knowledge (e.g., hash tables) and incrementally expanding the search, but do not provide theoretical guarantees. Nevertheless, the idea of progressive widening and iterative deepening is extremely appealing and one that we intend to explore more in future work.

The idea of using deep UCT search to generate high-quality state-action training pairs has been proposed recently (Guo et al., 2014). This work is intuitively similar to the work in this paper. The high-quality training data is used to train a stationary reactive policy using a deep neural network (DNN) policy. This is standard imitation learning, where the goal is to learn a reactive policy. The resulting DNN policy performs very well on a number of ATARI games. However, even this sophisticated reactive policy cannot outperform deep (but slower) UCT search. This work provides examples of domains in which an advanced reactive controller is outperformed by sufficiently deep lookahead tree search. Our work attempts to provide a rigorous middle ground between reactive decision-making and expensive unpruned search.

There have been a number of efforts for utilizing domain-specific knowledge in order to improve/speedup MCTS, many of which are covered in recent surveys (Browne et al., 2012; Gelly et al., 2012). The two main methods involve progressive bias and progressive widening. The core idea of progressive bias involves adding a quantity $f(s, a)$ for guiding action selection during search. $f(s, a)$ may be hand-provided (Chaslot et al., 2007), or learned (Gelly and Silver, 2007; Sorg et al., 2011). Generally, there are a number of parameters that dictate how strongly $f(s, a)$ influences search and how that influence decays as search progresses. In Sorg et al. (2011), control knowledge is learned via policy-gradient techniques in the form of a reward function and used to guide MCTS with the intention of better performance given a time budget. So far, however, the approach has not been analyzed formally and has not been demonstrated on large MDPs. Experiments in small MDPs have also not demonstrated improvement in terms of wall clock time over vanilla MCTS. The second method, progressive widening, is more popular in problems with large or continuous action spaces. The main idea here is to start the search with small action sets and introduce new actions as search progresses (Couëtoux et al., 2011a,b). For example, one may add a new action (sometimes called "unpruning") when the number of visits to a state exceeds a heuristically chosen threshold. Although progressive widening is intuitively appealing and clearly required for anytime search in large action spaces, we were unable to find any existing method that leverages control knowledge or works in a principled manner.

Finally, MCTS methods often utilize hand-coded or learned rollout policies (sometimes called "default policies") to improve anytime performance. A very well-known example is MoGo (Gelly et al., 2006). While this approach has shown promise in specific domains such as Go, where the policies can be highly engineered for efficiency, we have found that the large computational overhead of a learned rollout policy makes its usage hard to justify. The key issue is that the use of any learned rollout policy requires the evaluation of a feature vector at each state (or state-action pair) encountered during the rollout. Rollouts are typically far longer than the depth of the search tree, especially during the initial part of the sequential problem. This results in a massive number of additional feature evaluations, far greater than the number of evaluations performed by the heuristic bias methods discussed above which only require a feature evaluation once for every new action node added to the tree. Thus, the use of learned rollout policies may cause orders of magnitude fewer trajectories to be executed, compared to vanilla MCTS. In our experience, this can easily lead to degraded performance per unit time. Furthermore, there is little formal understanding about how to learn such rollout policies in principled ways, with straightforward approaches often yielding decreased performance (Silver and Tesauro, 2009).

6. Experiments

We perform a large-scale empirical evaluation of the partial policy learning algorithms on two challenging MDP problems. The first, Galcon, is a variant of a popular real-time strategy game, while the second is Yahtzee, a classic dice game. Both problems pose a serious challenge to MCTS algorithms, for different reasons. However, in both domains, UCT, enhanced with learned partial policies, is able to significantly improve real-time search performance over every baseline. The key experimental findings are listed next.

1. The search performance of UCT using a partial policy learned by the FT-QCM algorithm, denoted as UCT(FT-QCM), is significantly better than that of FT-OPI and OPI in both domains.

2. UCT(FT-QCM) performs significantly better than vanilla UCT and other "informed" UCT variants when the search budget is small. Given a larger search budget, UCT(FT-QCM) wins in Galcon and achieves parity in Yahtzee.

3. The average regret of the pruning policies on a supervised dataset of states is strongly correlated with the search performance, which is in line with our formal results. FT-QCM has significantly lower regret than FT-OPI and OPI.

4. The quality of the training data deteriorates quickly as depth d increases. Regret increases as d increases.

5. It is critical to incorporate the computational cost of using control knowledge into the search budget during evaluation. Ignoring the computational cost of knowledge can significantly change the perceived relative performance of different algorithms.

6.1 Experimental Domains

Galcon: The first domain is a variant of a popular two-player real-time strategy game, illustrated in Figure 3. The agent seeks to maximize its population by launching variable-sized population fleets from one of the planets currently under its control. The intention may be to either reinforce planets already controlled, attack an enemy planet, or capture an unclaimed planet. This produces a large action space ($O(|\text{planets}|^2)$), where an action is defined as the triple (source planet, destination planet, fleet size). In our experiments, we use maps with 20 planets and three discrete levels of fleet size (small, medium, large). This often leads to game states where the action space contains hundreds of actions. The exact set of actions varies during the game as control over planets changes. State transitions are typically deterministic, unless a battle occurs. During each battle, a unit is randomly chosen from the two armies for termination. The battle ends when only one army remains. The maximum length of a game is restricted to 90 moves.

The search complexity caused by the large action branching factor is exacerbated by the relatively slow simulator and the length of the game. In this setting, UCT can only perform a small number of rollouts even when the search budget is as large as eight seconds per move. In fact, even at the largest search budgets, UCT only sparsely expands the tree beyond depths two or three. Note, however, that the rollout component of UCT is run to the end of the game and hence provides information well beyond these shallow depths. Finally, the second player or adversary is fixed to be a UCT agent with a budget of one second per move. All evaluations are performed against this fixed adversary. Note that at one second per decision, the fixed adversary effectively acts randomly, especially during the initial part of the game. However, further increase in the adversary's search budget is impractical because of the significant computational cost of evaluating search agents in this domain.

The agent's performance can be measured in terms of the final population difference. This value is zero when the game is tied, a large positive value when the agent wins convincingly and a negative value when the agent is much weaker than the adversary. The rollout policy used is random unless mentioned otherwise, while the exploration bias c is set to 0.75 in Galcon, after a brief grid search showed that search performance was relatively invariant to the exploration bonus as long as the parameter was set sufficiently high. The exploration bias was not optimized in favor of any of the three algorithms presented in this paper and is kept constant.

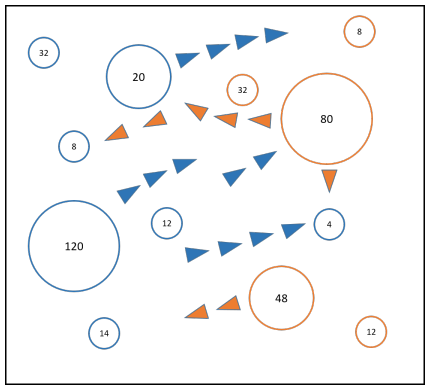


Figure 3: A visualization of a Galcon game in progress. The agent must direct variable-sized fleets to either reinforce planets under its control, take over enemy planets, or unoccupied ones. The decision space is large, often involving hundreds of actions.

Yahtzee: The second domain is a classic dice game which consists of thirteen stages. In each stage, the agent may roll any subset of the five dice at most twice, after which the player must select one of the empty categories. A category, once selected, may not be reselected in later stages. This produces a total of 39 decisions that must be made per game, where the last action trivially assigns the only remaining category. The categories are divided into upper and lower categories with each one scored differently. The lower categories roughly correspond to poker-style conditions ("full house", "three of a kind", "straight", etc.), while the upper categories seek to maximize the individual dice faces (ones, twos, ..., sixes). A bonus of 35 points is awarded when the sum of the upper category scores exceeds 63. This ensures that the player must very carefully assign categories since the loss of the bonus has a large impact on the final score. The maximum score at the end of any game is 375, which is very hard to achieve. The above implementation is the simplest version of the game, restricted to a single "Yahtzee" (all five dice faces are identical). Our implementation for the simulator will be made available upon request[4]. Figure 4 shows a summary of the rules of the game. The UCT algorithm, including all parameter values, is identical to the one used in Galcon.

The trees generated by UCT differ significantly in the two domains. Compared to Galcon, Yahtzee has a relatively small action branching factor, smaller time horizon and a fast simulator. Furthermore, the available actions and the branching factor in Yahtzee decrease as categories are filled, whereas in Galcon, the agent occupying more planets has a larger action set. Thus, we can play many more games of Yahtzee per unit time compared to Galcon. However, Yahtzee poses a difficult challenge for MCTS algorithms due to the dice rolls which increase the state branching factor and add variance to the Q value estimates. Consequently, even deep search with the UCT algorithm is error-prone and struggles to improve performance with larger search budgets. As in Galcon, the search quality of UCT degrades significantly at depths beyond two or three. Note, however, that

4. We are also in the process of open-sourcing the entire code base.

UPPER CATEGORIES

Aces	= 3	COUNT AND ADD ONLY ACES	
Twos	= 6	COUNT AND ADD ONLY TWOS	
Threes	= 9	COUNT AND ADD ONLY THREES	
Fours	=12	COUNT AND ADD ONLY FOURS	
Fives	=15	COUNT AND ADD ONLY FIVES	
Sixes	=18	COUNT AND ADD ONLY SIXES	
TOTAL	= 63	➡	U
Bonus	IF 63 OR OVER	SCORE 35	B

LOWER CATEGORIES

3 of a kind		ADD TOTAL OF ALL DICE	
4 of a kind		ADD TOTAL OF ALL DICE	
Full house		SCORE 25	
Sm. Straight	Sequence of 4	SCORE 30	
Lg. Straight	Sequence of 5	SCORE 40	
YAHTZEE	5 OF A KIND	SCORE 50	
Chance		SCORE TOTAL OF ALL 5 DICE	
TOTAL	OF LOWER HALF	➡	L

TOTAL SCORE = U + B + L

Figure 4: A typical scoring sheet for Yahtzee, consisting of five dice and 13 categories. In each game, all 13 categories must be scored, one per stage. Each stage consists of at most three dice rolls (original roll + two more controlled by the player). The objective is to maximize the total score. The dice rolls increase the tree size and adds variance during rollouts. Our simulator implements the variant where at most one "Yahtzee" (all five dice faces are identical) can be recorded.

the performance degradation in Yahtzee is primarily caused by the transition stochasticity from the dice rolls, although the action branching factor remains significant. As in Galcon, UCT rolls out trajectories to the end of the game. Doing so provides information from states much deeper than the depth of the tree.

Normalizing rewards

The final score is always non-negative for Yahtzee, but may be negative for Galcon if the agent loses the game. In order to achieve a consistent comparison between the supervised metrics and search performance, we perform a linear rescaling of the score so that the performance measure now lies in the interval $[0, 1]$. In our experiments, we provide this measure as reward to the agent at the end of each trajectory. That is, rewards are zero everywhere except in terminal state-action pairs. Note that a reward above 0.5 indicates a win in Galcon. In Yahtzee, a reward value of 1.0 is the theoretical upper limit achievable in the best possible outcome, which rarely occurs in practice. Furthermore, even a small increase in the reward (e.g., 0.1) is a significant improvement in Yahtzee.

6.2 Partial Policy Learning Setup

We now describe the procedure used to generate training data for the three learning algorithms, OPI, FT-OPI, and FT-QCM. Each algorithm is provided with root states generated by playing approximately 200 full games. In Galcon, UCT is allowed to search for 600 seconds per move, whereas in Yahtzee, UCT is given 120 seconds per move. Each game results in a trajectory of states visited during the game. All of those states across all games constitute the set of root states used for training as described in section 4. Despite the enormous search budget, depths beyond two or three are sparsely expanded, due to the large state and action branching factors. We therefore set the search depth bound (D) to 3 for learning, since the value estimates produced by UCT for deeper nodes are often inaccurate. Thus, the learned partial policy has the form $\psi = (\psi_0, \psi_1, \psi_2)$. When UCT generates tree nodes at depths greater than D, we prune using ψ_2. As previously mentioned, we do not limit the rollouts of UCT to depth D. Rather, the rollouts proceed until terminal states, which provide a longer-term heuristic evaluation for tree nodes. However, we have confirmed experimentally that such nodes are not visited frequently enough for pruning to have a significant impact.

As previously mentioned, we learn partial policies by learning linear scoring functions. This requires us to compute state-action features $\phi(s, a)$ that capture salient attributes. We now describe the features used in each domain.

Galcon features: The features are defined over the next state (or "afterstate") produced by simulating the action in the current state. The salient attributes of the next state for launch actions include attributes of the source and destination planet like the planet size, its current population, the growth rate, and the incoming enemy threat. All real-valued features are discretized into 100 bins, using a one-hot encoding. For destination planets, the feature vector consists of the concatenation of three one-hot sub-vectors, corresponding to whether the destination planet is occupied by the current player, by the adversary, or unoccupied by either player. Note that exactly one of the three sub-vectors is non-zero for a particular destination planet. For "do nothing" actions, we have a single indicator feature. Finally, we also have a constant (bias) feature. This leads to a total feature space of size 1602, but only a small number (typically nine) are active for a given state-action pair. However, the computation of $\phi(s, a)$ is still slow due to the simulator call to obtain the next state.

Yahtzee features: The features for the current state-action pair are defined over the next state, which is sampled by calling the simulator on the current state-action pair. The state-action feature vector is specified using a sparse one-hot encoding, where each of the 13 categories has 100 features assigned to it with one additional bias feature. The feature value for each unfilled category is a measure of how "close" the dice are to achieving the maximum score for that category, discretized into 100 bins. A perfect score returns a feature value of 1, whereas a category selection that does not add to the game score returns a feature value of 0. However, "select" actions for the upper categories that achieve binary outcomes (e.g., either a "full house" is achieved or not), are encoded using binary indicator features. All "roll" actions use discretized features, encoding how close the roll action brings the dice to a desired configuration. Note that there is large variance in the state-action features due to the dice rolls. This variance may be reduced by sampling more states and averaging. However, doing so would significantly increase the computational overhead of computing $\phi(s, a)$.

Evaluation metrics: We report the search budget in real-time (e.g., 4 seconds per move) instead of the number of simulations. Each point on the anytime curve is the average of 1000 final game scores, with 95% confidence intervals. This large-scale evaluation has massive computational cost and requires access to a cluster. The use of time-based search budgets in a cluster environment adds variance, requiring additional evaluations and averaging. However, an important result of the following experiments is that the use of time to measure budgets is essential for a fair comparison of "knowledge-injected" search algorithms.

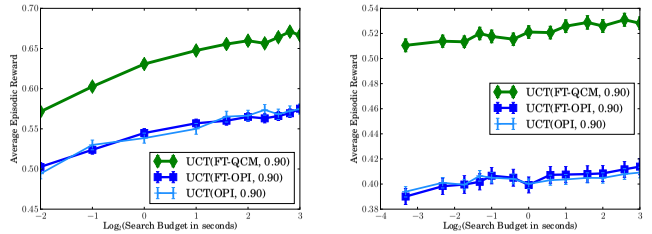
6.3 Comparing OPI, FT-OPI, and FT-QCM

We learn ranking functions using each of the three learning algorithms, as described in section 4.2. We obtain a set of partial policies by varying the pruning thresholds σ_d. Recall from section 4.2 that our implementation of rank function learning does not depend on σ_d. The learned partial policies are used to prune actions during search. Figure 5 shows the impact on search performance using UCT as the fixed base search algorithm. The first row shows the results for a large amount of pruning ($\sigma_d = 0.90$) while the second and third rows have progressively smaller pruning thresholds.

The first observation is that FT-QCM is clearly better than FT-OPI and OPI in both domains. FT-OPI and OPI are only competitive with FT-QCM in Galcon if small pruning is applied (bottom left). In Yahtzee, FT-QCM is significantly better even when the pruning is reduced to 50%.

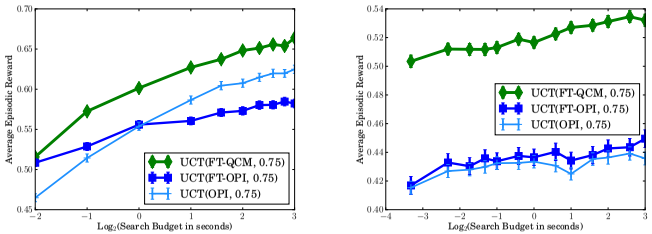
Second, we observe that all search algorithms perform better as the search budget is increased. However, the pruning thresholds significantly impact the performance gain across the anytime curve. In Galcon, this impact is highlighted by the steep curves for $\sigma_d = 0.5$, compared to the relatively flat curves for larger pruning thresholds. In Yahtzee, although performance appears to increase very slowly, this small increase corresponds to a significant improvement in playing strength.

The clear superiority of FT-QCM merits further investigation. Recall that FT-QCM differs from FT-OPI only in terms of the cost function. Given the near-identical performance of FT-OPI and OPI, it seems that forward training is less relevant to performance. Therefore, we conclude that the performance improvement comes from incorporating the Q-cost instead of zero-one cost. Next, we analyze the learned partial policies to gain additional insight into the impact of Q-cost learning, compared to zero-one cost and forward training.



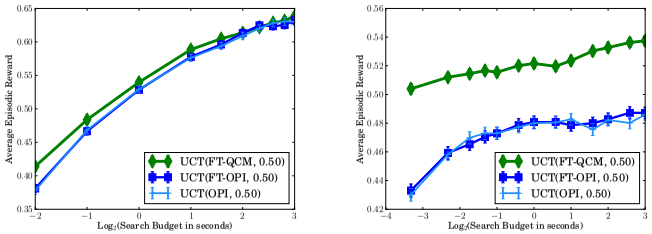
(a) Galcon, $\sigma_d = 0.90$

(b) Yahtzee, $\sigma_d = 0.90$



(c) Galcon, $\sigma_d = 0.75$

(d) Yahtzee, $\sigma_d = 0.75$



(e) Galcon, $\sigma_d = 0.50$

(f) Yahtzee, $\sigma_d = 0.50$

Figure 5: The search performance of UCT using partial policies learned by FT-QCM, FT-OPI, and OPI as a function of the search budget. The results for Galcon are in the first column, the results for Yahtzee in the second. Each row corresponds to a different set of pruning thresholds. The top row shows the results of pruning 90% of the actions whereas the bottom row prunes 50%. FT-QCM easily outperforms FT-OPI and OPI in five instances and is tied in the sixth. As mentioned in section 6.1, the reward (bounded by $[0, 1]$) is a linear function of the raw score, measured at the end of the game.

6.4 Relating search and regret

In order to better understand how search performance is related to the quality of the pruning policies, we analyze the learned partial policies on a held-out dataset of trajectories. In particular, for each linear partial policy, we compute its average pruning error and average regret as the level of pruning is increased from none ($\sigma_d = 0$) to maximum ($\sigma_d = 1.0$). Recall the definition of pruning error from Equation 4, which is the probability of ψ pruning away the optimal action (w.r.t. the expert policy). The definition for regret is given in Equation 3. These supervised metrics are computed by averaging over a held-out dataset of root states taken from at least one thousand trajectories. Figure 6 shows these supervised metrics on Galcon for each of the three depths. The left column shows the average regret while the right column is the average pruning error rate. The result of pruning randomly is also shown since it corresponds to pruning with random action subsets. Figure 7 shows the corresponding results for Yahtzee.

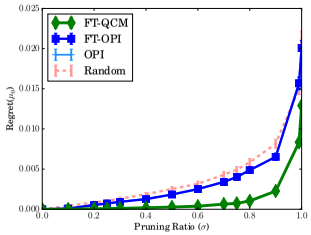
We start at the root, which is the first row in both figures. As expected, both regret and error increase as more actions are pruned. The key result here is that FT-QCM has significantly lower regret than FT-OPI and OPI for both domains. The lower regret is unsurprising given that FT-QCM uses the regret-based cost function while FT-OPI and OPI use zero-one costs. However, FT-QCM also demonstrates the least pruning error in both problems. Given that the same state distribution is used at the root, this observation merits additional analysis.

There are two main reasons why FT-OPI and OPI do not learn good policies. First, the linear policy space has very limited representational capacity. Second, the expert trajectories are noisy. The poor quality of the policy space is highlighted by the spike in the regret at maximum pruning, indicating that all three learners have difficulty imitating the expert. We also observe this at the end of training, where the final 0-1 error rate remains high. The expert's shortcomings lie in the difficulty of the problems themselves. For instance, in Galcon, very few action nodes at depths $d > 0$ receive sufficient visits. In Yahtzee, the stochasticity in dice rolls makes it very difficult to estimate action values accurately. Furthermore, in states where many actions are qualitatively similar (e.g., roll actions with nearly identical Q values), the expert's choice of action is effectively random. This adds noise to the training examples, which makes the imitation learning problem harder. However, FT-QCM's use of regret-based costs allows it to work well in Yahtzee at all depths, but only at the root in Galcon. In Galcon, the expert deteriorates rapidly away from the root policy and the training dataset becomes very noisy. Given the dominating performance of FT-QCM over OPI and FT-OPI, we focus on FT-QCM for the remainder of the experiments.

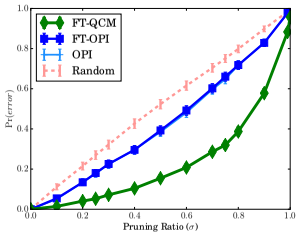
6.5 Selecting σ_d

We will now describe a simple technique for selecting σ_d. Recall that a good pruning threshold must "pay for itself". That is, in order to justify the time spent evaluating knowledge, a sufficient number of sub-optimal actions must be pruned so that the improvement produced is larger than what would have been achieved by simply searching more. At one extreme, too much pruning incurs large regret and little performance improvement with increasing search budgets. At the other extreme, if insufficient actions are pruned, search performance may be worse than that of uninformed (vanilla) search due to the additional overhead of computing state-action features.

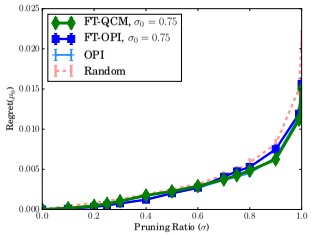
The key to selecting good values lies in the supervised metrics discussed above, where the regret and search graphs are correlated. Thus, one simple technique for selecting σ_d is to use the largest value that has sufficiently small regret. For instance, in Galcon, regret is near-zero for $\sigma_0 < 0.75$ and



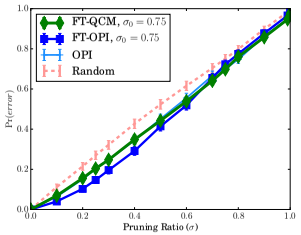
(a) Galcon: Regret at the root (d=0)



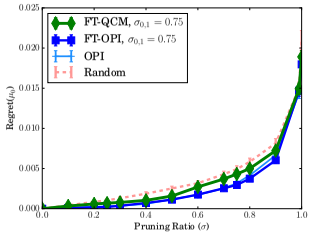
(b) Galcon: Pruning error rate at the root (d=0)



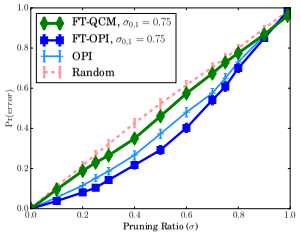
(c) Galcon: Regret at d=1



(d) Galcon: Pruning error rate at d=1

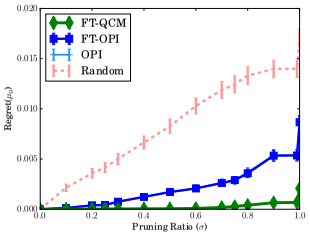


(e) Galcon: Regret at d=2

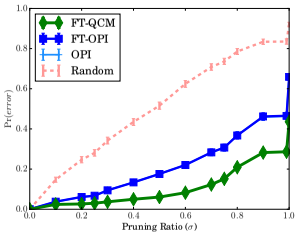


(f) Galcon: Pruning error rate at d=2

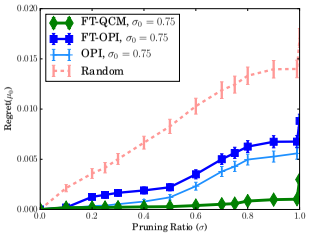
Figure 6: Average regret and pruning error for the learned partial policies ψ_d as a function of the pruning ratio in Galcon. The metrics are evaluated with respect to a held-out test set of states obtained from at least 1000 trajectories with root states sampled from μ_0. FT-QCM has the least error and regret at the root. At depths $d > 0$, performance degrades towards random due to noisy training data.



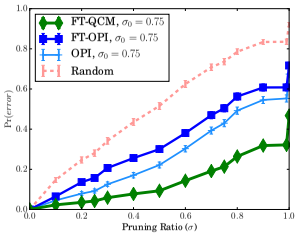
(a) Yahtzee: Regret at the root (d=0)



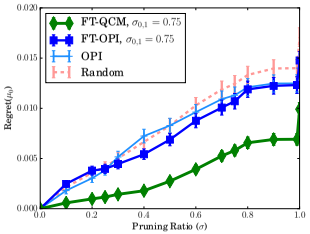
(b) Yahtzee: Pruning error rate at the root (d=0)



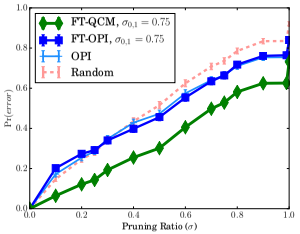
(c) Yahtzee: Regret at d=1



(d) Yahtzee: Pruning error rate at d=1



(e) Yahtzee: Regret at d=2



(f) Yahtzee: Pruning error rate at d=2

Figure 7: Average regret and pruning error for the learned partial policies ψ_d as a function of the pruning ratio in Yahtzee. The metrics are evaluated with respect to a held-out test set of states obtained from at least 1000 trajectories with root states sampled from μ_0. At depths $d > 0$, FT-QCM continues to perform better than FT-OPI and OPI due to the relatively higher quality of training data at deeper states.

increases sharply thereafter. The regret is practically zero for Yahtzee until the pruning threshold exceeds 0.5 and only seems to increase significantly around 0.75. For simplicity, we choose a constant pruning threshold and set σ_d to 0.75 for every value of d. As we will see next, using FT-QCM partial policies with $\sigma_d = 0.75$ results in large performance gains across the anytime curve. A more computationally expensive method of selecting σ_d would require the evaluation of the planning performance of a number of different pruning thresholds.

6.6 Comparing FT-QCM with baselines

We will now compare FT-QCM with other search-based and reactive agents. Most of the baselines require control knowledge in the form of either a policy or a heuristic evaluation function. Since the partial policy learned by FT-QCM at the root has the least regret, we will use it in every technique [5]. We denote this partial policy as ψ_G and its linear scoring function as $h_G(s,a) = w_G^T \phi(s,a)$.

Heuristic Bias (HB): Our first baseline is a popular technique for injecting a heuristic into the MCTS framework. HB first adds a bias to an action node's score and then slowly reduces it as the node is visited more often. This has the effect of favoring high-scoring actions early on and slowly switching to unbiased search for large budgets. This technique can be viewed as a form of progressive widening (Chaslot et al., 2007) with biased action selection and encompasses a large body of search variants. For example, one method uses the visit counts to explicitly control the branching factor at each node (Couëtoux et al., 2011a). However, none of these methods come with formal regret bounds and are agnostic to the choice of heuristic, if one is used. In our experiments, we observed that search performance is extremely sensitive to the interaction between the reward estimate, non-stationary exploration bonus, and the non-stationary bias term. In our domains, the following heuristic worked better than the other variations considered.

$$\begin{aligned} q(s,a) &= K h_G(s,a) && \text{if } n_a = 0 \\ &= Q_{UCB}(s,a) + (1-k_n) K h_G(s,a) && n_a > 0 \end{aligned}$$

where K is a domain-specific constant and $k_n = n(s,a)/(n(s,a)+1)$ is zero for an unvisited action and increases towards one as the action is visited more. Thus, the exploration scaling term $(1-k_n)$ is maximum (one) initially and decreases towards zero with each subsequent visit. Next, $Q_{UCB}(s,a)$ is the UCB score of an action node from Equation 6, combining the Q estimate with an exploration bonus for infrequently visited action nodes. The best values of K and the exploration constant c were obtained through a computationally expensive grid search.

Informed Rollout (IR): Another popular technique for using knowledge in UCT involves using an "informed" rollout policy. The underlying idea is that a biased rollout policy can produce more accurate state evaluations than uniform random action selection. Typically, this policy is hand-engineered using expert knowledge (e.g., MoGo (Gelly et al., 2006)), but it can also be learned offline using domain knowledge (Silver and Tesauro, 2009) or online in a domain-independent manner (Finnsson and Björnsson, 2010). Here, we use a MoGo-like strategy of selecting randomly from the partial policy ψ_G using a pruning ratio of 0.75. That is, at each state during a rollout from a leaf node, an action is randomly sampled from the top 25% of the action set with respect to the scoring function h_G. From the experiments in section 6.4, we expect this policy to have very small regret.

5. We also tried regression and RL techniques to learn an evaluation function but none of the approaches worked as well as FT-QCM.

However, "informed" rollout policies have significantly larger computational costs compared to random action selection, since each rollout requires knowledge to be evaluated a very large number of times. That is, the number of knowledge evaluations performed during policy rollouts is massive, compared to the number of knowledge evaluations performed for tree nodes. We experimented with other informed rollout policies, such as softmax as well as a complete policy that prunes all but one action. However, none of these approaches were able to perform significantly better.

UCT with Random pruning (URP): A simple yet interesting baseline is the UCT algorithm that prunes randomly. We use $\sigma_d = 0.75$ for consistency. URP avoids the cost of feature computation while simultaneously reducing the number of actions under consideration. URP is motivated by the insight obtained in the previous section, where performance at small search budgets improved with additional pruning. An important observation is that searching with random action subsets corresponds to evaluating the search performance of a random partial policy which we analyzed in Figure 6 and Figure 7.

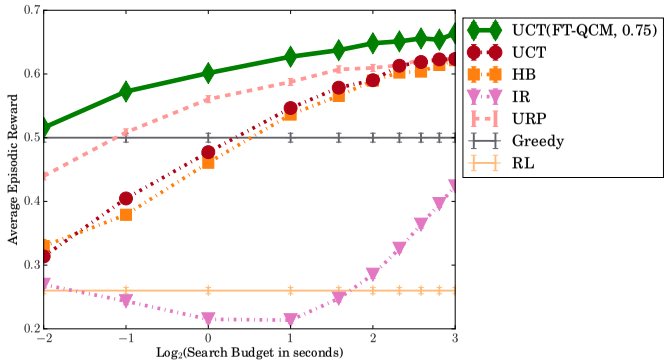
Reactive policies: Finally, we consider policies that do not search. The first, "Greedy", is simply $\arg\max_a h_G(s,a)$, acting greedily w.r.t. ψ_G. The second is a linear policy learned using the SARSA(λ) RL algorithm (Barto et al., 1995) over the same features used to learn ψ_G. We experimented with different values of λ, exploration and learning rates, and report the best settings for each domain. For Galcon, we observe that learning is negligible after 8000 games with exploration rate $\epsilon = 0.25$, decay $\lambda = 0.75$, with a constant learning rate of 0.01. For Yahtzee, which has a much faster simulator, we play 10^5 games using $\epsilon = 0.2$, $\lambda = 0.1$, and the same constant learning rate of 0.01. In both cases, RL is run until improvement becomes negligible.

Results

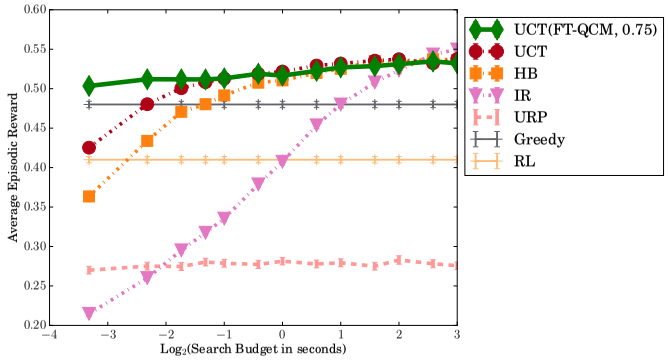
Figure 8(a) and Figure 8(b) show the results for Galcon and Yahtzee, respectively. They constitute the main experimental result of this paper. We discuss the results for each domain separately below.

Galcon: The key observation in Figure 8(a) is that FT-QCM injected into UCT significantly outperforms all other baselines at small search budgets and moderately large search budgets. The search budget is measured in real time. As the search budget increases, the FT-QCM injection continues to show slow improvement, whereas uninformed UCT fails to match performance even at eight seconds per move. We were unable to get HB and IR to perform well, which is surprising given prior reports of successful applications of these methods. IR performs particularly poorly and is worse than even the reactive policies (Greedy, SARSA). The main reason for IR's poor performance is the massive computational overhead of its "informed" rollouts. In Galcon, games (and therefore rollouts) are long which means that computing features for every legal state-action pair incurs significant computational cost. Random rollout policies are much faster in comparison and provide non-trivial value estimates. However, we observe that IR appears to be improving rapidly and may eventually outperform UCT, but only at unreasonably large search budgets (e.g., ten minutes per move).

In contrast, heuristic bias (HB) is relatively inexpensive since it must only be computed once for each state added to the tree. That is, HB has the same overhead as partial policy injection, which allows HB to perform better than IR. However, HB is unable to improve over uninformed UCT. This is surprising since prior work has described search agents that successfully leverage HB. In our experiments, we observe that HB is sensitive to the particular choice of bias decay. Getting HB to work well across the anytime curve is challenging and time-consuming. Next, we consider



(a) Galcon, FT-QCM v baselines



(b) Yahtzee, FT-QCM v baselines

Figure 8: The key empirical result of this paper shows that the search performance of UCT injected with FT-QCM significantly outperforms all other baselines at small budgets. At larger budgets, it either continues to win or achieves parity.

uninformed UCT which has zero overhead. As suspected, UCT is near-random at small time scales, getting outperformed by the variant that prunes randomly (URP). The difficulty of planning in large action spaces is highlighted by the strong performance of URP, which is second only to FT-QCM across the anytime curve. Finally, the reactive policies (Greedy, SARSA) perform reasonably well, but are incapable of utilizing larger time budgets. Greedy, in particular, appears to perform as well as UCT with a search budget of one second per move. This is much better than the policy learned

by the RL algorithm. Finally, at the smallest search budget, Greedy outperforms all other baselines and is second only to FT-QCM.

Yahtzee: FT-QCM easily outperforms all baselines in this domain. However, compared to Galcon, the baselines are far more competitive at larger time scales and a number of methods achieve parity with FT-QCM. Vanilla UCT is particularly good and quickly matches FT-QCM as the search budget is increased. However, it is important to note that despite pruning 75% of the actions, FT-QCM is able to significantly outperform vanilla UCT (and other baselines) for small decision times and does not lose to vanilla UCT at larger search budgets. This result also confirms that small supervised regret (Figure 6) translates into good performance as demonstrated by the strong performance of Greedy. However, as in Galcon, both reactive policies are outperformed by search-based techniques given sufficient time.

HB and IR are uncompetitive in this domain. IR performs very poorly at first, for reasons discussed previously. However, IR eventually overtakes all methods and the graph indicates that performance may improve further, given additional time. However, HB is worse than UCT and only manages to achieve parity at large time scales. The challenges of planning in Yahtzee are clearly visible in the relatively flat anytime curves for each search variant and the high residual variance, despite averaging over a thousand games. Given the relatively small action branching factor compared to Galcon, it is notable that in Yahtzee, FT-QCM is able to perform significantly better than all competing knowledge injections and reactive policies.

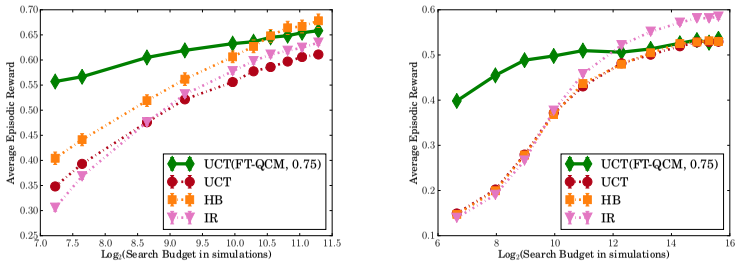
Finally, URP performs very poorly in Yahtzee, in contrast to its good performance in Galcon. The underlying reason is the relatively small percentage of good actions in a typical Yahtzee state. In Yahtzee, URP's random pruning will often result in the elimination of all reasonable actions, which decreases performance. In Galcon, this is less likely to occur since there are a much larger fraction of good actions.

6.7 The cost of knowledge

Given that many successful applications of biased initialization methods like HB and biased rollout methods like IR have exhibited improved search performance in the literature, we were surprised when HB and IR failed to improve over vanilla MCTS in our domains. Much tuning was required for HB to balance the effect of different score components. In contrast, it is relatively simple to bias rollouts and the results indicate that state-of-the-art performance is achievable with IR if one is prepared to search for very long periods of time.

The key issue seems to be the high computational cost of knowledge compared to conducting additional search. In order to evaluate the true cost of knowledge in IR and HB, we switch to measuring performance in terms of the number of simulated trajectories rather than wall-clock time. This evaluation approach ignores the time-cost of applying either the evaluation heuristic or the biased rollout policy. This reporting method has the advantage of being invariant to the architecture and implementation and is widely used in the MCTS literature. Note that we cannot re-compute these results by simply translating the time-based evaluations, since each rollout can take a variable amount of time to finish. For example, rollouts from states at the start of a game take more time compared to rollouts from near-terminal states. We therefore re-evaluate the agents, where each search is now allowed to perform a fixed number of simulations from the root. The impact of measuring search budgets using simulations instead of time is shown in Figure 9.

UCT(FT-QCM) continues to win easily at search budgets that are relatively small. However, as the search budget increases, HB and IR now demonstrate better performance. At the largest search budgets, HB outperforms UCT in Galcon and achieves parity in Yahtzee. IR eventually outperforms all other variants in Yahtzee. This improvement is noteworthy, given that IR performed very poorly in the time-based evaluation. These experiments show that using biased rollouts can improve search performance but only at exorbitant and unpredictable runtimes, varying with the particular state at the root of the tree. A broader observation is that it seems very important to correctly incorporate the cost of knowledge when comparing different injections, which has not always been the case in prior work on MCTS.



(a) Galcon, # of simulations (b) Yahtzee, # of simulations

Figure 9: The results of measuring search performance using the number of simulations, instead of time. Doing so eliminates the computational cost of using control knowledge. IR (UCT with biased rollouts) now appears to perform much better. However, UCT with FT-QCM continues to perform the best across significant portions of the anytime curve.

7. Summary and Future Work

We have shown algorithms for offline learning of partial policies, which can be used to reduce the action branching factor in time-bounded tree search. The algorithms leverage a reduction to i.i.d. supervised learning and the expected regret is shown to be bounded. Experiments in two challenging domains show significantly improved anytime performance in Monte-Carlo Tree Search.

There are a number of promising directions to explore next. The first improvement is to the base learner for the set learning problems. Recent work has studied more principled approaches for such set learning problems via a list learning approach (Dey et al., 2013; Ross et al., 2013). While the simplicity of the ranking approach used in our work is appealing, more sophisticated approaches may yield significant performance gains in certain cases. Importantly, a major benefit of studying algorithms within a reduction framework is that we can easily incorporate any such improvements to base learners without needing to change the overall algorithm.

Another important implementation detail that requires additional exploration is the use of non-stationary partial policies. As mentioned previously, imitation learning algorithms, such as DAGGER (Ross et al., 2011), provide the same regret bounds as forward training using a stationary policy, with data from different distributions combined into a single dataset. It is certainly appealing to have a single policy that shares data and parameters across depth instead of the depth-indexed non-stationary partial policies considered here. However, one practical difficulty that arises in our setting is that the training data extracted from long runs of UCT gets noisier with increasing depth. Our initial attempts to mix data across depths led to poor performance. One way to control noise in the supervised datasets, irrespective of depth, might be to run a separate, long UCT from any state that is to be included in the dataset. It may be feasible to do so given sufficient computational resources. The performance of stationary partial policies compared to non-stationary partial policies is worth future consideration.

Next, we seek a method than can replace the fixed-width forward pruning with a theoretically justified progressive widening method. Doing so would allow the search to smoothly transition from reactive decision-making to full-width search at larger time scales. It would also eliminate the need to specify σ_d upfront. Ideally, the pruning level should depend on the remaining search budget. Such a time-aware search procedure has the potential to produce state-of-the-art anytime search performance. We are unaware of any method that leverages control knowledge for performing time-aware search on a per-decision basis in a principled manner.

We would like to use the learned control knowledge to improve the expert policy, which was responsible for generating the deep trees used for training. The learned partial policy is only as good as the expert, so improvements to the expert may produce large improvements in search quality, particularly at small time scales. For instance, in Yahtzee, we observe that the expert makes serious, irreversible mistakes (for example, in category assignment at the end of the rolls). Avoiding expert mistakes seems crucial to further improvement. An iterative method, alternating between learning control knowledge and using the learned information to improve the expert, is likely to produce good results. However, FT-QCM can only obtain good value estimates for unpruned actions. Thus, it is important to analyze the regret bounds and performance from training only on a subset of actions.

Finally, we are interested in exploring principled learning algorithms for other types of control knowledge that can be shown to have guarantees similar to those in this paper. It can be argued that a huge part of the success of MCTS methods is their flexibility to incorporate different forms of knowledge, which are typically engineered for each application. Automating these processes in a principled way seems to be an extremely promising direction for improving planning systems.

Acknowledgments

This work was supported by NSF grants IIS-1219258 and IIS-1619433. We also thank the editor and anonymous reviewers for their detailed feedback and insights. In particular, the original version of this work stated a much looser version of Theorem 5. The new version is based on reviewer feedback.

References

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning Journal*, 47(2-3):235–256, 2002.

J Andrew Bagnell, Sham M Kakade, Jeff G Schneider, and Andrew Y Ng. Policy search by dynamic programming. In *Advances in neural information processing systems*, page None, 2003.

Radha-Krishna Balla and Alan Fern. Uct for tactical assault planning in real-time strategy games. In *International Joint Conference on Artificial Intelligence*, pages 40–45, 2009.

Andrew G Barto, Steven J Bradtke, and Satinder P Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72(1):81–138, 1995.

Ronald Bjarnason, Alan Fern, and Prasad Tadepalli. Lower Bounding Klondike Solitaire with Monte-Carlo Planning. In *International Conference on Automated Planning and Scheduling*, 2009.

Bonet Blai and Geffner Hector. Action Selection for MDPs: Anytime AO* Versus UCT. In *AAAI Conference on Artificial Intelligence*, 2012.

Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. A survey of monte carlo tree search methods. *Computational Intelligence and AI in Games, IEEE Transactions on*, 4(1):1–43, 2012.

Vadim Bulitko and Greg Lee. Learning in Real-Time Search: A Unifying Framework. *Journal of Artificial Intelligence Research*, 25:119–157, 2006.

Guillaume Chaslot, Mark Winands, Jaap H van den Herik, Jos Uiterwijk, and Bruno Bouzy. Progressive strategies for monte-carlo tree search. In *Joint Conference on Information Sciences*, pages 655–661, 2007.

Adrien Couëtoux, Jean-Baptiste Hoock, Nataliya Sokolovska, Olivier Teytaud, and Nicolas Bonnard. Continuous upper confidence trees. In *Learning and Intelligent Optimization*, pages 433–445. 2011a.

Adrien Couëtoux, Mario Milone, Matyas Brendel, Hassen Doghmen, Michele Sebag, Olivier Teytaud, et al. Continuous rapid action value estimates. In *The 3rd Asian Conference on Machine Learning*, volume 20, pages 19–31. JMLR, 2011b.

Debadeepta Dey, Tian Yu Liu, Martial Hebert, and J Andrew Bagnell. Contextual sequence prediction with application to control library optimization. *Proceedings of robotics: Science and systems VIII*, 2013.

Hilmar Finnsson. *Simulation-Based General Game Playing*. PhD thesis, Reykjavik University, 2012.

Hilmar Finnsson and Yngvi Björnsson. Learning Simulation Control in General Game-Playing Agents. In *AAAI Conference on Artificial Intelligence*, pages 954–959, 2010.

Romaric Gaudel and Michele Sebag. Feature selection as a one-player game. In *International Conference on Machine Learning*, pages 359–366, 2010.

Sylvain Gelly and David Silver. Combining online and offline knowledge in UCT. In *International Conference on Machine Learning*, pages 273–280, 2007.

Sylvain Gelly, Yizao Wang, Rémi Munos, and Olivier Teytaud. Modification of UCT with patterns in Monte-Carlo Go. Technical report, INRIA, 2006.

Sylvain Gelly, Levente Kocsis, Marc Schoenauer, Michele Sebag, David Silver, Csaba Szepesvári, and Olivier Teytaud. The grand challenge of computer go: Monte carlo tree search and extensions. *Communications of the ACM*, 55(3):106–113, 2012.

X. Guo, S. Singh, H. Lee, R. L. Lewis, and X. Wang. Deep learning for real-time atari game play using offline monte-carlo tree search planning. In *Advances in Neural Information Processing Systems*, 2014.

Todd Hester and Peter Stone. TEXPLORE: real-time sample-efficient reinforcement learning for robots. *Machine Learning Journal*, 90(3):385–429, 2013.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *ICML*, volume 2, pages 267–274, 2002.

Michael Kearns, Yishay Mansour, and Andrew Y Ng. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Machine Learning Journal*, 49(2-3):193–208, 2002.

Thomas Keller and Malte Helmert. Trial-based heuristic tree search for finite horizon mdps. In *International Conference on Automated Planning and Scheduling*, 2013.

Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *European Conference on Machine Learning*, pages 282–293. 2006.

Richard E Korf. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial intelligence*, 27(1):97–109, 1985.

John Langford. Vowpal Wabbit. *URL https://github. com/JohnLangford/vowpal_wabbit/wiki*, 2011.

Jean Méhat and Tristan Cazenave. Combining uct and nested monte carlo search for single-player general game playing. *Computational Intelligence and AI in Games, IEEE Transactions on*, 2(4): 271–277, 2010.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Jervis Pinto and Alan Fern. Learning partial policies to speedup mdp tree search. In *Conference on Uncertainty in Artificial Intelligence*, 2014.

D. Pomerleau. ALVINN: an autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems*, 1989.

Alexander Reinefeld and T. Anthony Marsland. Enhanced iterative-deepening search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(7):701–710, 1994.

Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *International Conference on Artificial Intelligence and Statistics*, pages 661–668, 2010.

Stephane Ross and J Andrew Bagnell. Reinforcement and imitation learning via interactive no-regret learning. *arXiv preprint arXiv:1406.5979*, 2014.

Stéphane Ross, Geoffrey J Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, pages 627–635, 2011.

Stephane Ross, Jiaji Zhou, Yisong Yue, Debadeepta Dey, and Drew Bagnell. Learning policies for contextual submodular prediction. In *Proceedings of The 30th International Conference on Machine Learning*, pages 1364–1372, 2013.

Claude Sammut, Scott Hurst, Dana Kedzier, and Donald Michie. Learning to fly. In *International Workshop on Machine Learning*, 1992.

David Silver and Gerald Tesauro. Monte-Carlo simulation balancing. In *International Conference on Machine Learning*, pages 945–952, 2009.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Jonathan Sorg, Satinder P Singh, and Richard L Lewis. Optimal Rewards versus Leaf-Evaluation Heuristics in Planning Agents. In *AAAI Conference on Artificial Intelligence*, 2011.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. Cambridge Univ Press, 1998.

Umar Syed and Robert E Schapire. A Reduction from Apprenticeship Learning to Classification. In *Advances in Neural Information Processing Systems*, pages 2253–2261, 2010.

Joel Veness, David Silver, Alan Blair, and William W Cohen. Bootstrapping from game tree search. In *Advances in Neural Information Processing Systems*, pages 1937–1945, 2009.

Joel Veness, Kee Siong Ng, Marcus Hutter, William Uther, and David Silver. A Monte-Carlo AIXI approximation. *Journal of Artificial Intelligence Research*, 40(1):95–142, 2011.

Thomas J Walsh, Sergiu Goschin, and Michael L Littman. Integrating Sample-Based Planning and Model-Based Reinforcement Learning. In *AAAI Conference on Artificial Intelligence*, 2010.

Yuehua Xu, Alan Fern, and Sungwook Yoon. Learning linear ranking functions for beam search with application to planning. *The Journal of Machine Learning Research*, 10:1571–1610, December 2009.

Sungwook Yoon, Alan Fern, and Robert Givan. Learning control knowledge for forward search planning. *The Journal of Machine Learning Research*, 9:683–718, 2008.

Journal of Machine Learning Research 18 (2017) 1-42 Submitted 7/15; Revised 6/17; Published 8/17

Hierarchically Compositional Kernels for Scalable Nonparametric Learning

Jie Chen chenjie@us.ibm.com
IBM Thomas J. Watson Research Center

Haim Avron haimav@post.tau.ac.il
Tel Aviv University

Vikas Sindhwani sindhwani@google.com
Google Brain, NYC

Editor: Koji Tsuda

Abstract

We propose a novel class of kernels to alleviate the high computational cost of large-scale nonparametric learning with kernel methods. The proposed kernel is defined based on a hierarchical partitioning of the underlying data domain, where the Nyström method (a globally low-rank approximation) is married with a locally lossless approximation in a hierarchical fashion. The kernel maintains (strict) positive-definiteness. The corresponding kernel matrix admits a recursively off-diagonal low-rank structure, which allows for fast linear algebra computations. Suppressing the factor of data dimension, the memory and arithmetic complexities for training a regression or a classifier are reduced from $O(n^2)$ and $O(n^3)$ to $O(nr)$ and $O(nr^2)$, respectively, where n is the number of training examples and r is the rank on each level of the hierarchy. Although other randomized approximate kernels entail a similar complexity, empirical results show that the proposed kernel achieves a matching performance with a smaller r. We demonstrate comprehensive experiments to show the effective use of the proposed kernel on data sizes up to the order of millions.

Keywords: Hierarchical kernels, Nonparametric Learning

1. Introduction

Kernel methods (Schölkopf and Smola, 2001; Hastie et al., 2009) constitute a principled framework that extends linear statistical techniques to nonparametric modeling and inference. Applications of kernel methods span the entire spectrum of statistical learning, including classification, regression, clustering, time-series analysis, sequence modeling (Song et al., 2013), dynamical systems (Boots et al., 2013), hypothesis testing (Harchaoui et al., 2013), and causal modeling (Zhang et al., 2011). Under a Bayesian treatment, kernel methods also admit a parallel view in Gaussian processes (GP) (Rasmussen and Williams, 2006; Stein, 1999) that find broad applications in statistics and computational sciences, including geostatistics (Chilès and Delfiner, 2012), design of experiments (Koehler and Owen, 1996), and uncertainty quantification (Smith, 2013).

This power and generality of kernel methods, however, are limited to moderate sized problems because of the high computational costs. The root cause of the bottleneck is the fact that kernel matrices generated by kernel functions are typically dense and unstructured. For n training examples, storing the matrix costs $O(n^2)$ memory and performing matrix factorizations requires $O(n^3)$ arithmetic operations. One remedy is to resort to compactly supported kernels (e.g., splines (Monaghan and Lattanzio, 1985) and Wendland functions (Wendland, 2004)) that potentially lead to a sparse matrix. In practice, however, the support of the kernel may not be sufficiently narrow for sparse linear algebra computations to be competitive. Moreover, prior work (Anitescu et al., 2012) revealed a subtle drawback of compactly supported kernels in the context of parameter estimation, where the likelihood surface is bumpy and the optimum is difficult to locate. For this reason, we focus on the dense setting in this paper and the goal is to exploit structures that can reduce the prohibitive cost of dense linear algebra computations.

1.1 Preliminary

Let $\mathcal{X}$ be a set and let $k(\cdot,\cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric and strictly positive-definite function. Denote by $X = \{\boldsymbol{x}_i \in \mathcal{X}\}_{i=1,\ldots n}$ a set of points. We write $K(X,X)$, or sometimes K for short when the context is clear, to denote the kernel matrix of elements $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. *Because of the confusing terminology on functions and their counterparts on matrices, here, we follow the convention that a strictly positive-definite function k corresponds to a positive-definite matrix K, whereas a positive-definite function corresponds to a positive semi-definite matrix.* For notational convenience, we write $k(\boldsymbol{x}, X)$ to denote the row vector of elements $k(\boldsymbol{x}, \boldsymbol{x}_j)$ and similarly $k(X, \boldsymbol{x})$ to denote the column vector. In the context of regression/classification, the set $\mathcal{X}$ is often the d-dimensional Euclidean space $\mathbb{R}^d$ or a domain $S \subset \mathbb{R}^d$. Some of the methods discussed in this paper naturally generalize to a more abstract space. Associated to each point $\boldsymbol{x}_i$ is a target value $y_i \in \mathbb{R}$. We write $\boldsymbol{y}$ for the vector of all target values.

The Reproducing Kernel Hilbert Space $\mathcal{H}_k$ associated to a kernel k is the completion of the function space

$$\left\{ \sum_{i=1}^{m} \alpha_i k(\boldsymbol{z}_i, \cdot) \mid \boldsymbol{z}_i \in \mathcal{X},\, \alpha_i \in \mathbb{R},\, m \in \mathbb{Z}_+ \right\}$$

equipped with the inner product

$$\left\langle \sum_i \alpha_i k(\boldsymbol{z}_i, \cdot),\ \sum_j \beta_j k(\boldsymbol{w}_j, \cdot) \right\rangle = \sum_{ij} \alpha_i \beta_j k(\boldsymbol{z}_i, \boldsymbol{w}_j).$$

Given training data $\{(\boldsymbol{x}_i, y_i)\}_{i=1,\ldots,n}$, a typical kernel method finds a function $f \in \mathcal{H}_k$ that minimizes the following risk functional

$$\mathcal{L}(f) = \sum_{i=1}^{n} V(f(\boldsymbol{x_i}), y_i) + \lambda \|f\|^2_{\mathcal{H}_k}, \tag{1}$$

where V is a loss function and $\lambda > 0$ is a regularization. When V is the squared loss $V(t,y) = (t-y)^2$, the Representer Theorem (Schölkopf et al., 2001) implies that the minimizer is

$$f(\boldsymbol{x}) = k(\boldsymbol{x}, X)[K(X,X) + \lambda I]^{-1} \boldsymbol{y}, \tag{2}$$

which is nothing but the well-known kernel ridge regression. Similarly, when V is the hinge loss, the minimizer leads to support vector machines.

©2017 Jie Chen, Haim Avron and Vikas Sindhwani.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-376.html.

In the GP view, the kernel k serves as a covariance function. Assuming a zero-mean Gaussian prior with covariance k, for any separate set of points X_* and the associated vector of target values $\boldsymbol{y}_*$, the joint distribution of $\boldsymbol{y}$ and $\boldsymbol{y}_*$ is thus

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{y}_* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right).$$

Because of the Gaussian assumption, the posterior is the conditional $\boldsymbol{y}_*|X_*, X, \boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ where

$$\boldsymbol{\mu} = K(X_*, X)K(X,X)^{-1}\boldsymbol{y}, \tag{3}$$

and

$$\Sigma = K(X_*, X_*) - K(X_*, X)K(X,X)^{-1}K(X, X_*). \tag{4}$$

A white noise of variance λ may be injected to the observations $\boldsymbol{y}$ so that the mean prediction $\boldsymbol{\mu}$ in (3) is identical to (2). One may also impose a nonzero-mean model in the prior to capture the trend in the observations (see, e.g., the classic paper O'Hagan and Kingman (1978) and also Rasmussen and Williams (2006, Section 2.7)).

Equations (2)–(4) exemplify the demand for kernels that may simplify computations for a large $K(X,X)$. We discuss a few popular approaches in the following. To motivate the discussion, we consider stationary kernels whose function value depends on only the difference of the two input arguments; that is, we can write by abuse of notation $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{r})$, where $\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{x}'$; for example, the Gaussian kernel

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2}{2\sigma^2}\right) \tag{5}$$

parameterized by σ. The Fourier transform of $k(\boldsymbol{r})$, coined *spectral density*, in a sense characterizes the decay of eigenvalues of the finitely dimensional covariance matrix K (Stein, 1999; Chen, 2013). The decay is known to be the fastest among the Matérn class of kernels (Stein, 1999; Rasmussen and Williams, 2006; Chilès and Delfiner, 2012), where Gaussian being a special case is the smoothest. Additionally, the range parameter σ also affects the decay. When $\sigma \to \infty$, K tends to a rank-1 matrix; whereas when $\sigma \to 0$, K tends to the identity. The numerical rank of the matrix varies when σ moves between the two extremes. The decay of eigenvalues plays an important role on the effectiveness of the approximate kernels discussed below.

1.2 Approximate Kernels

The first approach is low-rank kernels. Examples are Nyström approximation (Williams and Seeger, 2000; Schölkopf and Smola, 2001; Drineas and Mahoney, 2005), random Fourier features (Rahimi and Recht, 2007), and variants (Yang et al., 2014). The Nyström approximation is based on a set of landmark points, $\underline{X}$, randomly sampled from the training data X. Then, the kernel can be written as

$$k_{\text{Nyström}}(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \underline{X})K(\underline{X}, \underline{X})^{-1}k(\underline{X}, \boldsymbol{x}'). \tag{6}$$

For the convenience of deriving approximation bounds, Drineas and Mahoney (2005) consider sampling with repetition, which makes $\underline{X}$ possibly a multiset and the matrix inverse

in (6) necessarily replaced by a pseudo inverse. Various approaches for choosing the landmark points were compared in Zhang and Kwok (2010). For random Fourier features, let $\hat{k}(\boldsymbol{\omega})$ be the Fourier transform of $k(\boldsymbol{r})$ and let $\hat{k}$ be normalized such that it integrates to unity; that is, $\hat{k}$ is the normalized spectral density of k. Then, the kernel is

$$k_{\text{Fourier}}(\boldsymbol{x}, \boldsymbol{x}') = \frac{2}{r}\sum_{i=1}^{r} \cos(\boldsymbol{\omega}_i^T \boldsymbol{x} + b_i)\cos(\boldsymbol{\omega}_i^T \boldsymbol{x}' + b_i), \tag{7}$$

where r is the rank, and b_i and $\boldsymbol{\omega}_i$ are iid samples of Uniform$(0, 2\pi)$ and of a distribution with density $\hat{k}$, respectively. Note that (6) applies to any kernel whereas (7) applies to only stationary ones.

The Nyström approximation admits a conditional interpretation in the context of GP. The covariance kernel (6) can be equivalently written as

$$k_{\text{Nyström}}(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') - k(\boldsymbol{x}, \boldsymbol{x}'|\underline{X}),$$

where

$$k(\boldsymbol{x}, \boldsymbol{x}'|\underline{X}) = k(\boldsymbol{x}, \boldsymbol{x}') - k(\boldsymbol{x}, \underline{X})K(\underline{X}, \underline{X})^{-1}k(\underline{X}, \boldsymbol{x}')$$

is nothing but the covariance of $\boldsymbol{x}$ and $\boldsymbol{x}'$ conditioned on $\underline{X}$. In other words, the covariance kernel of Nyström approximation comes from a deduction of the original covariance by a conditional covariance. The conditional covariance for any $\boldsymbol{x}$ (or symmetrically, $\boldsymbol{x}'$) within $\underline{X}$ vanishes and hence the approximation is lossless. Intuitively speaking, the closer the sites are to $\underline{X}$, the smaller the loss is. This explains frequent observations that when the size of the set $\underline{X}$ is small, using the centers of a k-means clustering as the landmark points often improves the approximation (Zhang et al., 2008; Yang et al., 2012). A caveat is that the time cost of performing k-means clustering is often much higher than that of the Nyström calculation itself. Hence, the improvement gained from clustering may not be as significant as that from increasing the size of the conditioned set $\underline{X}$. When $\underline{X}$ is large, the improvement brought about by clustering is less significant (see, e.g., Rasmussen and Williams (2006, Section 8.3.7)).

A limitation of the low-rank kernels is that the size of the conditioned set, or equivalently the rank, needs to correlate with the decay of the spectrum of K in order to yield a good approximation. For a slow decay, it is not rare to see in practice that the rank r grows to thousands (Yen et al., 2014) or even several hundred thousands (Huang et al., 2014; Avron and Sindhwani, 2016) in order to yield comparable results with other methods, for a data set of size on the order of millions. See also the experimental results in Section 5.

The second approach is a cross-domain independent kernel. Simply speaking, the kernel matrix is approximated by keeping only the diagonal blocks of the matrix. In a GP language, we partition the domain S into m sub-domains S_j, $j = 1, \ldots, m$, and make an independence assumption across sub-domains. Then, the covariance between $\boldsymbol{x}$ and $\boldsymbol{x}'$ vanishes when the two sites come from different sub-domains. That is,

$$k_{\text{independent}}(\boldsymbol{x}, \boldsymbol{x}') = \begin{cases} k(\boldsymbol{x}, \boldsymbol{x}'), & \text{if } \boldsymbol{x}, \boldsymbol{x}' \in S_j \text{ for some } j, \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

Whereas such a kernel appears ad hoc and associated theory is possibly limited, the scenarios when it exhibits superiority over a low-rank kernel of comparable sizes are not rare (see the

likelihood comparison in Stein (2014) and also the experimental results in Section 5). An intuitive explanation exists in the context of classification. The cross-domain independent kernel works well when geographically nearby points possess a majority of the signal for classifying points within the domain. This happens more often when the kernel has a reasonably centralized bandwidth (outside of which the kernel value becomes marginal). In such a case, nearby points are the most influential.

The third approach is covariance tapering (Furrer et al., 2006; Kaufman et al., 2008). It amounts to defining a new kernel by multiplying the original kernel k with a compactly supported kernel k_{compact}:

$$k_{\text{taper}}(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') \cdot k_{\text{compact}}(\boldsymbol{x}, \boldsymbol{x}').$$

The tapered kernel matrix is an elementwise product of two positive-definite matrices; hence, it is positive-definite, too (Horn and Johnson, 1994, Theorem 5.2.1). The primary motivation of this kernel is to introduce sparsity to the matrix. The supporting theory is drawn on the confidence interval (cf. (4)) rather than on the prediction (3). It is cast in the setting of fixed-domain asymptotics, which is similar to a usual practice in machine learning—a prescaling of each attribute to within a finite interval. The theory hints that if the spectral density of k_{compact} has a lighter tail (i.e., the spectrum of the corresponding kernel matrix decays faster) than that of k, then the ratio between the prediction variance by using the tapered kernel k_{taper} and that by using the original kernel k tends to a finite limit, as the number of training data increases to infinity in the domain. The theory holds a guarantee on the prediction confidence if we choose k_{compact} judiciously. Tapering is generally applicable to heavy-tailed kernels (e.g., Matérn kernels with low smoothness) rather than light-tailed kernels such as the Gaussian. Nevertheless, a drawback of this approach is similar to the one we stated earlier for using a compactly supported kernel alone: the range of the support must be sufficiently small for sparse linear algebra to be efficient.

1.3 Proposed Kernel

In this paper, we propose a novel approach for constructing approximate kernels motivated by low-rank kernels and cross-domain independent kernels. The construction aims at deriving a kernel that (a) maintains the (strict) positive-definiteness, (b) leverages the advantages of low-rank and independent approaches, (c) facilitates the evaluation of the kernel matrix $K(X, X)$ and the out-of-sample extension $k(X, \boldsymbol{x})$, and (d) admits fast algorithms for a variety of matrix operations. The premise of the idea is a hierarchical partitioning of the data domain and a recursive approximation across the hierarchy. Space partitioning is a frequently encountered idea for kernel matrix approximations (Si et al., 2014; March et al., 2014; Yu et al., 2017; Si et al., 2017), but maintaining positive definiteness is quite challenging. Moreover, when the approximation is performed in a hierarchical fashion and is focused on only the matrix, it is not always easy to generalize to out of samples. A particularly intriguing property of the approach proposed in this article is that both positive definiteness and out-of-sample extensions are guaranteed, because the construction acts on the kernel function itself.

2. Hierarchically Compositional Kernel

The low-rank kernel (in particular, the Nyström approximation $k_{\text{Nyström}}$) and the cross-domain independent kernel $k_{\text{independent}}$ are complementary to each other in the following sense: the former acts on the global space, where the covariance at every pair of points $\boldsymbol{x}$ and $\boldsymbol{x}'$ are deducted by a conditional covariance based on the conditioned set $\underline{X}$ chosen globally; whereas the latter preserves all the local information but completely ignores the interrelationship outside the local domain. We argue that an organic composition of the two will carry both advantages and alleviate the shortcomings. Further, a hierarchical composition may reduce the information loss in nearby local domains.

2.1 Composition of Low-Rank Kernel with Cross-Domain Independent Kernel

Let the domain S be partitioned into disjoint sub-domains $\bigcup S_j = S$. Let $\underline{X}$ be a set of landmark points in S. For generality, $\underline{X}$ needs not be a subset of the training data X. Consider the function

$$k_{\text{compositional}}(\boldsymbol{x}, \boldsymbol{x}') = \begin{cases} k(\boldsymbol{x}, \boldsymbol{x}'), & \text{if } \boldsymbol{x}, \boldsymbol{x}' \in S_j \text{ for some } j, \\ k(\boldsymbol{x}, \underline{X}) K(\underline{X}, \underline{X})^{-1} k(\underline{X}, \boldsymbol{x}'), & \text{otherwise.} \end{cases}$$

Clearly, $k_{\text{compositional}}$ leverages both (6) and (8). When two points $\boldsymbol{x}$ and $\boldsymbol{x}'$ are located in the same domain, they maintain the full covariance (8); whereas when they are located in separate domains, their covariance comes from the low-rank kernel $k_{\text{Nyström}}$ (6). Such a composition complements missing information across domains in $k_{\text{independent}}$ and also complements the information loss in local domains caused by the Nyström approximation. The following result is straightforward in light of the fact that if $\boldsymbol{x} \in \underline{X}$, then $K(\underline{X}, \underline{X})^{-1} k(\underline{X}, \boldsymbol{x})$ is a column of the identity matrix where the only nonzero element (i.e., 1) is located with respect to the location of $\boldsymbol{x}$ inside $\underline{X}$.

Proposition 1 *We have*

$$k_{compositional}(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}'),$$

if $\boldsymbol{x}, \boldsymbol{x}' \in S_j$ for some j, or if either of $\boldsymbol{x}, \boldsymbol{x}'$ belongs to $\underline{X}$.

An alternative view of the kernel $k_{\text{compositional}}$ is that it is an additive combination of a globally low-rank approximation and local Schur complements within each sub-domain. Hence, the kernel is (strictly) positive-definite. See Lemma 2 and Theorem 3 in the following.

Lemma 2 *The Schur-complement function*

$$k_{Schur}(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') - k(\boldsymbol{x}, \underline{X}) K(\underline{X}, \underline{X})^{-1} k(\underline{X}, \boldsymbol{x}')$$

is positive-definite, if k is strictly positive-definite, or if k is positive-definite and $K(\underline{X}, \underline{X})$ is invertible.

Proof For any set X, let $Y = X \cup \underline{X}$. It amounts to showing that the corresponding kernel matrix $K_{\text{Schur}}(Y, Y)$ is positive semi-definite; then, $K_{\text{Schur}}(X, X)$ as a principal submatrix is also positive semi-definite.

Denote by $\underline{X}^c = Y \backslash \underline{X}$, which could possibly be empty, and let the points in $\underline{X}$ be ordered before those in $\underline{X}^c$. Then,

$$K_{\text{Schur}}(Y,Y) = \begin{bmatrix} 0 & 0 \\ 0 & K(\underline{X}^c,\underline{X}^c) - K(\underline{X}^c,\underline{X})K(\underline{X},\underline{X})^{-1}K(\underline{X},\underline{X}^c) \end{bmatrix}.$$

By the law of inertia, the matrices

$$\begin{bmatrix} K(\underline{X},\underline{X}) & K(\underline{X},\underline{X}^c) \\ K(\underline{X}^c,\underline{X}) & K(\underline{X}^c,\underline{X}^c) \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} K(\underline{X},\underline{X}) & 0 \\ 0 & K(\underline{X}^c,\underline{X}^c) - K(\underline{X}^c,\underline{X})K(\underline{X},\underline{X})^{-1}K(\underline{X},\underline{X}^c) \end{bmatrix}$$

have the same number of positive, zero, and negative eigenvalues, respectively. If k is strictly positive-definite, then the eigenvalues of both matrices are all positive. If k is positive-definite, then the eigenvalues of both matrices are all nonnegative. In both cases, the Schur-complement matrix $K(\underline{X}^c,\underline{X}^c) - K(\underline{X}^c,\underline{X})K(\underline{X},\underline{X})^{-1}K(\underline{X},\underline{X}^c)$ is positive semi-definite and thus so is $K_{\text{Schur}}(Y,Y)$. ■

Theorem 3 *The function $k_{compositional}$ is positive-definite if k is positive-definite and $K(\underline{X},\underline{X})$ is invertible. Moreover, $k_{compositional}$ is strictly positive-definite if k is so.*

Proof Write $k_{\text{compositional}} = k_1 + k_2$, where $k_1(\boldsymbol{x},\boldsymbol{x}') = k(\boldsymbol{x},\underline{X})K(\underline{X},\underline{X})^{-1}k(\underline{X},\boldsymbol{x}')$ and

$$k_2(\boldsymbol{x},\boldsymbol{x}') = \begin{cases} k(\boldsymbol{x},\boldsymbol{x}') - k(\boldsymbol{x},\underline{X})K(\underline{X},\underline{X})^{-1}k(\underline{X},\boldsymbol{x}'), & \boldsymbol{x},\boldsymbol{x}' \in S_j \text{ for some } j, \\ 0, & \text{otherwise.} \end{cases}$$

Clearly, k_1 is positive-definite; by Lemma 2, k_2 is so, too. Thus, $k_{\text{compositional}}$ is positive-definite.

We next show the strict definiteness when k is strictly positive-definite. That is, for any set of points $\{\boldsymbol{x}_i\}$ and any set of coefficients $\{\alpha_i\}$ that are not all zero, the bilinear form $\sum_{il}\alpha_i\alpha_l k_{\text{compositional}}(\boldsymbol{x}_i,\boldsymbol{x}_l)$ cannot be zero. Note that $k_2(\boldsymbol{x},\boldsymbol{x}') = 0$ whenever $\boldsymbol{x}$ or $\boldsymbol{x}' \in \underline{X}$. Moreover, we have seen in the proof of Lemma 2 that the Schur-complement matrix $K(\underline{X}^c,\underline{X}^c) - K(\underline{X}^c,\underline{X})K(\underline{X},\underline{X})^{-1}K(\underline{X},\underline{X}^c)$ is positive-definite when k is strictly positive-definite. Therefore, we have $\sum_{il}\alpha_i\alpha_l k_2(\boldsymbol{x}_i,\boldsymbol{x}_l) = 0$ only when $\alpha_i = 0$ for all i satisfying $\boldsymbol{x}_i \notin \underline{X}$. In such a case,

$$\sum_{il}\alpha_i\alpha_l k_1(\boldsymbol{x}_i,\boldsymbol{x}_k) = \sum_{\boldsymbol{x}_i,\boldsymbol{x}_l \in \underline{X}} \alpha_i\alpha_l k(\boldsymbol{x}_i,\underline{X})K(\underline{X},\underline{X})^{-1}k(\underline{X},\boldsymbol{x}_l).$$

Because of the strict positive-definiteness of k, the above summation cannot be zero if any of the involved α_i (that is, those satisfying $\boldsymbol{x}_i \in \underline{X}$) is nonzero. Then, $\sum_{il}\alpha_i\alpha_l[k_1(\boldsymbol{x}_i,\boldsymbol{x}_l) + k_2(\boldsymbol{x}_i,\boldsymbol{x}_l)] = 0$ only when all α_i are zero. ■

Since the composition replaces the Nyström approximation in local domains by the full covariance, it bares no surprise that $k_{\text{compositional}}$ improves over $k_{\text{Nyström}}$ in terms of matrix approximation.

Theorem 4 *Given a set $\underline{X}$ of landmark points and for any set $X \neq \underline{X}$,*

$$\|K(X,X) - K_{compositional}(X,X)\| < \|K(X,X) - K_{Nyström}(X,X)\|,$$

where $\|\cdot\|$ is the 2-norm or the Frobenius norm.

Proof Based on the split $k_{\text{compositional}} = k_1 + k_2$ in the proof of Theorem 3, one easily sees that

$$K - K_{\text{compositional}} = (K - K_{\text{Nyström}}) - \text{block-diag}(K - K_{\text{Nyström}}),$$

where block-diag means keeping only the diagonal blocks of a matrix. Denote by $A = K - K_{\text{Nyström}}$ and $D = \text{block-diag}(A)$. In what follows we show that

$$\|A - D\| < \|A\|. \tag{9}$$

Because $A = K(X,X) - K(X,\underline{X})K(\underline{X},\underline{X})^{-1}K(\underline{X},X)$ is positive semi-definite and nonzero, its diagonal cannot be zero. Then, eliminating the block-diagonal part D reduces the Frobenius norm. Thus, (9) holds for the Frobenius norm. To see that (9) also holds for the 2-norm, let $Y = X\backslash\underline{X}$ and let the points in Y be ordered before those in $X\backslash Y$. Then,

$$A = \begin{bmatrix} K(Y,Y) - K(Y,\underline{X})K(\underline{X},\underline{X})^{-1}K(\underline{X},Y) & 0 \\ 0 & 0 \end{bmatrix}.$$

Because the zero rows and columns do not contribute to the 2-norm, and because the top-left block of A is positive-definite, it suffices to prove (9) for any positive-definite matrix A.

Note the following two straightforward inequalities

$$\lambda_{\min}(A - D) \geq \lambda_{\min}(A) - \lambda_{\max}(D), \tag{10}$$
$$\lambda_{\max}(A - D) \leq \lambda_{\max}(A) - \lambda_{\min}(D). \tag{11}$$

Because D consists of the diagonal blocks of A, the interlacing theorem of eigenvalues states that for each diagonal block D_i, we have

$$\lambda_{\min}(A) \leq \lambda_{\min}(D_i) \leq \lambda_{\max}(D_i) \leq \lambda_{\max}(A).$$

Then, taking the max/min eigenvalues of all blocks, we obtain

$$\lambda_{\min}(A) \leq \lambda_{\min}(D) \leq \lambda_{\max}(D) \leq \lambda_{\max}(A). \tag{12}$$

Substituting (12) into (10) and (11), together with $\lambda_{\min}(A) > 0$, we obtain

$$\lambda_{\min}(A - D) > -\lambda_{\max}(A) \quad \text{and} \quad \lambda_{\max}(A - D) < \lambda_{\max}(A),$$

which immediately implies that $\|A - D\|_2 < \|A\|_2$. ■

2.2 Hierarchical Composition

While $k_{\text{compositional}}$ maintains the full information inside each domain S_j, the information loss across domains caused by the low-rank approximation may still be dramatic. Consider the scenario of a large number of disjoint domains S_j; such a scenario is necessarily typical for the purpose of reducing the computational cost. If each domain is adjacent to only a few neighboring domains, it is possible to reduce the information loss in nearby domains.

The idea is to form a hierarchy. Let us first take a two-level hierarchy for example. A few of the neighboring domains S_j form a super-domain $S_J = \bigcup_{j\in J} S_j$. These super-domains are formed such that they are disjoint and they collectively partition the whole domain; i.e., $\bigcup S_J = S$. Under such a hierarchical formation, instead of using landmark points $\underline{X}$ in S to define the covariance across the bottom-level domains S_j, we may use landmark points $\underline{X}_J$ chosen from the super-domain S_J to define the covariance. The intuition is that the conditional covariance $k(\boldsymbol{x}, \boldsymbol{x}'|\underline{X}_J)$ for $\boldsymbol{x}, \boldsymbol{x}' \in S_J$ tends to be smaller than $k(\boldsymbol{x}, \boldsymbol{x}'|\underline{X})$, because $\boldsymbol{x}$ and $\boldsymbol{x}'$ are geographically closer to $\underline{X}_J$ than to $\underline{X}$. Then, the information loss is reduced for points inside the same super-domain S_J.

To formalize this idea, we consider an arbitrary hierarchy, which is represented by a rooted tree T. See Figure 1 for an example. The root node 1 is associated with the whole domain $S =: S_1$. Each nonleaf node i possesses a set of children $Ch(i)$; correspondingly, the associated domain S_i is partitioned into disjoint sub-domains S_j satisfying $\bigcup_{j\in Ch(i)} = S_i$. The partitioning tree T is almost the most general rooted tree, except that no nodes in T have exactly one child.

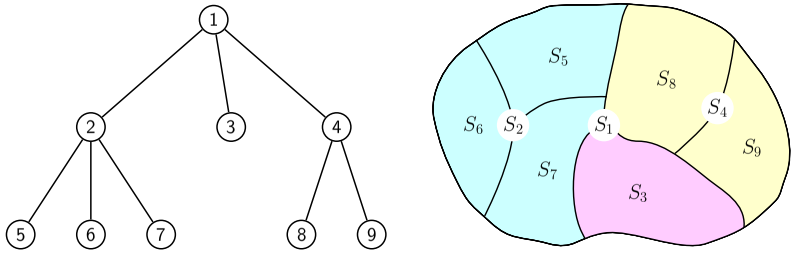


Figure 1: Partitioning tree T and the domain S.

Each nonleaf node i is associated with a set $\underline{X}_i$ of landmark points, located within the domain S_i. We now recursively define the kernel on domains across levels. Continuing the example of Figure 1, node 4 has two children 8 and 9. Since these two children are leaf nodes, the covariance within S_8 (or S_9) comes from the original kernel k, whereas the covariance across S_8 and S_9 comes from the Nyström approximation by using landmark points $\underline{X}_4$. That is, the kernel is equal to $k(\boldsymbol{x}, \boldsymbol{x}')$ if $\boldsymbol{x}$ and $\boldsymbol{x}'$ are both in S_8 (or in S_9), and equal to $k(\boldsymbol{x}, \underline{X}_4)K(\underline{X}_4, \underline{X}_4)^{-1}k(\underline{X}_4, \boldsymbol{x}')$ if they are in S_8 and S_9 separately. Such a covariance bares less information loss caused by the conditioned set, compared with the use of landmark points located within the whole domain S.

Next, consider the covariance between child domains of S_2 and those of S_4 (say, S_6 and S_9, respectively). At a first glance, we could have used $k(\boldsymbol{x}, \underline{X}_1)K(\underline{X}_1, \underline{X}_1)^{-1}k(\underline{X}_1, \boldsymbol{x}')$ to define the kernel, because $\underline{X}_1$ consists of landmark points located in the domain that covers both S_6 and S_9. However, such a definition cannot guarantee the positive-definiteness of the overall kernel. Instead, we approximate $k(\boldsymbol{x}, \underline{X}_1)$ by using the Nyström approximation $k(\boldsymbol{x}, \underline{X}_2)K(\underline{X}_2, \underline{X}_2)^{-1}K(\underline{X}_2, \underline{X}_1)$ based on the landmark points $\underline{X}_2$. Then, the covariance for $\boldsymbol{x} \in S_6$ and $\boldsymbol{x}' \in S_9$ is defined as

$$\Big[k(\boldsymbol{x}, \underline{X}_2)K(\underline{X}_2, \underline{X}_2)^{-1}K(\underline{X}_2, \underline{X}_1)\Big]K(\underline{X}_1, \underline{X}_1)^{-1}\Big[K(\underline{X}_2, \underline{X}_1)K(\underline{X}_2, \underline{X}_2)^{-1}k(\underline{X}_2, \boldsymbol{x}')\Big].$$

Formally, for a leaf node j and $\boldsymbol{x}, \boldsymbol{x}' \in S_j$, define $k^{(j)}(\boldsymbol{x}, \boldsymbol{x}') \equiv k(\boldsymbol{x}, \boldsymbol{x}')$. For a nonleaf node i and $\boldsymbol{x}, \boldsymbol{x}' \in S_i$, define

$$k^{(i)}(\boldsymbol{x}, \boldsymbol{x}') := \begin{cases} k^{(j)}(\boldsymbol{x}, \boldsymbol{x}'), & \text{if } \boldsymbol{x}, \boldsymbol{x}' \in S_j \text{ for some } j \in Ch(i), \\ \psi^{(i)}(\boldsymbol{x}, \underline{X}_i)K(\underline{X}_i, \underline{X}_i)^{-1}\psi^{(i)}(\underline{X}_i, \boldsymbol{x}'), & \text{otherwise,} \end{cases} \tag{13}$$

where if j is a child of i and if $\boldsymbol{x} \in S_j$, then

$$\psi^{(i)}(\boldsymbol{x}, \underline{X}_i) := \begin{cases} k(\boldsymbol{x}, \underline{X}_i), & \text{if } j \text{ is a leaf node,} \\ \psi^{(j)}(\boldsymbol{x}, \underline{X}_j)K(\underline{X}_j, \underline{X}_j)^{-1}K(\underline{X}_j, \underline{X}_i), & \text{otherwise.} \end{cases} \tag{14}$$

The $k^{(i)}$ at the root level gives the *hierarchically compositional kernel* of this paper:

$$k_{\text{hierarchical}} := k^{(\text{root})}. \tag{15}$$

Clearly, the kernel $k_{\text{compositional}}$ in Section 2.1 is a special case of $k_{\text{hierarchical}}$ when the partitioning tree consists of only the root and the leaf nodes (which are children of the root).

Expanding the recursive formulas (13) and (14), for two distinct leaf nodes j and l, we see that the covariance between $\boldsymbol{x} \in S_j$ and $\boldsymbol{x}' \in S_l$ is

$$\begin{aligned} &k_{\text{hierarchical}}(\boldsymbol{x}, \boldsymbol{x}') = k^{(r)}(\boldsymbol{x}, \boldsymbol{x}') \\ &= \underbrace{k(\boldsymbol{x}, \underline{X}_{j_1})K(\underline{X}_{j_1}, \underline{X}_{j_1})^{-1}K(\underline{X}_{j_1}, \underline{X}_{j_2})\cdots K(\underline{X}_{j_s}, \underline{X}_{j_s})^{-1}K(\underline{X}_{j_s}, \underline{X}_r)}_{\psi^{(r)}(\boldsymbol{x}, \underline{X}_r)} K(\underline{X}_r, \underline{X}_r)^{-1} \\ &\qquad \cdot \underbrace{K(\underline{X}_r, \underline{X}_{l_t})K(\underline{X}_{l_t}, \underline{X}_{l_t})^{-1}\cdots K(\underline{X}_{l_2}, \underline{X}_{l_1})K(\underline{X}_{l_1}, \underline{X}_{l_1})^{-1}k(\underline{X}_{l_1}, \boldsymbol{x}')}_{\psi^{(r)}(\underline{X}_r, \boldsymbol{x}')}, \end{aligned} \tag{16}$$

where r is the least common ancestor of j and l, and $(j, j_1, j_2, \ldots, j_s, r)$ and $(l, l_1, l_2, \ldots, l_t, r)$ are the paths connecting r and the two leaf nodes, respectively. Therefore, we have the following result.

Proposition 5 *Based on the notation in the preceding paragraph, for $\boldsymbol{x} \in S_j$ and $\boldsymbol{x}' \in S_l$, we have*

$$k_{hierarchical}(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}'),$$

whenever $\boldsymbol{x} \in \underline{X}_j \cap \underline{X}_{j_1} \cdots \cap \underline{X}_{j_s}$, $\boldsymbol{x}' \in \underline{X}_l \cap \underline{X}_{l_1} \cdots \cap \underline{X}_{l_t}$, and either of $\boldsymbol{x}, \boldsymbol{x}'$ belongs to $\underline{X}_r$.

Proof On (16), recursively apply the fact that if $\boldsymbol{x} \in \underline{X}$, then $K(\underline{X}, \underline{X})^{-1} k(\underline{X}, \boldsymbol{x})$ is a column of the identity matrix where the only nonzero element (i.e., 1) is located with respect to the location of $\boldsymbol{x}$ inside $\underline{X}$. ■

The following theorem guarantees the validity of the kernel. Its proof strategy is similar to that of Theorem 3, but it is complex because of recursion. We defer the proof to Appendix A.

Theorem 6 *The function $k_{hierarchical}$ is positive-definite if k is positive-definite and $K(\underline{X}_i, \underline{X}_i)$ is invertible for all sets of landmark points $\underline{X}_i$ associated with the nonleaf nodes i. Moreover, $k_{hierarchical}$ is strictly positive-definite if k is so.*

3. Matrix View

The kernel matrix $K_{\text{hierarchical}}(X, X)$ for a set of training points X exhibits a hierarchical block structure. Figure 2 pictorially shows such a structure for the example in Figure 1. To avoid degenerate empty blocks, we assume that $X \cap S_j \neq \emptyset$ for all leaf nodes j in the partitioning tree T.

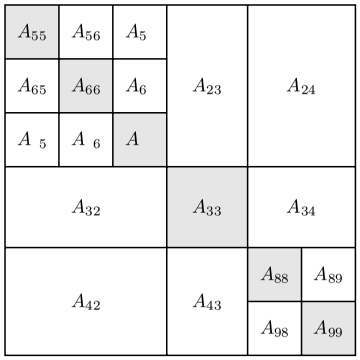


Figure 2: The matrix $K_{\text{hierarchical}}$ corresponding to the partitioning tree T in Figure 1.

Formally, for any node i, let $X_i = X \cap S_i$; that is, X_i consists of the training points that fall within the domain S_i. Then, define a matrix $A \in \mathbb{R}^{n \times n}$ with the following structure:

1. For every node i, A_{ii} is a diagonal block whose rows and columns correspond to X_i; for every pair of sibling nodes i and j, A_{ij} is an off-diagonal block whose rows correspond to X_i and columns to X_j.
2. For every leaf node i, $A_{ii} = K(X_i, X_i)$.
3. For every pair of sibling nodes i and j, $A_{ij} = U_i \Sigma_p U_j^T$, where p is the parent of i and j, with Σ_p and U_i defined next.

4. For every nonleaf node p, $\Sigma_p = K(\underline{X}_p, \underline{X}_p)$.
5. For every leaf node i, $U_i = K(X_i, \underline{X}_p) K(\underline{X}_p, \underline{X}_p)^{-1}$, where p is the parent of i.
6. For every pair of child node i and parent node p not being the root, the block of U_p corresponding to node i is $U_i W_p$, where $W_p = K(\underline{X}_p, \underline{X}_r) K(\underline{X}_r, \underline{X}_r)^{-1}$ and r is the parent of p. That is, in the matrix form,

$$U_p = \begin{bmatrix} \vdots \\ U_i \\ \vdots \end{bmatrix}_{i \in Ch(p)} \cdot W_p.$$

One sees that A is exactly equal to $K_{\text{hierarchical}}(X, X)$ by verifying against the definition of $k_{\text{hierarchical}}$ in (13)–(15).

Such a hierarchical block structure is a special case of the *recursively low-rank compressed matrix* studied in Chen (2014b). In this matrix, off-diagonal blocks are recursively compressed into low rank through change of basis, while the main-diagonal blocks at the leaf level remain intact. Its connections and distinctions with related matrices (e.g., FMM matrices (Sun and Pitsianis, 2001) and hierarchical matrices (Hackbusch, 1999)) were discussed in detail in Chen (2014b). The signature of a recursively low-rank compressed matrix is that many matrix operations can be performed with a cost, loosely speaking, linear in n. The matrix structure in this paper, which results from the kernel $k_{\text{hierarchical}}$, specializes a general recursively low-rank compressed matrix in the following aspects: (a) the matrix is symmetric; (b) the middle factor Σ_p in each A_{ij} is the same for all child pairs i, j of p; and (c) the change-of-basis factor W_p is the same for all children i of p. Hence, storage and time costs of matrix operations are reduced by a constant factor compared with those of the algorithms in Chen (2014b). In what follows, we discuss the algorithmic aspects of the matrix operations needed to carry out kernel method computations, but defer the complexity analysis in Section 4. Discussions of additional matrix operations useful in a general machine learning context are made toward the end of the paper.

3.1 Matrix-Vector Multiplication

First, we consider computing the matrix-vector product $y = Ab$. Let the blocks of a vector be labeled in the same manner as those of the matrix A. That is, for any node i, b_i denotes the part of b that corresponds to the point set X_i. Then, the vector y is clearly an accumulation of the smaller matrix-vector products $A_{ij} b_j$ in the appropriate blocks, for all sibling pairs i, j and for all leaf nodes $i = j$ (cf. Figure 2). When $i = j$ is a leaf node, the computation of $A_{ij} b_j$ is straightforward. On the other hand, when i and j constitute a pair of sibling nodes, we consider a descendant leaf node l of i. The block of $A_{ij} b_j$ corresponding to node l admits an expanded expression

$$U_l W_{l_1} W_{l_2} \cdots W_{l_s} W_i \Sigma_p W_j^T \left(\sum_{q_t \in Ch(j)} W_{q_t}^T \cdots \left(\sum_{q_2 \in Ch(q_3)} W_{q_2}^T \left(\sum_{q_1 \in Ch(q_2)} W_{q_1}^T \left(\sum_{q \in Ch(q_1)} U_q^T b_q \right) \right) \right) \right), \tag{17}$$

where $(l, l_1, l_2, \ldots, l_s, i, p)$ is the path connecting l and the parent p of i. This expression assumes that all the leaf descendants of j are on the same level so that the expression is not overly complex; but the subsequent reasoning applies to the general case. The terms inside the nested parentheses of (17) motivate the following recursive definition:

$$c_i = \begin{cases} U_i^T b_i, & i \text{ is a leaf node,} \\ W_i^T \sum_{j \in Ch(i)} c_j, & \text{otherwise.} \end{cases}$$

Clearly, all c_j's can be computed within one pass of a post-order tree traversal. Upon the completion of the traversal, we rewrite (17) as $U_l W_{l_1} W_{l_2} \cdots W_{l_s} W_i \Sigma_p c_j$. Then we note that for any leaf node l, y_l is a sum of these expressions with all p along the path connecting l and the root, and additionally, of $A_{ll} b_l$. In other words, we have

$$y_l = A_{ll} b_l + \left(\sum_{l' \in Ch(l_1) \setminus \{l\}} U_l \Sigma_{l_1} c_{l'} \right) + \left(\sum_{l' \in Ch(l_2) \setminus \{l_1\}} U_l W_{l_1} \Sigma_{l_2} c_{l'} \right) + \cdots + \left(\sum_{l' \in Ch(\text{root}) \setminus \{l_t\}} U_l W_{l_1} W_{l_2} \cdots W_{l_t} \Sigma_{\text{root}} c_{l'} \right),$$

where $(l, l_1, l_2, \ldots, l_t, \text{root})$ is the path connecting l and the root. Therefore, we recursively define another quantity

$$d_j = W_i d_i + \sum_{j' \in Ch(i) \setminus \{j\}} \Sigma_i c_{j'}$$

for all nonroot nodes j with parent i. Clearly, all d_j's can also be computed within one pass of a pre-order tree traversal. Upon the completion of this second traversal, we have $y_l = A_{ll} b_l + U_l d_l$, which concludes the computation of the whole vector y.

As such, computing the matrix-vector product $y = Ab$ consists of one post-order tree traversal, followed by a pre-order one. We refer the reader to Chen (2014b) for a full account of the computational steps. For completeness, we summarize the pesudocode in Algorithm 1 as a reference for computer implementation.

3.2 Matrix Inversion

Next, we consider computing A^{-1}, for which we use the tilded notation $\tilde{A} := A^{-1}$. One can show (Chen, 2014b) that $\tilde{A}$ has exactly the same hierarchical block structure as does A. In other words, the structure of $\tilde{A}$ can be described by reusing the verbatim at the beginning of Section 3, with the factors A_{ii}, A_{ij}, U_i, Σ_p, W_p replaced by the tilded version $\tilde{A}_{ii}$, $\tilde{A}_{ij}$, $\tilde{U}_i$, $\tilde{\Sigma}_p$, $\tilde{W}_p$, respectively. The proof is both inductive and constructive, providing explicit formulas to compute $\tilde{A}_{ii}$, $\tilde{A}_{ij}$, $\tilde{U}_i$, $\tilde{\Sigma}_p$, $\tilde{W}_p$ level by level. The essential idea is that if $(A_{ii} - U_i \Sigma_p U_i^T)^{-1} = \tilde{A}_{ii} - \tilde{U}_i \tilde{\Sigma}_p \tilde{U}_i^T$ have been computed for all children i of a node p, and if r is the parent of p, then

$$A_{pp} - U_p \Sigma_r U_p^T = \begin{bmatrix} \ddots & & \\ & A_{ii} - U_i \Sigma_p U_i^T & \\ & & \ddots \end{bmatrix} + \begin{bmatrix} \vdots \\ U_i \\ \vdots \end{bmatrix} \Sigma_p \begin{bmatrix} \cdots & U_i^T & \cdots \end{bmatrix} - U_p \Sigma_r U_p^T.$$

Algorithm 1 Computing $y = Ab$

```
Initialize c_i ← 0, d_i ← 0 for each nonroot node i of the tree
Upward(root)
Downward(root)

function Upward(i)
    if i is leaf then
        c_i ← U_i^T b_i;   y_i ← A_ii b_i
    else
        for all children j of i do
            Upward(j)
            c_i ← c_i + W_i^T c_j, if i is not root
        end for
    end if
    if i is not root then
        for all siblings l of i do d_l ← d_l + Σ_p c_i end for        ▷ p is parent of i
    end if
end function

function Downward(i)
    if i is leaf then y_i ← y_i + U_i d_i and return end if
    for all children j of i do
        d_j ← d_j + W_i d_i, if i is not root
        Downward(j)
    end for
end function
```

Hence, the inversion of $A_{pp} - U_p \Sigma_r U_p^T$ can be done by applying the Sherman–Morrison–Woodbury formula, which results in a form $\tilde{A}_{pp} - \tilde{U}_p \tilde{\Sigma}_r \tilde{U}_p^T$, where $\tilde{W}_r$ is determined based on the change of basis from $\tilde{U}_i$ to $\tilde{U}_p$ and the middle factor $\tilde{\Sigma}_r$ is also determined. In addition, the factors $\tilde{\Sigma}_j$ (computed previously) for all descendants j of r must be corrected because the diagonal block of $\tilde{A}_{pp}$ corresponding to node j is a low-rank correction of the corresponding diagonal block of $\tilde{A}_{ii}$ with the same basis. Such a down-cascading correction occurs whenever the induction proceeds from one child level to the parent level; but computationally, the correction on any node can be accumulated during the whole induction process. The net result of the accumulation is that each $\tilde{\Sigma}_i$ needs be corrected only once, which ensures an efficient computation.

We refer the reader to Chen (2014b) for a full account of the computational steps. Similar to the matrix-vector multiplication, the overall computation here consists of one post-order tree traversal (corresponding to the induction), followed by a pre-order one (corresponding to the down-cascading correction). For completeness, we summarize the pseudocode in Algorithm 2 as a reference for computer implementation.

Algorithm 2 Computing $\tilde{A} = A^{-1}$

Upward(`root`)
Downward(`root`)

function Upward(i)
  if i is leaf **then**
    $\tilde{A}_{ii} \leftarrow (A_{ii} - U_i \Sigma_p U_i^T)^{-1}$; $\tilde{U}_i \leftarrow \tilde{A}_{ii} U_i$; $\tilde{\Theta}_i \leftarrow U_i^T \tilde{U}_i$ ▷ p is parent of i
    return
  end if
  for all children j of i **do**
    Upward(j)
    $\tilde{W}_j \leftarrow (I + \tilde{\Sigma}_j \tilde{\Xi}_j) W_j$ **if** j is not leaf
    $\tilde{\Theta}_j \leftarrow W_j^T \tilde{\Xi}_j \tilde{W}_j$ **if** j is not leaf
  end for
  $\tilde{\Xi}_i \leftarrow \sum_{j \in Ch(i)} \tilde{\Theta}_j$
  if i is not root **then** $\tilde{\Lambda}_i \leftarrow \Sigma_i - W_i \Sigma_p W_i^T$ **else** $\tilde{\Lambda}_i \leftarrow \Sigma_i$ **end if** ▷ p is parent of i
  $\tilde{\Sigma}_i \leftarrow -(I + \tilde{\Lambda}_i \tilde{\Xi}_i)^{-1} \tilde{\Lambda}_i$
  for all children j of i **do**
    $\tilde{E}_j \leftarrow \tilde{W}_j \tilde{\Sigma}_i \tilde{W}_j^T$ **if** j is not leaf
  end for
  $\tilde{E}_i \leftarrow 0$ if i is root
end function

function Downward(i)
  if i is leaf **then**
    $\tilde{A}_{ii} \leftarrow \tilde{A}_{ii} + \tilde{U}_i \tilde{\Sigma}_p \tilde{U}_i^T$ if i is not root ▷ p is parent of i
  else
    $\tilde{E}_i \leftarrow \tilde{E}_i + \tilde{W}_i \tilde{E}_p \tilde{W}_i^T$ if i is not root ▷ p is parent of i
    $\tilde{\Sigma}_i \leftarrow \tilde{\Sigma}_i + \tilde{E}_i$
    for all children j of i **do** Downward(j) **end for**
  end if
end function

3.3 (Implicit) Out-of-Sample Construction

Now, we consider the vector $k_{\text{hierarchical}}(X, \boldsymbol{x})$ for an existing training set X and a new point $\boldsymbol{x}$ in the testing set. Generally, this vector is not used alone but it appears in the inner product with some other vector w (see (2)). Whereas the construction of $k_{\text{hierarchical}}(X, \boldsymbol{x})$ can be done at $O(n)$ cost (details omitted here), we shall consider instead the computation of the inner product $w^T k_{\text{hierarchical}}(X, \boldsymbol{x})$, because the cost of computing this inner product is proportional to only the height of the tree per $\boldsymbol{x}$ after an $O(n)$ preprocessing. The

preprocessing cost is amortized on each $\boldsymbol{x}$ and thus is generally negligible for a large number of $\boldsymbol{x}$'s.

The computation of $w^T k_{\text{hierarchical}}(X, \boldsymbol{x})$ was not described in Chen (2014b), but the idea is similar to that of the matrix-vector multiplication in Section 3.1. To simplify notation, let $v \equiv k_{\text{hierarchical}}(X, \boldsymbol{x})$. Assume that $\boldsymbol{x}$ falls in the domain S_j for some leaf node j. Then, $v_j = k(X_j, \boldsymbol{x})$ and for any leaf node $l \neq j$,

$$v_l = U_l W_{l_1} W_{l_2} \cdots W_{l_s} \Sigma_p W_{j_t}^T \cdots W_{j_2}^T W_{j_1}^T K(\underline{X}_{j_1}, \underline{X}_{j_1})^{-1} k(\underline{X}_{j_1}, \boldsymbol{x}),$$

where p is the least common ancestor of l and j, and $(l, l_1, l_2, \ldots, l_s, p)$ and $(j, j_1, j_2, \ldots, j_t, p)$ are the paths connecting l and p, and j and p, respectively. Therefore, if we define

$$d_{j_u} = W_{j_u}^T d_{j_{u-1}}, \quad \cdots \quad d_{j_2} = W_{j_2}^T d_{j_1}, \quad d_{j_1} = W_{j_1}^T d_j, \quad d_j = K(\underline{X}_{j_1}, \underline{X}_{j_1})^{-1} k(\underline{X}_{j_1}, \boldsymbol{x}), \tag{18}$$

where $(j, j_1, j_2, \ldots, j_u, \text{root})$ is the path connecting j and the root, then we have

$$w^T v = w_j^T k(X_j, \boldsymbol{x}) + \sum_{l \neq j,\ l \text{ is leaf}} w_l^T U_l W_{l_1} W_{l_2} \cdots W_{l_s} \Sigma_p d_{j_t}. \tag{19}$$

If we further define

$$e_l = \begin{cases} U_l^T w_l, & l \text{ is leaf}, \\ \displaystyle\sum_{i \in Ch(l)} W_l^T e_i, & \text{otherwise}, \end{cases} \qquad \text{and} \qquad c_q = \Sigma_p^T e_l, \tag{20}$$

where p is the parent of the sibling pair q and l, then (19) is simplified to

$$w^T v = w_j^T k(X_j, \boldsymbol{x}) + \sum_{j_t \in \text{path}(j, \text{root})} c_{j_t}^T d_{j_t}. \tag{21}$$

Based on (18)–(21), we see that $w^T v \equiv w^T k_{\text{hierarchical}}(X, \boldsymbol{x})$ can be computed in the following two phases. In the first phase, we compute e_l and c_l for all nonleaf nodes l according to (20). Such a computation can be done by using a post-order tree traversal and is independent of $\boldsymbol{x}$. Thus, this computation is the preprocessing step. In the second phase, we compute d_{j_t} for all j_t along the path $(j, j_1, j_2, \ldots, j_u, \text{root})$ according to (18). Once they are computed, the summation (21) is straightforward and we thus conclude the computation. Note that the second phase is always conducted along a certain path connecting the root and the leaf j. As long as the determination of which leaf j the point $\boldsymbol{x}$ falls in is restricted on this path, the cost of the second phase is always asymptotically smaller than that of a tree traversal. We summarize the pseudocode in Algorithm 3.

4. Practical Considerations

Sections 2 and 3 provide a general framework for the interpretation of and the computation with the proposed kernel $k_{\text{hierarchical}}$; they, however, have not covered a range of practical issues. Some issues are dimension dependent (this paper concerns particularly dimensions higher than three, which dominates the use of related matrices, e.g., FMM and hierarchical matrices, in scientific computing); whereas others are numerically related. In this section, we discuss the handling of several of these issues and ensure that the proposed kernel is computationally efficient.

Algorithm 3 Computing $z = w^T k_{\text{hierarchical}}(X, \boldsymbol{x})$ for $\boldsymbol{x} \notin X$

Prefactorize $K(\underline{X}_p, \underline{X}_p)$ for all parents p of leaf nodes
Initialize $c_i \leftarrow 0$, $d_i \leftarrow 0$ for each nonroot node i of the tree
Common-Upward(`root`)
▷ The above three steps are independent of $\boldsymbol{x}$ and are treated as precomputation. In computer implementation, the intermediate results c_i are carried over to the next step Second-Upward, whereas the contents of d_i are discarded and the allocated memory is reused.
Second-Upward(`root`)

function Common-Upward(i)
  if i is leaf **then**
    $d_i \leftarrow U_i^T w_i$
  else
    for all children j of i **do**
      Common-Upward(j)
      $d_i \leftarrow d_i + W_i^T d_j$, **if** i is not root
    end for
  end if
  if i is not root **then**
    for all siblings l of i **do** $c_l \leftarrow \Sigma_p^T d_i$ **end for** ▷ p is parent of i
  end if
end function

function Second-Upward(i)
  if i is leaf **then**
    $d_i \leftarrow K(\underline{X}_p, \underline{X}_p)^{-1} k(\underline{X}_p, \boldsymbol{x})$ ▷ p is parent of i
    $z \leftarrow w_i^T k(X_i, \boldsymbol{x})$
  else
    Find the child j (among all children of i) where $\boldsymbol{x}$ lies on
    Second-Upward(j)
    $d_i \leftarrow W_i^T d_j$ if i is not root
  end if
  $z \leftarrow z + c_i^T d_i$ if i is not root
end function

4.1 Partitioning of Domain

Whereas the partitioning of a low-dimensional domain can be made regular, the partitioning of a high-dimensional domain is inherently difficult because of the curse of dimensionality. Several data-driven approaches appear to be natural choices but they have pros and cons.

The k-d tree (Bentley, 1975) approach is known to be efficient for nearest neighbor search. Generally, the approach iteratively selects an axis of the bounding box that contains the training points and partitions the axis such that the numbers of points on both sides are balanced. A drawback of this approach, particularly in the context of classification, lies in

the case that some attributes of the data are binary and the counts are highly imbalanced. These attributes sometimes result from a conversion of categorical attributes to numeric ones. Partitioning on these axes is unlikely to be balanced. An alternative is to partition the axis into two segments of equal length. This approach often results in highly imbalanced partitioning because data are generally not evenly distributed.

The k-means (MacQueen, 1967) approach partitions a point set through a k-means clustering and hence the partitioning is a Voronoi diagram of the cluster centers. The resulting tree is not necessarily binary if there exhibits more than two natural clusters in the data on some level. An advantage of this approach is that the clustering often results in a tight grouping of the points, such that the subsequent Nyström approximations bare a good quality. A disadvantage is that the approach suffers from loss of clusters during iterations and it is less robust if only one set of initial guesses is used. Our experience indicates that a robust implementation of k-means sometimes costs much more than does the rest of the training computation.

The PCA (Pearson, 1901) approach recursively partitions the data according to the principal axis (the direction along which the data varies the most). This approach is based on a Gaussian assumption and often results in compact partitions if the data indeed conforms to this assumption. However, it ignores the skewness and other higher order moments, which are also crucial to the shape of data in high dimensions. In the standard case, the hyperplane that partitions the data passes through the mean, which may result in highly imbalanced partitions. An alternative is to move the hyperplane along the principal direction such that the two partitions are always balanced. Computationwise, this approach requires computing the dominant singular vectors of the shifted data matrix (Chen et al., 2009), which can be achieved by using a power iteration (Golub and Van Loan, 1996) or the Lanczos algorithm (Saad, 2003). However, even if we compute the singular vectors only approximately, we find that the cost is still too high (see e.g., Section 5.2).

For computational efficiency, we recommend the random projection (Johnson and Lindenstrauss, 1984) approach. This approach uses a random vector as the normal direction of the partitioning hyperplane and positions the hyperplane such that the numbers of points on the two sides are balanced. Computationwise, it amounts to projecting the points along the random direction and splitting them in two halves around the median. This approach is motivated by the study of dimension reduction, where it was observed that the projection quality on a random subspace is as good as that on the space spanned by the dominant singular vectors, in the sense that Euclidean distances are approximately preserved (Dasgupta and Gupta, 2002). Moreover, the heuristic use of a one-dimensional subspace for projection is robust and is computationally highly efficient. In such a case, the purpose is not to preserve the Euclidean distance, but rather, to serve as an efficient procedure for partitioning. In our experience, the eventual regression/classification performance of using this approach is almost identical to that of the PCA approach (see Section 5.2).

Under random projection, one can quickly determine which partition a new point $\boldsymbol{x}$ lies in, by comparing its projected coordinate with the median of those of the training points.

4.2 Choice of Landmark Points

The set of landmark points $\underline{X}_i$ associated to each domain S_i simply consists of uniformly random samples of the point set X_i.

Some alternatives exist. Using the k-means centers of X_i as the landmark points may improve the Nyström approximations, but computing them for every nonleaf node i is often much more expensive than the rest of the training computation. On the other hand, using the uniformly random samples of the domain S_i as the landmark points may sound theoretically preferable, but they are difficult to compute because the domains are polyhedra rather than regular shapes (e.g., boxes).

We mentioned in Section 2.2 that the compositional kernel $k_{\text{compositional}}$ is a special case of $k_{\text{hierarchical}}$, if in the partitioning tree every child of the root is a leaf node. Interestingly, another interpretation exists. If we relax the requirement that the landmark points $\underline{X}_i$ must reside within the domain S_i, then for any partitioning tree, using the same set of landmark points $\underline{X}_i$ for every i also results in the compositional kernel $k_{\text{compositional}}$. By following the proof of Theorem 6, one sees that the relation $\underline{X}_i \subset S_i$ is in fact nonessential. Hence, the requirement $\underline{X}_i \subset S_i$ reflects only the modeling desire of a better kernel approximation in local domains; it is not a necessary condition for kernel validity.

4.3 Numerical Stability

A notorious numerical challenge for kernel matrices K is that they are exceedingly ill-conditioned as the size increases. Hence, the regularization λI serves as an important rescue of matrix inversion. This challenge is less severe for the matrix of the proposed kernel, $K_{\text{hierarchical}}(X, X)$, but since the components of this matrix are defined with the term $K(\underline{X}_i, \underline{X}_i)^{-1}$ for nonleaf nodes i, one must ensure that each $K(\underline{X}_i, \underline{X}_i)$ is not too ill conditioned. Generally, when the set $\underline{X}_i$ is not large, the conditioning of $K(\underline{X}_i, \underline{X}_i)$ is acceptable; however, for a robust safeguard, we use a small $\lambda' < \lambda$ for help. Instead of treating $k(\boldsymbol{x}, \boldsymbol{x}')$ as the base kernel and λ as the amount of regularization, we treat $k'(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x}, \boldsymbol{x}') + \lambda' \delta_{\boldsymbol{x}, \boldsymbol{x}'}$ as the base kernel and $\lambda - \lambda'$ the regularization, where δ is the Kronecker delta. Then, we use k' to build the kernel $k'_{\text{hierarchical}}$ and add to the matrix $K'_{\text{hierarchical}}$ a regularization $(\lambda - \lambda')I$.

4.4 Sizes

Two parameters exist for the proposed kernel. First, the recursive partitioning needs a stopping criterion; we say that a set of points is not further partitioned when the cardinality is less than or equal to n_0. Second, we need to specify the sizes of the conditioned sets $\underline{X}_i$. Whereas one may argue that the conditioned set needs to grow with the size of the point set for maintaining the approximation quality, a substantial increase in the size of $\underline{X}_i$ across levels will result in too high a computational cost, forfeiting the purpose of the kernel. Therefore, we mandate that each conditioned set have the same size r.

It will be convenient if we can consolidate the two parameters so that different approximate kernels are comparable. It will also be beneficial to have a guided choice of them. These requirements are indeed achievable. Under a balanced binary partitioning, n_0 can effectively take only these values: $\lceil n/2^j \rceil$, for $j = 0, 1, \ldots \lfloor \log_2 n \rfloor$. On the other hand, it

is not sensible to have the rank r greater than n_0 because the smallest off-diagonal blocks have a size at most $n_0 \times n_0$. Then, we take

$$n_0 = \lceil n/2^j \rceil \quad \text{and} \quad r = \lfloor n/2^j \rfloor \quad \text{for some } j. \tag{22}$$

4.5 Cost Analysis

We are now ready to perform a full analysis of the computational costs based on the practical choices made in the preceding subsections.

Memory Cost Let us first consider the memory cost of storing the matrix A. For simplicity, assume that n is a power of 2 (and thus $n_0 = r$). Then, the partitioning tree has n/n_0 leaf nodes and $n/n_0 - 1$ nonleaf nodes. Hence, the memory costs for the factors A_{ii}, U_i, Σ_p, and W_p are $n_0^2 n/n_0$, nr, $r^2(n/n_0 - 1)$, and $r^2(n/n_0 - 1)$, respectively. Therefore, the total memory cost is approximately $4nr$.

Arithmetic Cost, Algorithm 1 The arithmetic cost of matrix-vector multiplication (Algorithm 1) is $O(nr)$, because the subroutines Upward and Downward for each node performs $O(r^2)$ calculations, discounting the recursion calls. These recursions essentially constitute a tree traversal, which visits all the $2n/n_0 - 1$ nodes. Therefore, the total cost is $O(nr)$. In fact, a more detailed analysis may reveal the constant inside the big-O notation. Note that the matrix-vector multiplications $W_i^T c_j$ and $W_i d_i$ occur for each node j, $\Sigma_p c_i$ occurs for each node i, and $U_i^T b_i$, $A_{ii} b_i$, and $U_i d_i$ occur for each leaf node i. The matrix size of all these multiplications is $r \times r$. Therefore, if each multiplication requires $2r^2$ arithmetic operations, then the total cost is approximately $2r^2 \times (2n/n_0 + 2n/n_0 + 2n/n_0 + n/n_0 + n/n_0 + n/n_0) = 18nr$, ignoring lower-order terms.

Arithmetic Cost, Algorithm 2 Based on a similar argument, the arithmetic cost of matrix inversion (Algorithm 2) is $O(nr^2)$. A more precise estimate is $37nr^2$, if factorizing a general $r \times r$ matrix, factorizing a symmetric positive-definite matrix, performing a triangular solve, and multiplying two $r \times r$ matrices require $2r^3/3$, $r^3/3$, r^2, and $2r^3$ operations, respectively. As before, we ignore the lower-order terms.

Arithmetic Cost, Algorithm 3, Precomputation Phase Now consider the precomputation phase of the inner product $w^T k_{\text{hierarchical}}(X, \boldsymbol{x})$ (Algorithm 3). In computer implementation, we move the first line (prefactorizing $K(\underline{X}_p, \underline{X}_p)$) to the matrix construction part (to be discussed in Section 4.5). Hence, the precomputation of Algorithm 3 is dominated by the third line Common-Upward(`root`). The arithmetic cost is $O(nr)$, or $10nr$ in a more precise account.

Arithmetic Cost, Algorithm 3, $\boldsymbol{x}$-Dependent Phase Assume that evaluating the kernel function $k(\boldsymbol{x}, \boldsymbol{x}')$ requires $O(\text{nz}(\boldsymbol{x}) + \text{nz}(\boldsymbol{x}'))$ arithmetic operations, where $\text{nz}(\cdot)$ denotes the number of nonzeros of a data point. Further assume that for two point sets Y and Z of sizes n_Y and n_Z respectively, evaluating the vector $k(Y, \boldsymbol{x})$ requires $O(n_Y \text{ nz}(\boldsymbol{x}) + \text{nz}(Y))$ operations, and evaluating the matrix $K(Y, Z)$ requires $O(n_Y \text{ nz}(Z) + n_Z \text{ nz}(Y))$ operations, where the notation $\text{nz}(\cdot)$ is extended for point sets in a straightforward manner.

Now we consider the cost of the $\boldsymbol{x}$-dependent phase of Algorithm 3. Line 20 costs $O(r \text{ nz}(\boldsymbol{x}) + \text{nz}(\underline{X}_p)) + O(r^2)$, where the first term comes from evaluating $k(\underline{X}_p, \boldsymbol{x})$ and the

second term a linear-solve. Note that the matrix $K(\underline{X}_p, \underline{X}_p)$ has already been evaluated and prefactorized prior to this phase. In fact, such a computation is done when constructing $K_{\text{hierarchical}}$. Similarly, line 21 costs $O(n_0 \operatorname{nz}(\boldsymbol{x}) + \operatorname{nz}(X_i)) + O(n_0)$. Line 23 costs $O(\operatorname{nz}(\boldsymbol{x}))$, because it amounts to projecting $\boldsymbol{x}$ on a direction. Then, the overall cost of the $\boldsymbol{x}$-dependent phase of Algorithm 3 is

$$O\Big(r \operatorname{nz}(\boldsymbol{x}) + \operatorname{nz}(X_i) + \operatorname{nz}(\underline{X}_p) + (r^2 + \operatorname{nz}(\boldsymbol{x})) \log_2(n/r)\Big), \tag{23}$$

where $\boldsymbol{x}$ lies in the leaf node i and p is its parent.

Arithmetic Cost, Matrix Construction We shall also consider the cost of hierarchical partitioning and the instantiation of the matrix $A = K_{\text{hierarchical}}$. In the partitioning of a set of n points X, generating a random projection direction costs $O(d)$, computing the projected coordinates costs $O(\operatorname{nz}(X))$, finding the median costs $O(n)$, and permuting the points costs $O(\operatorname{nz}(X))$. Then, counting recursion, the total cost is $O((d + n + \operatorname{nz}(X)) \log_2(n/r))$. On the other hand, in the instantiation of the matrix A, computing A_{ii} for all leaf nodes i costs $O(r \operatorname{nz}(X))$, computing U_i for all leaf nodes i costs $O(r[n + \operatorname{nz}(X) + \sum_{i \text{ is leaf}} \operatorname{nz}(\underline{X}_p)])$, computing Σ_p for all nonleaf nodes p costs $O(r[\sum_{p \text{ is nonleaf}} \operatorname{nz}(\underline{X}_p)])$, factorizing these Σ_p's costs $O(nr^2)$, computing W_i for all nonleaf and nonroot nodes i costs $O(r[n + \sum_{i \text{ is not leaf and not root}} \operatorname{nz}(\underline{X}_i) + \operatorname{nz}(\underline{X}_p)])$, and finding all landmark point sets costs $O(n)$. Therefore, the total cost of instantiating A is $O(r[nr + \operatorname{nz}(X) + \sum_{i \text{ is not leaf}} \operatorname{nz}(\underline{X}_i)])$.

Arithmetic Cost Summary, Training and Testing Summarizing the above analysis, we see that the overall cost of training (including partitioning, instantiating A, matrix inversion, matrix-vector multiplication, and preprocessing of Algorithm 3) is

$$O\left((d + n + \operatorname{nz}(X)) \log_2(n/r) + nr^2 + r\left[\operatorname{nz}(X) + \sum_{i \text{ is not leaf}} \operatorname{nz}(\underline{X}_i)\right]\right). \tag{24}$$

The overall cost of testing per $\boldsymbol{x}$ has been given in (23).

We remark that the term $\log_2(n/r)$ in both (23) and (24) is dominated by r when $n \leq r2^r$. Moreover, when the data X is dense, the training cost (24) simplifies to $O(nr^2 + ndr)$ and the testing cost per point (23) simplifies to $O(r^3 + dr)$. Suppressing the data dimension d, these costs are further simplified to $O(nr^2)$ and $O(r^3)$, respectively.

5. Experimental Results

In this section, we present comprehensive experiments to demonstrate the effectiveness of the proposed kernel. The experiments were performed on one node of a computing cluster with 512 GB memory and an IBM POWER8 12-core processor, unless otherwise stated. The programs were implemented in C++. The linear algebra routines (BLAS and LAPACK) were linked against the IBM ESSL library and some obviously parallelizable for-loops were decorated by OpenMP pragmas, so that the experiments with large data sets can be completed in a reasonable time frame. However, the programs were neither fully optimized nor fully parallelized, leaving room for improvement on both memory consumption and execution time.

Table 1: Data sets.

Name	Type	d	n Train	n' Test
cadata	regression	8	16,512	4,128
YearPredictionMSD	regression	90	463,518	51,630
ijcnn1	binary classification	22	35,000	91,701
covtype.binary	binary classification	54	464,809	116,203
SUSY	binary classification	18	4,000,000	1,000,000
mnist	10 classes	780	60,000	10,000
acoustic	3 classes	50	78,823	19,705
covtype	7 classes	54	464,809	116,203

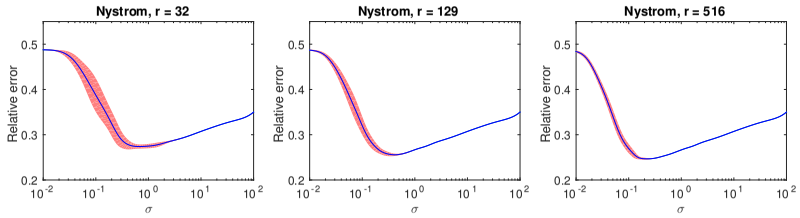
Table 1 summarizes the data sets used for experiments. They were all downloaded from `http://www.csie.ntu.edu.tw/~cjlin/libsvm/` and are widely used as benchmarks of kernel methods. We selected these data sets primarily because of their varying sizes. Some of the data sets come with a training and a testing part; for those not, we performed a 4:1 split to create the two parts, respectively. In some of the data sets, the attributes had already been normalized to within $[0, 1]$ or $[-1, 1]$; for those not, we performed such a normalization. We also preprocessed the data sets by removing duplicate and conflicting records (whose labels are inconsistent) in the training sets. Such records are infrequent.

5.1 Effect of Randomness

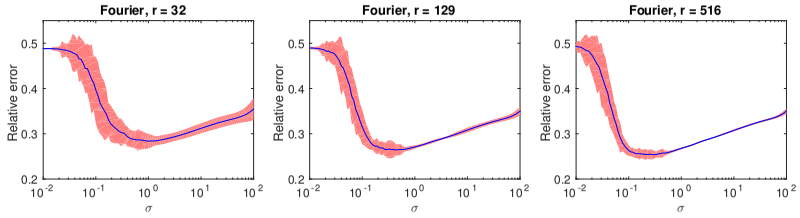
Throughout Section 5, we will compare various approximate kernels: Nyström approximation $k_{\text{Nyström}}$, random Fourier features k_{Fourier}, cross-domain independent kernel $k_{\text{independent}}$, and the proposed kernel $k_{\text{hierarchical}}$. The partitioning in the cross-domain independent kernel is the same as that in the proposed kernel, except that the hierarchy is flattened. Because of the random nature of all these kernels (e.g., landmark points, sampling, and partitioning), we first study how the performance is affected by randomization. Note that the quantity r is comparable across kernels, even though its specific meaning is different.

The data set for demonstration is cadata. We use the Gaussian kernel (5) as an example. As hinted earlier, the choice of the range parameter σ affects the quality of various kernels. Therefore, the experiment setup is to use a reasonable regularization $\lambda = 0.01$ and to vary the choice of σ in a large interval (between 0.01 and 100) such that the optimal σ falls within the interval. We set the rank r (and the leaf size n_0) according to (22) with three particular choices: $r = 32$, 129, and 516.

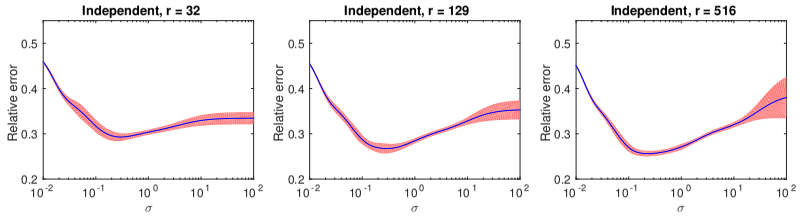
For each r, we repeat 30 times with different random seeds; but the seed always stays the same every time when the range of σ is swept. The results (relative testing error versus σ) are summarized in Figure 3, with mean (blue curve) and standard deviation (red band) plotted. One sees that the red bands of random Fourier features are not smooth; this is because each single error curve from a fixed seed is nonsmooth. Moreover, the error curves for Nyström approximation have a nonnegligible variation when σ is small, whereas those for the independent kernel vary significantly when σ is large. The error curves of the proposed kernel are the most stable, with the narrowest standard deviation band. Generally



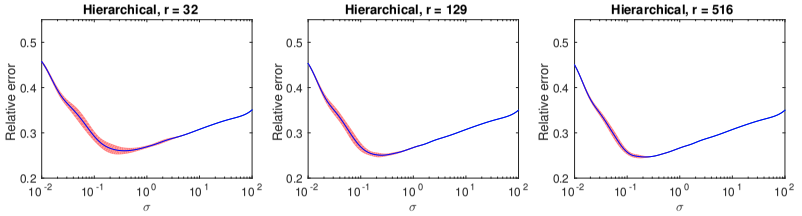
(a) Nyström approximation



(b) random Fourier features



(c) Cross-domain independent kernel



(d) Hierarchically compositional kernel

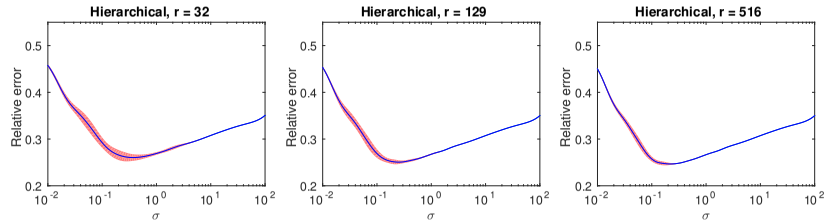
Figure 3: Regression error curves versus σ. Results are summarized from 30 repeated runs and the red bands show three times standard deviation. Data set: cadata.

speaking, as r increases, all approximate kernels yield a more and more stable error curve, except the peculiar case of the independent kernel at large σ.

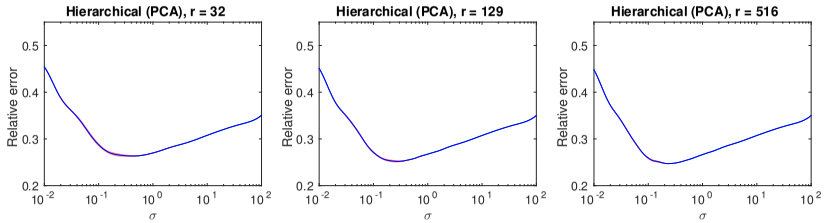
The unstable performance caused by randomness is unfavorable for parameter estimation, because the valley, where the optimal parameter stands, may move substantially. The nonsmoothness exhibited in the Fourier approach also renders difficulty. Although the unfavorable behaviors are substantially alleviated when r increases to 516 in this particular data set, to our experience, the relieving size for r correlates with the data size; that is, the larger n, the higher r needs to increase to. In this regard, the proposed kernel is the most favorable because its performance is relatively stable even for small r.

5.2 Partitioning Approaches

We further compare the methods for partitioning needed by the proposed kernel. The comparison focuses on three aspects: effect of randomness, testing error, and computational efficiency. Among the several possible choices discussed in Section 4.1, only the PCA approach and the random projection approach (which is recommended) yield a balanced partitioning; hence, we compare only these two.



(a) Random partitioning



(b) PCA partitioning

Figure 4: Regression error curves of the proposed method. Data are recursively partitioned by using random projections (top row) and by PCA (bottom row), respectively.

Figure 4 summarizes the error curves (mean and standard deviation) resulting from the two approaches, based on the same setting as in the preceding subsection. The top row corresponds to the recommended approach whereas the bottom row PCA. One sees that

the mean error curves are almost identical in both approaches. PCA is less influenced by randomization. Such a result should not be surprising, because PCA bares no randomness at all on partitioning; the only randomness comes from the landmark points. Although the variation of the error curves of the random projection approach is higher, such a variation is acceptable, particularly when it is compared with that of other approximate kernels in the preceding subsection.

The essential reason why random projection is favored over PCA comes from the consideration of computational efficiency. To generate the normal direction of a partitioning hyperplane, random projection amounts to only generating d random numbers, whereas PCA requires computing the dominant singular vector of the shifted data matrix. Table 2 presents the time costs of the additional singular vector computation, against the partitioning cost and the overall training cost by using random projection. We call the singular vector computation "overhead." The overhead is much higher with respect to the partitioning step, because such a step has a negligible cost compared to the overall training. The overhead is also generally higher when r is smaller because there requires more partitioning. The overhead with respect to partitioning easily exceeds 100% for quite a few of the data sets, and sometimes it is even a few thousand percents. For the data set mnist, which has the largest dimension d, the overhead with respect to overall training ranges from approximately 50% to 800%.

5.3 Performance Results for Various Data Sets

We now compare various approximate kernels on all the data sets listed in Table 1. For each kernel and each r, we obtain the performance result through a grid search of the optimal parameters σ and λ. Because of the high cost of parameter tuning, no repetitions are run and hence the performances may be susceptible to randomization. Hence, conclusions are drawn across data sets and we refrain from over-interpreting the results of an individual data set. We run experiments with a few r's and we are particularly interested in the performance trend as r increases.

The results are summarized in Figures 5 and 6. In the panel of the plots, each row corresponds to one data set. The three columns are performance versus r, training time, and memory consumption, respectively. The performance is measured as the relative error in the regression case and the accuracy in the classification case. The memory cost is estimated and normalized. According to the analysis in Section 4.5, the memory cost of the proposed kernel is approximately $4r$ per training point, whereas for the other approximate kernels we use an estimation of r per point.

A few observations follow. First, the proposed kernel almost always yields the best performance versus r, except for the data set YearPredictionMSD. Such an observation confirms the effectiveness of the proposed method, which combines the advantages of the Nyström method globally and the cross-domain independent kernel locally.

Second, the Fourier method generally runs the fastest, followed by Nyström and independent, whose speeds are comparably similar; and the proposed method falls behind. Although all methods have the same asymptotic cost $O(nr^2)$, in a finer level of analysis, Fourier wins over Nyström because the generation of random features in the former is often less expensive than the kernel evaluation in the latter. Of course, such a fact could also be

Table 2: PCA overhead with respect to partitioning and to overall training, under different r.

cadata		
r	partition.	train.
32	91.16%	9.48%
64	139.23%	7.59%
129	51.33%	2.29%
258	37.93%	0.94%
516	52.86%	0.78%

YearPredictionMSD		
r	partition.	train.
56	687.69%	71.74%
113	616.68%	50.97%
226	630.07%	21.07%
452	226.84%	5.64%
905	216.37%	2.66%

ijcnn1		
r	partition.	train.
34	199.52%	15.95%
68	8.80%	0.51%
136	153.16%	4.23%
273	58.42%	1.37%
546	64.74%	0.58%

covtype.binary		
r	partition.	train.
56	86.40%	7.03%
113	74.75%	4.18%
226	56.26%	1.88%
453	28.75%	0.59%
907	93.58%	0.66%

SUSY		
r	partition.	train.
61	85.40%	4.08%
122	99.44%	3.21%
244	52.10%	1.61%
488	27.41%	0.40%
976	79.71%	0.49%

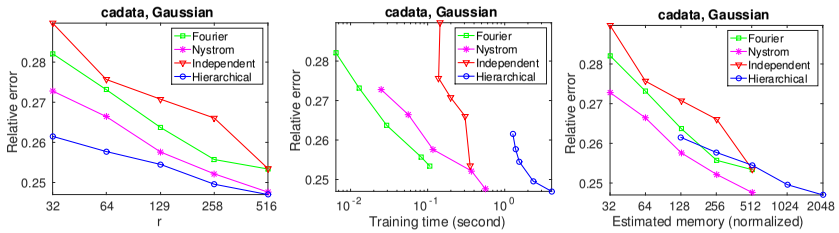
mnist		
r	partition.	train.
58	3973.02%	805.67%
117	3775.47%	508.89%
234	4341.62%	383.34%
468	3126.89%	151.64%
937	2175.98%	51.17%

acoustic		
r	partition.	train.
38	664.45%	53.07%
76	411.56%	29.15%
153	435.01%	21.66%
307	264.07%	7.62%
615	164.62%	2.28%

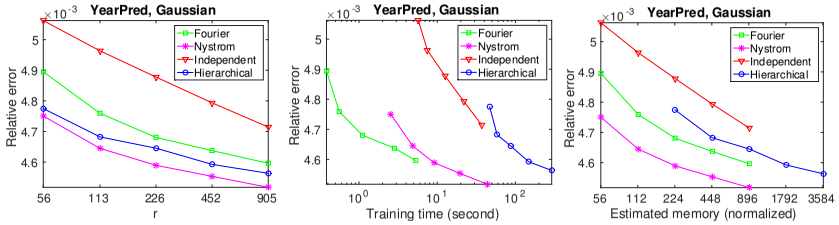
covtype		
r	partition.	train.
56	96.86%	7.02%
113	155.34%	6.29%
226	75.37%	2.77%
453	105.68%	1.60%
907	32.17%	0.30%

interpreted on the other hand as a weakness of the Fourier method: It is applicable to only a numeric data representation, whereas Nyström assumes no numeric form of data but a kernel. Nyström has a similar time cost as does the independent kernel; the former requires more kernel evaluations but the latter needs computing the sequence of subdomains each point belongs to. The independent kernel and the proposed kernel need similar computations (small matrix multiplications and factorizations), but the number of such operations is a constant times more for the proposed kernel; hence, not surprisingly it requires more computational effort.

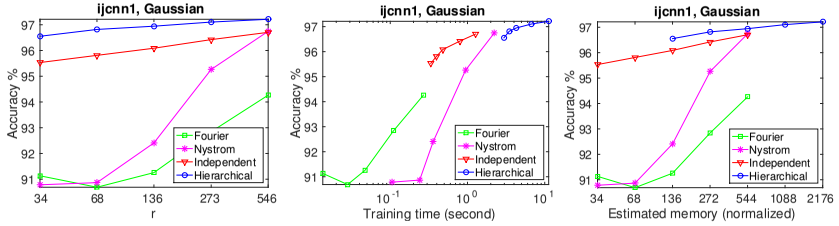
Note that in the Fourier and the Nyström methods, large matrices are operated in a small number of times. The implementation straightforwardly makes use of sophisticated linear algebra libraries, where parallelism and cache reuse have been aggressively optimized (Anderson et al., 1999; Goto and Geijn, 2008). On the other hand, in the independent and the proposed kernels, small matrices are operated in a large number of times. Moreover, there exist fragmented computations such as tree traversals and data shuffles. Optimizing these computations in terms of parallelism and cache reuse is possible and the executation times have the potential to be substantially improved.



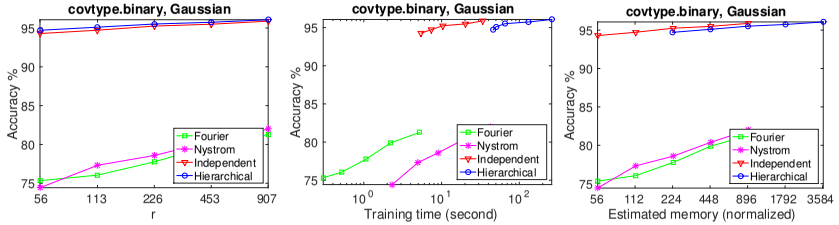
(a) cadata, regression



(b) YearPredictionMSD, regression

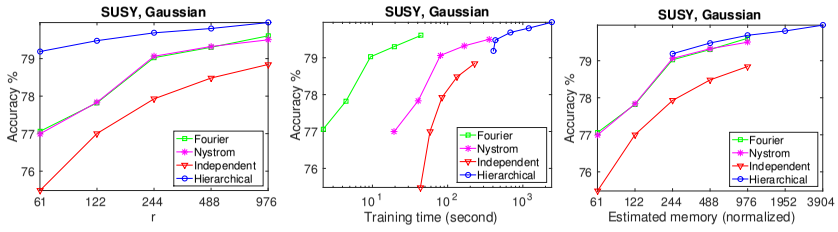


(c) ijcnn1, binary classification

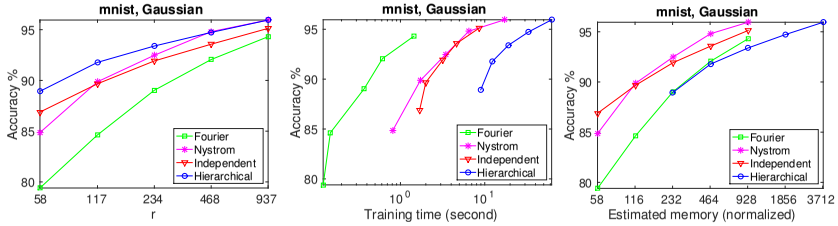


(d) covtype.binary, binary classification

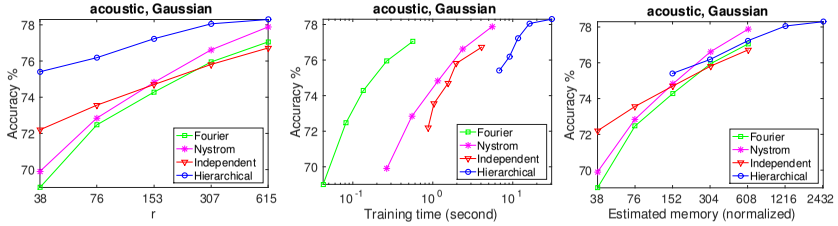
Figure 5: Performance versus r, time, and memory. Gaussian kernel.



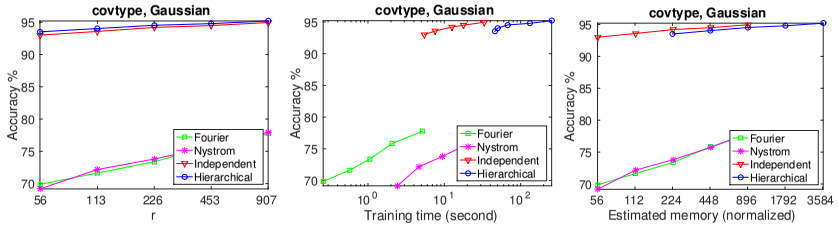
(a) SUSY, binary classification



(b) mnist, multiclass classification



(c) acoustic, multiclass classification



(d) covtype, multiclass classification

Figure 6: (Continued) Performance versus r, time, and memory. Gaussian kernel.

The third observation of Figures 5 and 6 focuses on the performance versus memory consumption. Since all the other methods are estimated to use the same amount of memory, the plots are essentially moving only the curve of the proposed kernel to the right compared with the performance-versus-r plots. Then, with the same amount of memory, some methods behave better than others in some data sets. There does not exist a consistent winner. One sees that the proposed method is the best performing for the data sets ijcnn1 and SUSY.

The last observation is that for the data set covtype (both the binary and the multiclass case), a significant performance gap occurs between the low-rank (Fourier and Nyström) and the full-rank (independent and hierarchical) kernels. This is the exemplified case where the eigenvalues in the kernel matrix decay too slowly, such that low-rank kernels with an insufficiently large rank r are clearly underperforming.

5.4 A Different Base Kernel

We consider using a different base kernel and inspect whether the observations made in the preceding subsection still hold. Here, we use the Laplace kernel

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|_1}{\sigma}\right)$$

popularized by Rahimi and Recht (2007). Clearly, this kernel is the tensor product of one-dimensional exponential kernels $\exp(-|x_i - x_i'|/\sigma)$ for all attributes i. In the GP context, the exponential kernel gives rise to the well-known Ornstein-Uhlenbeck process. Both the exponential kernel and the Gaussian kernel are special cases of the Matérn family of kernels, but their characteristics are opposite: the latter yields an extremely smooth process whereas the former highly rough (Stein, 1999). Hence, one might expect that their results substantially differ.

However, the answer is to the contrary. To avoid cluttering, we leave the plots to the appendix (see Figures 9 and 10); they are similar to those shown in Figures 5 and 6 of the Gaussian kernel. Specifically, the general trends and the specific performance values are quite close across the two base kernels. A possible reason is that the (optimal) regularization λ generally lies on the order of 0.01 to 0.1, which is relatively large compared with the kernel values, whose peak occurs at $k(0) = 1$. In the GP context, the noise level λ is so high that it eclipses the effect of smoothness—the rough variation of data may as well be interpreted as noise instead. Then, the smoothness of the kernel matters little and thus the results of different base kernels look similar.

Based on this observation, we additionally perform experiments with the inverse multiquadric kernel

$$k(\boldsymbol{x}, \boldsymbol{x}') = \frac{\sigma^2}{\sqrt{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2 + \sigma^2}}.$$

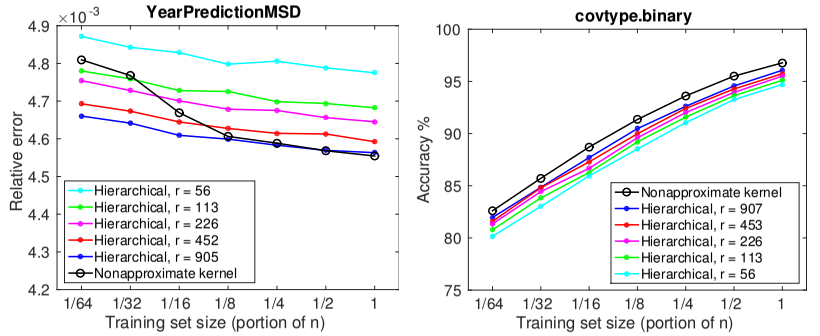
The strict positive definiteness of this kernel is proved in Micchelli (1986); but its Fourier transform is little known and hence we do not compare with the random Fourier method. The results are shown in the appendix (see Figures 11 and 12). One sees again that they are quite similar to those of the Gaussian and Laplace kernels.

5.5 Trade-Off between n and r

A folk wisdom in machine learning is that more data beats a more complex model (Domingos, 2012). The applicability of this knowledge lies in the regime where the hypothesis space has not been saturated by data. For kernel methods, a natural question to ask is whether subsampling is viable, considering that the computational effort is nontrivial with respect to the data size n. A relevant question in the context of approximate kernels is whether a trade-off exists between n and r, given a fixed budget nr. This budget comes from the memory constraint, because the memory cost is $O(nr)$ in various approximate kernels. Note that if nr is fixed, the arithmetic cost $O(nr^2)$ will increase when n becomes smaller, but the kernels tend to admit a better approximation quality.

Another scenario that resolves the interplay between n and r is the following: Suppose the current computational resources have been fully leveraged for computation and one is offered an upgrade such that the memory capacity is increased t times. To expect the best performance improvement, should one seek t times more training samples (provided feasible), or a smaller increase in training samples but meanwhile also an increase in r?

To answer this question, we use two data sets, YearPredictionMSD and covtype.binary, and progressively downsize them by a factor of two. Recall that the former is a regression problem whereas the latter classification. We plot in Figure 7 the performance curves versus the training size, for a few r's in the progression of approximately a factor of two. For comparison, we also perform the full-fledged computation with the nonapproximate kernel through solving (2) directly by using a preconditioned Krylov method on a cluster of AWS EC2 machines (see Avron et al. (2016) for details). The performance curve is black with circle markers.



(a) YearPredictionMSD, regression (b) covtype.binary, binary classification

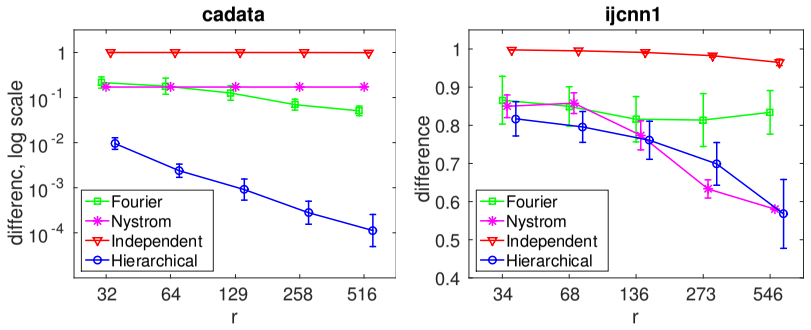
Figure 7: Performance under different training sizes and r. Kernel: Gaussian.

The question does not seem to bare a clear and simple answer after one investigates the two plots in Figure 7. Commonly, the performance of the proposed kernel improves in a consistent pace as r increases. For covtype.binary, the curves approach that of the nonapproximate kernel; however, such is not the case for YearPredictionMSD. In this data

set, when the training size is small, increasing r may surpass the performance of the non-approximate kernel. Furthermore, for covtype.binary, increasing the training size clearly yields a much higher performance than does increasing r by the same factor; however, for YearPredictionMSD, increasing r is to the contrary more beneficial. Hence, what the trade-off between n and r favors appears to be data set dependent.

5.6 Kernel PCA

Kernel principal component analysis (kernel PCA; see Schölkopf et al. (1998)) is another popular application of kernel methods. The standard PCA defines the embedding of a point $\boldsymbol{x}$ as the projected coordinates along the principal components of the training set X; whereas kernel PCA defines the embedding as the projected coordinates of $\phi(\boldsymbol{x})$ along the principal components of $\phi(X)$, where ϕ denotes the mapping from the input space to the feature space. For low-rank kernels (e.g., Nyström and Fourier) that give the explicit feature map ϕ, kernel PCA may be straightforwardly computed through singular value decomposition of the feature points $\phi(X)$ after centering. For other kernels (e.g., cross-domain independent kernel and the proposed kernel), one may leverage the relation $k(\boldsymbol{x},\boldsymbol{x}') = \langle\phi(\boldsymbol{x}),\phi(\boldsymbol{x}')\rangle$ and compute the embedding through eigenvalue decomposition of the kernel matrix $K(X,X)$ after centering.



(a) cadata, $n = 16,512$ (b) ijcnn1, $n = 35,000$

Figure 8: Kernel PCA: Embedding alignment difference (embedding dimension = 3). The horizontal placement of each curve is slightly jittered to avoid cluttering.

We follow Zhang et al. (2008) and evaluate the embedding quality of different kernels against that of the base kernel. Specifically, let U be the embedding matrix, where each row corresponds to the embedding of one point $\boldsymbol{x}$; similarly, let $\widetilde{U}$ be the embedding matrix resulting from an approximate kernel. We use a matrix M to align $\widetilde{U}$ with U; that is, M minimizes $\|U - \widetilde{U}M\|_F$. Then, we show in Figure 8 the alignment difference $\|U - \widetilde{U}M\|_F/\|U\|_F$ for different approximate kernels with various (rank) r. The embedding dimension is fixed to 3 and the base kernel is Gaussian, with the choice of bandwidth σ approximately equal to the optimum found in the preceding experiments. One sees that generally the proposed kernel results in the smallest alignment difference.

6. Concluding Remarks

Kernel methods enjoy a strong theoretical support for data interpretation and predictive modeling; however, they come with a prohibitive computational price. Substantial efforts exist in the search of approximate kernels. Low-rank kernels are one of the most popular approaches because their costs have a linear dependency on the data size n and because of their relatively simple linear algebra implementation. From the matrix angle, the effectiveness of a low-rank approximation, however, depends heavily on the decay of eigenvalues, which is often found to be insufficiently fast for large data. We may interpret that low-rank approximations tackle the problem globally, because its purpose is to approximate the whole matrix. On the other hand, block diagonal approximations (e.g., the cross-domain independent kernel) focus on the preservation of local information only. They may approximate very poorly the kernel matrix, but they often produce surprisingly good results for regression and classification (see the covtype example in Section 5). Hence, the hierarchically compositional kernel proposed in this paper is motivated by the separate characteristics of these approximations: it preserves the full information in local domains and gradually applies low-rank approximations when the domains expand in a hierarchical fashion. The net effect is that the kernel admits an $O(nr)$ storage and an $O(nr^2)$ arithmetic cost similar to that of others, but the factor r may be reduced for a matching performance when n is large. This is favorable when computational resources are constrained; that is, as the data size n increases, the additional factor r must remain small in order for computation to be affordable.

The computational efficiency of the proposed kernel is supported by a delicate matrix structure. The resulting kernel matrix $K_{\text{hierarchical}}$ is a special case of the *recursively low-rank compressed matrices* studied in Chen (2014b). Such a hierarchical structure allows many otherwise $O(n^2)$–$O(n^3)$ dense matrix operations done with an $O(nr)$–$O(nr^2)$ cost. Examples of these matrix operations are matrix-vector multiplication, matrix inversion, log-determinant calculation (Chen, 2014b), and square-root decomposition (Chen, 2014a). In addition to the focus of this paper—regression and classification, such matrix operations are crucial in many machine learning and statistical settings, including Gaussian process modeling, simulation of random processes, likelihood calculation, Markov Chain Monte Carlo, and Bayesian inference. The proposed kernel hints a direction for performing scalable computations facing an ever increasing n.

The recursively low-rank compressed structure is not a unique discovery in the prosperous literature of structured matrix computations. Many practically useful structures in scientific computing (e.g., FMM matrices (Sun and Pitsianis, 2001) and hierarchical matrices (Hackbusch, 1999)) exploit the same low-rank property exhibited in the off-diagonal blocks of the matrix, but they differ from ours in fine details, including the admissibility of low-rank approximation, the method of approximation, the nesting of basis, the shape of the hierarchy tree, and the supported matrix operations. Apart from these differences, the most distinguishing feature of our work is that the matrix structure is amenable for high dimensional data, an arena where other structured matrix methods are rarely applicable.

This peculiarity roots in the fact that the decay of singular values of the off-diagonal blocks becomes much slower when the data dimension increases. These blocks generally have a full, or nearly full, rank when the data dimension is larger than three or four. Hence, it is challenging to develop computational methods from the angle of matrix approximations, if evaluation of quality is based on the approximation error. Instead, we develop theory from the perspective of kernels and ensure the preservation of positive-definiteness. Although the new kernel arises from some approximation of the old one, we may bypass the approximation interpretation and consider them separate kernels. The choice of kernels will then be justified via model selection. An additional benefit is that out-of-sample extension is straightforward.

Although the setting of this paper casts the data domain $\mathcal{X}$ as an Euclidean space for simplicity, it is possible to generalize the proposed method to a more abstract space (e.g., a metric space). In fact, the construction of the proposed kernel and the instantiation of the kernel matrix requires as minimal as the fact that the base kernel k is strictly positive-definite. The only exception is the partitioning of the data domain, where the partition membership must be inferred for every data point (including out of samples). In a metric space, one solution is the k-means clustering, because the clustering straightforwardly defines the partitions for the training data; moreover, the partition membership for a testing point is determined by the minimal distance (metric) from the point to the cluster centers.

Through experimentation, we have demonstrated the complex performance curves with respect to kernel parameters. The curves may vary significantly due to randomization, may be multimodal, and may even be nonsmooth. These phenomena pose a challenge for selecting the optimal parameters through cross validation and grid search. To make things worse, grid search is applicable only when the number of parameters is very small. An example when one may want more parameters is to introduce anisotropy to the kernel through specifying one range parameter for each or a few attributes. Hence, a more principled approach, which in theory can incorporate an arbitrary number of parameters, is to take the GP view and to maximize the Gaussian log-likelihood

$$L(\boldsymbol{\theta}) = -\frac{1}{2}\boldsymbol{y}^T K(\boldsymbol{\theta})^{-1}\boldsymbol{y} - \frac{1}{2}\log\det K(\boldsymbol{\theta}) - \frac{n}{2}\log 2\pi \tag{25}$$

through numerical optimization, where $\boldsymbol{\theta}$ is the vector of unknown parameters and $K(\boldsymbol{\theta})$ is the kernel matrix depending on $\boldsymbol{\theta}$. This approach, coined *maximum likelihood estimation* (MLE), is also a central subject in estimation theory. Optimizing (25) nevertheless is nontrivial because of the $O(n^3)$ cost for evaluating the objective function and its derivatives when K is dense. Existing research (Anitescu et al., 2012; Stein et al., 2013) solves an approximate problem and bounds the variance of the result against that of the original problem (25). Fortunately, most of the approximate kernels discussed in this paper allow an $O(nr^2)$ cost for evaluating $L(\boldsymbol{\theta})$. For example, the algorithm for computing the log-determinant term for a recursively low-rank compressed matrix is described in Chen (2014b). An avenue of future work is to adapt the log-determinant calculation discussed in Chen (2014b) to the proposed kernel and to develop robust optimization for parameter estimation.

Acknowledgments

We would like to acknowledge support for this work by the XDATA program of the Defense Advanced Research Projects Agency (DARPA), administered through Air Force Research Laboratory contract FA8750-12-C-0323. We are indebted to the editor and the referees whose comments have substantially improved the paper.

Appendix A. Proof of Theorem 6

The proof technique is similar to that of Theorem 3: we decompose the kernel $k_{\text{hierarchical}}$ as a sum of positive-definite terms and show that if k is strictly positive-definite, then the bilinear form cannot be zero if the coefficients are not all zero. To prevent cluttering, we use the short-hand notation $K_{\underline{i},\underline{j}}$ to denote $K(\underline{X}_i, \underline{X}_j)$.

For any node i with parent p (if any), define a function $\xi^{(i)}$ for $\boldsymbol{x}, \boldsymbol{x}' \in S_i$:

$$\xi^{(i)}(\boldsymbol{x},\boldsymbol{x}') = \begin{cases} k(\boldsymbol{x},\boldsymbol{x}') - k(\boldsymbol{x},\underline{X}_p)K^{-1}_{\underline{p},\underline{p}}k(\underline{X}_p,\boldsymbol{x}'), & i \text{ is leaf,} \\ \psi^{(i)}(\boldsymbol{x},\underline{X}_i)K^{-1}_{\underline{i},\underline{i}}[K_{\underline{i},\underline{i}} - K_{\underline{i},\underline{p}}K^{-1}_{\underline{p},\underline{p}}K_{\underline{p},\underline{i}}]K^{-1}_{\underline{i},\underline{i}}\psi^{(i)}(\underline{X}_i,\boldsymbol{x}'), & i \text{ is neither leaf nor root,} \\ \psi^{(i)}(\boldsymbol{x},\underline{X}_i)K^{-1}_{\underline{i},\underline{i}}\psi^{(i)}(\underline{X}_i,\boldsymbol{x}'), & i \text{ is root.} \end{cases} \tag{26}$$

Through telescoping, one sees that $k_{\text{hierarchical}}(\boldsymbol{x},\boldsymbol{x}') = k^{(\text{root})}(\boldsymbol{x},\boldsymbol{x}')$ is a sum of $\xi^{(i)}(\boldsymbol{x},\boldsymbol{x}')$ for all nodes i where $\boldsymbol{x},\boldsymbol{x}' \in S_i$. Because each $\xi^{(i)}$ is clearly positive-definite, so is $k_{\text{hierarchical}}$.

We now show the strict definiteness when k is strictly positive-definite. For any set of points $\{\boldsymbol{x}_i\}$ and any set of coefficients $\{\alpha_i\}$, the bilinear form $\sum_{jl}\alpha_j\alpha_l k_{\text{hierarchical}}(\boldsymbol{x}_j,\boldsymbol{x}_l)$ is zero if and only if

$$\sum_{\boldsymbol{x}_j,\boldsymbol{x}_l\in S_i}\alpha_j\alpha_l\xi^{(i)}(\boldsymbol{x}_j,\boldsymbol{x}_l) = 0, \quad \forall \text{ nodes } i. \tag{27}$$

If (27) holds, we shall prove the following statement:

> (Q) If i is not the root, it holds that $\alpha_j = 0$ for all j satisfying
>
> $$\boldsymbol{x}_j \in S_i \setminus \left[\bigcap_{q\in Des(i)} \underline{X}_q \cap \underline{X}_p\right],$$
>
> where p is the parent of i and $Des(i)$ means all the descendant nonleaf nodes of i, including i itself.

Once we prove this statement, we see that the only α_j that can possibly be nonzero are those satisfying $\boldsymbol{x}_j \in S \cap_{\text{all nodes } q} \underline{X}_q$. However, if any of these α_j is nonzero, then applying the third case of (26) on the root yields

$$\begin{aligned}\sum_{\boldsymbol{x}_j,\boldsymbol{x}_l\in S_{\text{root}}}\alpha_j\alpha_l\xi^{(\text{root})}(\boldsymbol{x}_j,\boldsymbol{x}_l) &= \sum_{\boldsymbol{x}_j,\boldsymbol{x}_l\in S_{\text{root}}}\alpha_j\alpha_l\psi^{(\text{root})}(\boldsymbol{x}_j,\underline{X}_{\text{root}})K^{-1}_{\underline{\text{root}},\underline{\text{root}}}\psi^{(\text{root})}(\underline{X}_{\text{root}},\boldsymbol{x}_l) \\ &= \sum_{\boldsymbol{x}_j,\boldsymbol{x}_l\in S\cap_{\text{all nodes } q}\underline{X}_q}\alpha_j\alpha_l e_j^T K_{\underline{\text{root}},\underline{\text{root}}}e_l \neq 0,\end{aligned}$$

contradicting (27). Here, e_j means the jth column of the identity matrix. Therefore, the bilinear form $\sum_{jl}\alpha_j\alpha_l k_{\text{hierarchical}}(\boldsymbol{x}_j,\boldsymbol{x}_l)$ cannot be zero if the coefficients are not all zero.

We now prove statement (Q) by induction. The base case is when i is a leaf node. Applying the same argument as that in the proof of Theorem 3 on the first case of (26), we have that $\alpha_j = 0$ for all j satisfying $\boldsymbol{x}_j \in S_i \backslash \underline{X}_p$. In the induction step, if (Q) holds for all children nodes i of p and if r is the parent of p, we apply the second case of (26) and obtain

$$0 = \sum_{\boldsymbol{x}_j, \boldsymbol{x}_l \in S_p} \alpha_j \alpha_l \xi^{(p)}(\boldsymbol{x}_j, \boldsymbol{x}_l) = \sum_{\boldsymbol{x}_j, \boldsymbol{x}_l \in S_p} \alpha_j \alpha_l \psi^{(p)}(\boldsymbol{x}_j, \underline{X}_p) K_{\underline{p},\underline{p}}^{-1} [K_{\underline{p},\underline{p}} - K_{\underline{p},\underline{r}} K_{\underline{r},\underline{r}}^{-1} K_{\underline{r},\underline{p}}] K_{\underline{p},\underline{p}}^{-1} \psi^{(p)}(\underline{X}_p, \boldsymbol{x}_l)$$
$$= \sum_{\boldsymbol{x}_j, \boldsymbol{x}_l \in S_p \backslash \cap_{q \in Des(p)} \underline{X}_q} \alpha_j \alpha_l e_j^T [K_{\underline{p},\underline{p}} - K_{\underline{p},\underline{r}} K_{\underline{r},\underline{r}}^{-1} K_{\underline{r},\underline{p}}] e_l. \qquad (28)$$

For the matrix $K_{\underline{p},\underline{p}} - K_{\underline{p},\underline{r}} K_{\underline{r},\underline{r}}^{-1} K_{\underline{r},\underline{p}}$ inside the square bracket, the block corresponding to $\underline{X}_p \backslash \underline{X}_r$ has full rank, whereas the rest is zero. Then, in order for (28) to be zero, one must have that $\alpha_j = 0$ for all j satisfying $\boldsymbol{x}_j \in \underline{X}_p \backslash \underline{X}_r$. Making a disjunction of this relation with that in the induction hypothesis: $\boldsymbol{x}_j \in S_p \backslash \cap_{q \in Des(p)} \underline{X}_q$, we obtain

$$\boldsymbol{x}_j \in S_p \Bigg\backslash \left[\bigcap_{q \in Des(p)} \underline{X}_q \cap \underline{X}_r \right],$$

which concludes the induction and subsequently concludes the proof of Theorem 6.

Appendix B. Performance Results with the Laplace Kernel

See Figures 9 and 10. These plots are qualitatively similar to those of the Gaussian kernel (Figures 5 and 6).

Appendix C. Performance Results with the Inverse Multiquadric Kernel

See Figures 11 and 12. These plots are qualitatively similar to those of the Gaussian kernel (Figures 5 and 6).

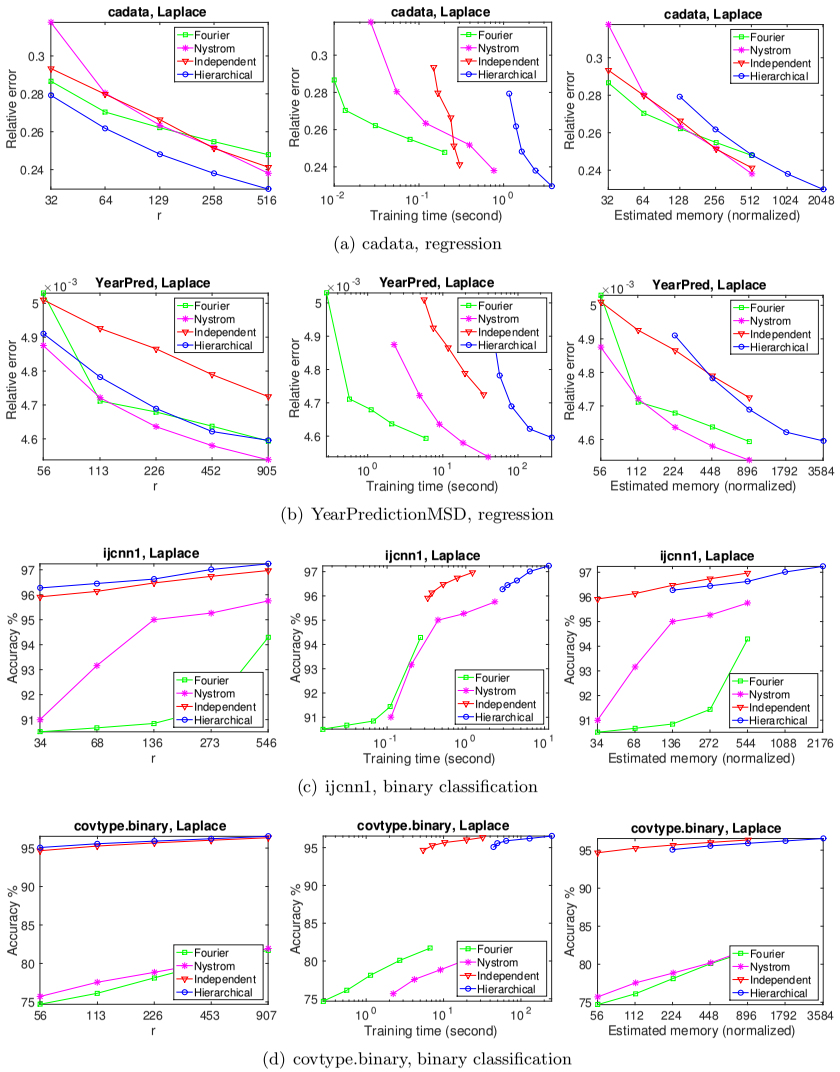
References

E. Anderson, Z. Bai, C. Bischof, L. S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, 1999.

Mihai Anitescu, Jie Chen, and Lei Wang. A matrix-free approach for solving the parametric Gaussian process maximum likelihood problem. *SIAM Journal on Scientific Computing*, 34(1):A240–A262, 2012.

Haim Avron and Vikas Sindhwani. High-performance kernel machines with implicit distributed optimization and randomization. *Technometrics*, 58(3):341–349, 2016.

Haim Avron, Kenneth Clarkson, and David Woodruff. Faster kernel ridge regression using sketching and preconditioning. Submitted, 2016.



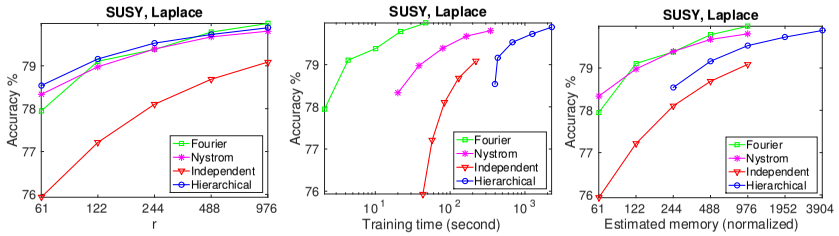
(a) cadata, regression

(b) YearPredictionMSD, regression

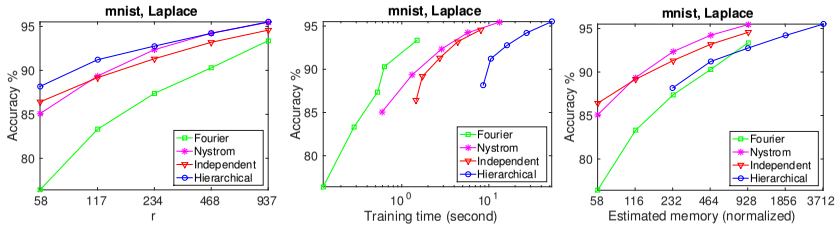
(c) ijcnn1, binary classification

(d) covtype.binary, binary classification

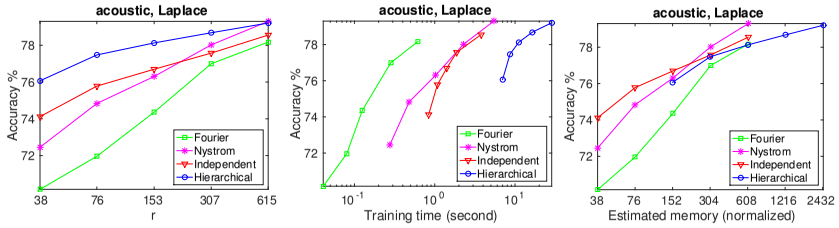
Figure 9: Performance versus r, time, and memory. Laplace kernel.



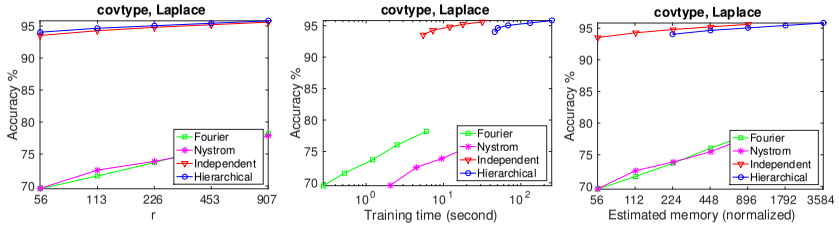
(a) SUSY, binary classification



(b) mnist, multiclass classification

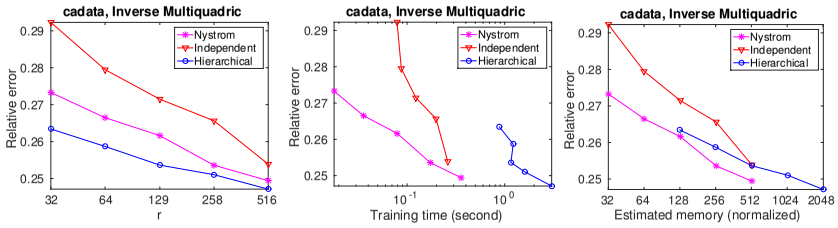


(c) acoustic, multiclass classification

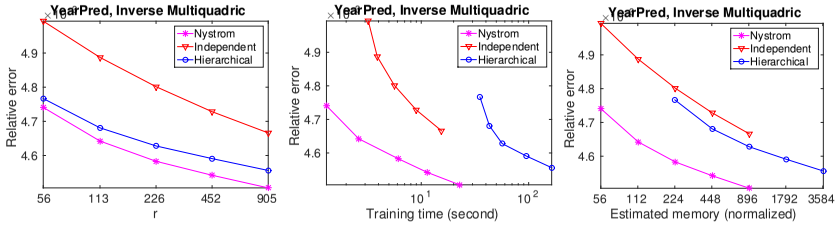


(d) covtype, multiclass classification

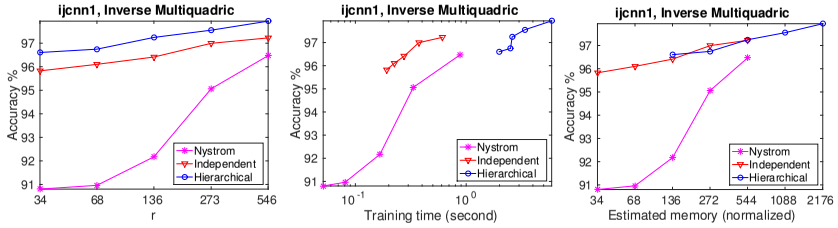
Figure 10: (Continued) Performance versus r, time, and memory. Laplace kernel.



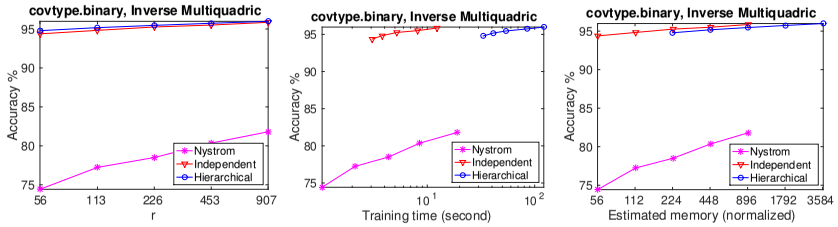
(a) cadata, regression



(b) YearPredictionMSD, regression

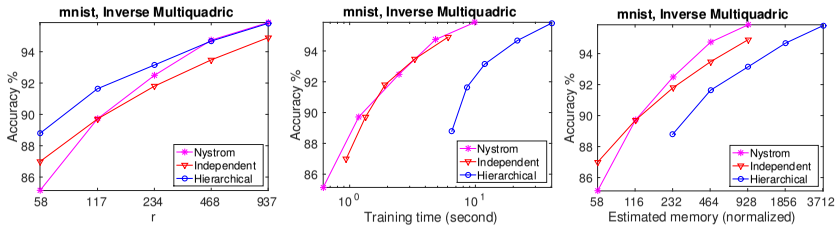


(c) ijcnn1, binary classification

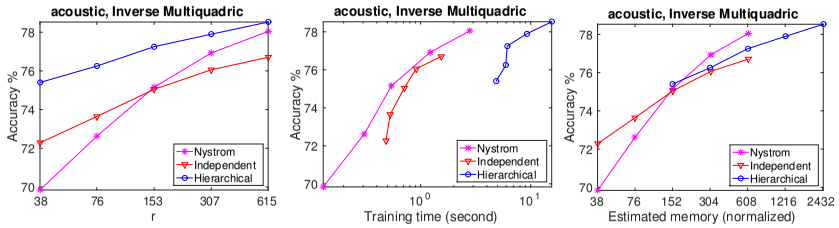


(d) covtype.binary, binary classification

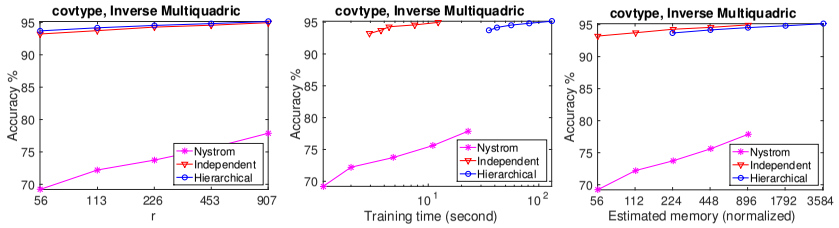
Figure 11: Performance versus r, time, and memory. Inverse multiquadric kernel.



(a) mnist, multiclass classification



(b) acoustic, multiclass classification



(c) covtype, multiclass classification

Figure 12: (Continued) Performance versus r, time, and memory. Inverse multiquadric kernel.

Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 8(9):509–517, 1975.

B. Boots, A. Gretton, and G. J. Gordon. Hilbert space embeddings of predictive state representations. In *Proc. 29th Intl. Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2013.

Jie Chen. On the use of discrete Laplace operator for preconditioning kernel matrices. *SIAM Journal on Scientific Computing*, 35(2):A577–A602, 2013.

Jie Chen. Computing square root factorization for recursively low-rank compressed matrices. Technical Report RC25499, IBM Thomas J. Watson Research Center, 2014a.

Jie Chen. Data structure and algorithms for recursively low-rank compressed matrices. Technical Report ANL/MCS-P5112-0314, Argonne National Laboratory, 2014b.

Jie Chen, Haw-Ren Fang, and Yousef Saad. Fast approximate kNN graph construction for high dimensional data via recursive Lanczos bisection. *Journal of Machine Learning Research*, 10(Sep):1989–2012, 2009.

Jean-Paul Chilès and Pierre Delfiner. *Geostatistics: Modeling Spatial Uncertainty*. Wiley, 2012.

Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2002.

Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.

Petros Drineas and Michael W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6:2153–2175, 2005.

Reinhard Furrer, Marc G Genton, and Douglas Nychka. Covariance tapering for interpolation of large spatial datasets. *Journal of Computational and Graphical Statistics*, 15(3): 502–523, 2006.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

Kazushige Goto and Robert Van De Geijn. High-performance implementation of the level-3 BLAS. *ACM Transactions on Mathematical Software*, 35(1), 2008.

W. Hackbusch. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. part I: Introduction to $\mathcal{H}$-matrices. *Computing*, 62(2):89–108, 1999.

Z. Harchaoui, F. Bach, O. Cappe, and E. Moulines. Kernel-based methods for hypothesis testing: A unified view. *Signal Processing Magazine, IEEE*, 30(4):87–97, July 2013.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, second edition, 2009.

Roger A. Horn and Charles R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1994.

P. Huang, H. Avron, T. N. Sainath, V. Sindhwani, and B. Ramabhadran. Kernel methods match deep neural networks on TIMIT. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014.

William B. Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *Conference on Modern Analysis and Probability*, volume 26, pages 189–206. American Mathematical Society, 1984.

Cari Kaufman, Mark Schervish, and Douglas Nychka. Covariance tapering for likelihood-based estimation in large spatial data sets. *Journal of the American Statistical Association*, 103:1545–1555, 2008.

J. R. Koehler and A. B. Owen. *Handbook of statistics*, volume 13, chapter 9 Computer Experiments, pages 261–308. Elsevier B.V., 1996.

J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. Fifth Berkeley Symp. on Math. Statist. and Prob.*, volume 1, pages 281–297, 1967.

William B. March, Bo Xiao, and George Biros. ASKIT: Approximate skeletonization kernel-independent treecode in high dimensions. Preprint arXiv:1410.0260, 2014.

Charles A. Micchelli. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constructive Approximation*, 2(1):11–22, 1986.

J. J. Monaghan and J. C. Lattanzio. A refined particle method for astrophysical problems. *Astronomy and Astrophysics*, 149(1):135–143, 1985.

A. O'Hagan and J. F. C. Kingman. Curve fitting and optimal design for prediction. *Journal of the Royal Statistical Society. Series B (Methodological)*, 40(1):1–42, 1978.

K Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(11):559–572, 1901.

Ali Rahimi and Ben Recht. Random features for large-scale kernel machines. In *Neural Infomration Processing Systems*, 2007.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, second edition, 2003.

Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press, 2001.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A generalized representer theorem. *Lecture Notes in Computer Science*, 2111:416426, 2001.

S. Si, C.-J. Hsieh, and I.S. Dhillon. Memory efficient kernel approximation. In *Proceedings of the 25th international conference on Machine learning*, 2014.

Si Si, Cho-Jui Hsieh, and Inderjit S. Dhillon. Memory efficient kernel approximation. *Journal of Machine Learning Research*, 18(20):1–32, 2017.

Ralph C. Smith. *Uncertainty Quantification: Theory, Implementation, and Applications*. Society for Industrial and Applied Mathematics, 2013.

L Song, B. Boots, S. Siddiqi, G. Gordon, and A. Smola. Hilbert space embeddings of hidden Markov models. In *Proc. of the 30th International Conference on Machine Learning (ICML)*, 2013.

Michael L. Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer, 1999.

Michael L. Stein. Limitations on low rank approximations for covariance matrices of spatial data. *Spatial Statistics*, 8:1–19, 2014.

Michael L. Stein, Jie Chen, and Mihai Anitescu. Stochastic approximation of score functions for Gaussian processes. *Annals of Applied Statistics*, 7(2):1162–1191, 2013.

Xiaobai Sun and Nikos P. Pitsianis. A matrix version of the fast multipole method. *SIAM Review*, 43(2):289–300, 2001.

Holger Wendland. *Scattered Data Approximation*. Cambridge University Press, 2004.

Christopher K. I. Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems 13*, 2000.

J. Yang, V. Sindhwani, Q. Fan, H. Avron, and M. Mahoney. Random Laplace feature maps for semigroup kernels on histograms. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

Tianbao Yang, Yu feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi-Hua Zhou. Nyström method vs random Fourier features: A theoretical and empirical comparison. In *Advances in Neural Information Processing Systems 25*, 2012.

I. Yen, T. Lin, S. Lin, P. Ravikumar, and I. Dhillon. Sparse random feature algorithm as coordinate descent in Hilbert space. In *Neural Information Processing Systems*, pages 2456–2464, 2014.

Chenhan D. Yu, William B. March, and George Biros. An $n \log n$ parallel fast direct solver for kernel matrices. Preprint arXiv:1701.02324, 2017.

K. Zhang, J. Peters, D. Janzing, and B. Scholkopf. Kernel based conditional independence test and application in causal discovery. In *Proc. 27th Intl. Conf. on Uncertainty in Artificial Intelligence*, 2011.

Kai Zhang and James T. Kwok. Clustered Nyström method for large scale manifold learning and dimension reduction. *IEEE Transactions on Neural Networks*, 21(10):1576–1587, 2010.

Kai Zhang, Ivor W. Tsang, and James T. Kwok. Improved Nyström low-rank approximation and error analysis. In *Proceedings of the 25th international conference on Machine learning*, 2008.

Journal of Machine Learning Research 18 (2017) 1-29 Submitted 9/15; Revised 6/16; Published 8/17

Sharp Oracle Inequalities for Square Root Regularization

Benjamin Stucky stucky@stat.math.ethz.ch
Seminar for Statistics
ETH Zürich
Rämistrasse 101
8092 Zurich, Switzerland

Sara van de Geer geer@stat.math.ethz.ch
Seminar for Statistics
ETH Zürich
Rämistrasse 101
8092 Zurich, Switzerland

Editor: Nicolas Vayatis

Abstract

We study a set of regularization methods for high-dimensional linear regression models. These penalized estimators have the square root of the residual sum of squared errors as loss function, and any weakly decomposable norm as penalty function. This fit measure is chosen because of its property that the estimator does not depend on the unknown standard deviation of the noise. On the other hand, a generalized weakly decomposable norm penalty is very useful in being able to deal with different underlying sparsity structures. We can choose a different sparsity inducing norm depending on how we want to interpret the unknown parameter vector β. Structured sparsity norms, as defined in Micchelli et al. (2010), are special cases of weakly decomposable norms, therefore we also include the square root LASSO (Belloni et al., 2011), the group square root LASSO (Bunea et al., 2014) and a new method called the square root SLOPE (in a similar fashion to the SLOPE from Bogdan et al. 2015). For this collection of estimators our results provide sharp oracle inequalities with the Karush-Kuhn-Tucker conditions. We discuss some examples of estimators. Based on a simulation we illustrate some advantages of the square root SLOPE.

Keywords: Square root LASSO, structured sparsity, Karush-Kuhn-Tucker, sharp oracale inequality, weak decomposability

1. Introduction and Model

The recent development of new technologies makes data gathering not a big problem any more. In some sense there is more data than we can handle, or than we need. The problem has shifted towards finding useful and meaningful information in the big sea of data. An example where such problems arise is the high-dimensional linear regression model

$$Y = X\beta^0 + \epsilon. \tag{1.1}$$

Here Y is the $n-$dimensional response variable, X is the $n \times p$ design matrix and ϵ is the identical and independent distributed noise vector. The noise has $\mathrm{E}(\epsilon_i) = 0, \mathrm{Var}(\epsilon_i) = \sigma^2, \ \forall i \in \{1, ..., n\}$. Assume that σ is **unknown**, and that β^0 is the "true" underlying $p-$dimensional parameter vector of the linear regression model with active set $S_0 := \mathrm{supp}(\beta^0)$.

While trying to explain Y through different other variables, in the high-dimensional linear regression model, we need to set less important explanatory variables to zero. Otherwise we would have overfitting. This is the process of finding a trade-off between a good fit and a sparse solution. In other words we are trying to find a solution that explains our data well, but at the same time only uses more important variables to do so.
The most famous and widely used estimator for the high-dimensional regression model is the ℓ_1-regularized version of least squares, called LASSO (Tibshirani, 1996)

$$\hat{\beta}_L(\sigma) := \arg\min_{\beta \in \mathbb{R}^p} \left\{ \|Y - X\beta\|_n^2 + 2\lambda_1 \sigma \|\beta\|_1 \right\}.$$

Here λ_1 is a constant called the regularization level, which regulates how sparse our solution should be. Also note that the construction of the LASSO estimator depends on the unknown noise level σ. We moreover let $\|a\|_1 := \sum_{i=1}^p |a_i|$ for any $a \in \mathbb{R}^p$ denote the ℓ_1-norm and for any $a \in \mathbb{R}^n$ we write $\|a\|_n^2 = \sum_{j=1}^n a_j^2/n$, the ℓ_2- norm squared and divided by n. The LASSO uses the ℓ_1-norm as a measure of sparsity. This measure as regulizer sets a number of parameters to zero.
Let us rewrite the LASSO into the following form

$$\hat{\beta}_L = \arg\min_{\beta \in \mathbb{R}^p} \left\{ \left(\|Y - X\beta\|_n + \lambda'(\beta)\|\beta\|_1 \right) \cdot \frac{2\lambda_1 \sigma}{\lambda'(\beta)} \right\},$$

where $\lambda'(\beta) := \frac{2\lambda_1 \sigma}{\|Y - X\beta\|_n}$. Instead of minimizing with $\lambda'(\beta)$, a function of β, let us assume that we keep $\lambda'(\beta)$ a fixed constant. Then we get the Square Root LASSO method

$$\hat{\beta}_{srL} := \arg\min_{\beta \in \mathbb{R}^p} \left\{ \|Y - X\beta\|_n + \lambda \|\beta\|_1 \right\}.$$

So in some sense the λ for the Square Root LASSO is a scaled version, scaled by an adaptive estimator of σ, of λ_1 from the LASSO. By the optimality conditions it is true that

$$\hat{\beta}_L(\|Y - X\hat{\beta}_{srL}\|_n) = \hat{\beta}_{srL}.$$

The Square Root LASSO was introduced by Belloni et al. (2011) in order to get a pivotal method. An equivalent formulation as a joint convex optimization program can be found in Owen (2007). This method has been studied under the name Scaled LASSO in Sun and Zhang (2012). Pivotal means that the theoretical λ does not depend on the unknown standard deviation σ or on any estimated version of it. The estimator does not require the estimation of the unknown σ. Belloni et al. (2014) also showed that under Gaussian noise the theoretical λ can be chosen of order $\Phi^{-1}(1 - \alpha/2p)/\sqrt{n-1}$, with Φ^{-1} denoting the inverse of the standard Gaussian cumulative distribution function, and α being some small probability. This is independent of σ and achieves a near oracle inequality for the prediction norm of convergence rate $\sigma\sqrt{(|S_0|/n)\log p}$. In contrast to that, the theoretical penalty level of the LASSO depends on knowing σ in order to achieve similar oracle inequalities for the prediction norm.

©2017 Benjamin Stucky and Sara van de Geer.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-482.html.

The idea of the square root LASSO was further developed in Bunea et al. (2014) to the group square root LASSO, in order to get a selection of groups of predictors. The group LASSO norm is another way to describe an underlying sparsity, namely if groups of parameters should be set to zero, instead of individual parameters. Another extension for the the square root LASSO in the case of matrix completion was given by Klopp (2014).
Now in this paper we go further and generalize the idea of the square root LASSO to any sparsity inducing norm. From now on we will look at the family of norm penalty regularization methods, which are of the following square root type

$$\hat{\beta} := \arg\min_{\beta \in \mathbb{R}^p} \{\|Y - X\beta\|_n + \lambda\Omega(\beta)\},$$

where Ω is any norm on $\mathbb{R}^p$. This set of regularization methods will be called square root regularization methods. Furthermore, we introduce the following notations

$\hat{\epsilon} := Y - X\hat{\beta}$	the residuals,
$\Omega^*(x) := \max_{z,\Omega(z)\leq 1} z^T x,\ x \in \mathbb{R}^p$	the dual norm of the norm Ω, and
$\beta_S = \{\beta_j : j \in S\}$	$\forall S \subset \{1, ..., p\}$ and all vectors $\beta \in \mathbb{R}^p$.

Later we will see that describing the underlying sparsity with an appropriate sparsity norm can make a difference in how good the errors will be. Therefore in this paper we extend the idea of the square root LASSO with the ℓ_1-penalty to more general weakly decomposable norm penalties. The theoretical λ of such an estimator will not depend on σ either. We introduce the Karush-Kuhn-Tucker conditions for these estimators and give sharp oracle inequalities. In the last two sections we will give some examples of different norms and simulations comparing the square root LASSO with the square root SLOPE.

2. Karush-Kuhn-Tucker Conditions

As we already have seen before, these estimators need to calculate a minimum over β. The Karush-Kuhn-Tucker conditions characterize this minimum. In order to formulate these optimality conditions we need some concepts of convex optimization. For the reader who is not familiar with this topic, we will introduce the subdifferential, which generalizes the differential, and give a short overview of some properties, as can be found for example in Bach et al. (2012). For any convex function $g : \mathbb{R}^p \to \mathbb{R}$ and any vector $w \in \mathbb{R}^p$ we define its subdifferential as

$$\partial g(w) := \{z \in \mathbb{R}^p;\ g(w') \geq g(w) + z^T(w' - w)\ \forall w' \in \mathbb{R}^p\}.$$

The elements of $\partial g(w)$ are called the subgradients of g at w.
Let us remark that all convex functions have non empty subdifferentials at every point. Moreover by the definition of the subdifferential any subgradient defines a tangent space $w' \mapsto g(w) + z^T \cdot (w' - w)$, that goes through $g(w)$ and is at any point lower than the function g. If g is differentiable at w, then its subdifferential at w is the usual gradient. Now the

next lemma, which dates back to Pierre Fermat (see Bauschke and Combettes 2011), shows how to find a global minimum for a convex function g.

Lemma 1 (Fermat's Rule) *For all convex functions $g : \mathbb{R}^p \to \mathbb{R}$ it holds that*

$$v \in \mathbb{R}^p \text{ is a global minimum of } g \ \Leftrightarrow 0 \in \partial g(v).$$

For any norm Ω on $\mathbb{R}^p$ with $\omega \in \mathbb{R}^p$ it holds true that its subdifferential can be written as (see Bach et al. 2012 Proposition 1.2)

$$\partial\Omega(\omega) = \begin{cases} \{z \in \mathbb{R}^p; \Omega^*(z) \leq 1\} & \text{if } \omega = 0 \\ \{z \in \mathbb{R}^p; \Omega^*(z) = 1 \bigwedge z^T w = \Omega(\omega)\} & \text{if } \omega \neq 0. \end{cases} \tag{2.1}$$

We are able to apply these properties to our estimator $\hat{\beta}$. Lemma 1 implies that

$$\hat{\beta} \text{ is optimal } \Leftrightarrow -\frac{1}{\lambda}\nabla\|Y - X\hat{\beta}\|_n \in \partial\Omega(\hat{\beta}).$$

This means that, in the case $\|\hat{\epsilon}\|_n > 0$, for the square root regularization estimator $\hat{\beta}$ it holds true that

$$\hat{\beta} \text{ is optimal } \Leftrightarrow \frac{X^T(Y - X\hat{\beta})}{n\lambda\|Y - X\hat{\beta}\|_n} \in \partial\Omega(\hat{\beta}). \tag{2.2}$$

By combining equation (2.1) with (2.2) we can write the KKT conditions as

$$\hat{\beta} \text{ is optimal } \Leftrightarrow \begin{cases} \Omega^*\left(\frac{\hat{\epsilon}^T X}{n\|\hat{\epsilon}\|_n}\right) \leq \lambda & \text{if } \hat{\beta} = 0 \\ \Omega^*\left(\frac{\hat{\epsilon}^T X}{n\|\hat{\epsilon}\|_n}\right) = \lambda & \text{if } \hat{\beta} \neq 0. \\ \bigwedge \frac{\hat{\epsilon}^T X\hat{\beta}}{n\|\hat{\epsilon}\|_n} = \lambda\Omega(\hat{\beta}) \end{cases} \tag{2.3}$$

What we might first remark about equation (2.3) is that in the case of $\hat{\beta} \neq 0$ the second part can be written as

$$\hat{\epsilon}^T X\hat{\beta}/n = \Omega(\hat{\beta}) \cdot \Omega^*\left(\frac{\hat{\epsilon}^T X}{n}\right).$$

This means that we in fact have equality in the generalized Cauchy-Schwartz Inequality for these two $p-$dimensional vectors. Furthermore let us remark that the equality

$$\hat{\epsilon}^T X\hat{\beta}/n = \Omega(\hat{\beta})\lambda\|\hat{\epsilon}\|_n$$

trivially holds true for the case where $\hat{\beta} = 0$. It is important to remark here that, in contrast to the KKT conditions for the LASSO, we have an additional $\|\hat{\epsilon}\|_n$ term in the expression $\Omega^*\left(\frac{\hat{\epsilon}^T X}{n\|\hat{\epsilon}\|_n}\right)$. This nice scaling leads to the property that the theoretical λ is independent of σ.
With the KKT conditions we are able to formulate a generalized type of KKT conditions. This next lemma is needed for the proofs in the next chapter.

Lemma 2 *For the square root type estimator $\hat{\beta}$ we have for any $\beta \in \mathbb{R}^p$ and when $\|\hat{\epsilon}\|_n \neq 0$*

$$\frac{1}{\|\hat{\epsilon}\|_n}\hat{\epsilon}^T X(\beta - \hat{\beta})/n + \lambda\Omega(\hat{\beta}) \leq \lambda\Omega(\beta).$$

Proof First we need to look at the inequality from the KKT's, which holds in any case

$$\Omega^*\left(\frac{\hat{\epsilon}^T X}{n\|\hat{\epsilon}\|_n}\right) \leq \lambda. \tag{2.4}$$

And by the definition of the dual norm and the maximum, we have with (2.4)

$$\begin{aligned}\frac{1}{\|\hat{\epsilon}\|_n}\hat{\epsilon}^T X\beta/n &\leq \Omega(\beta)\cdot \max_{\beta\in\mathbb{R}^p,\Omega(\beta)\leq 1}\frac{\hat{\epsilon}^T}{\|\hat{\epsilon}\|_n}X\beta/n \\ &= \Omega(\beta)\cdot\Omega^*\left(\frac{\hat{\epsilon}^T X}{n\|\hat{\epsilon}\|_n}\right) \\ &\leq \Omega(\beta)\lambda. \end{aligned}\tag{2.5}$$

The second equation from the KKT's, which again holds in any case, is

$$\frac{1}{\|\hat{\epsilon}\|_n}\hat{\epsilon}^T X\hat{\beta}/n = \lambda\Omega(\hat{\beta}). \tag{2.6}$$

Now putting (2.5) and (2.6) together we get the result.

■

3. Sharp Oracle Inequalities for the square root regularization estimators

We provide sharp oracle inequalities for the estimator $\hat{\beta}$ with a norm Ω that satisfies a so called weak decomposability condition. An oracle inequality is a bound on the estimation and prediction errors. This shows how good these estimators are in estimating the parameter vector β^0. This is an extension of the sharp oracle results given in van de Geer (2014) for LASSO type of estimators, which in turn was an generalization of the sharp oracle inequalities for the LASSO and nuclear norm penalization in Koltchinskii (2011) and Koltchinskii et al. (2011).

3.1 Concepts and Assumptions

Let us first introduce all the necessary definitions and concepts. Some normed versions of values need to be introduced:

$$\begin{aligned} f &= \frac{\lambda\Omega(\beta^0)}{\|\epsilon\|_n} \\ \lambda^{S^c} &= \frac{\Omega^{S^c *}((\epsilon^T X)_{S^c})}{n\|\epsilon\|_n} \\ \lambda^S &= \frac{\Omega^*((\epsilon^T X)_S)}{n\|\epsilon\|_n} \\ \lambda^m &= \max(\lambda^S, \lambda^{S^c}) \\ \lambda^0 &= \frac{\Omega^*(\epsilon^T X)}{n\|\epsilon\|_n}. \end{aligned}$$

For example the quantity f gives the measure of the true underlying normalized sparsity. Ω^{S^c} denotes a norm on $\mathbb{R}^{p-|S|}$ which will shortly be defined in Assumption II. Furthermore λ^m will take the role of the theoretical (unknown) λ. If we compare this to the case of the LASSO we see that instead of the $\ell_\infty-$norm we generalized it to the dual norm of Ω. Also remark that in λ^m a term $\frac{1}{\|\epsilon\|_n}$ appears. This scaling is due to the square root regularization, which will be the reason that λ can be chosen independently of the unknown standard deviation σ. Now we will give the two main assumptions that need to hold in order to prove the oracle inequalities. Assumption I deals with avoiding overfitting, and the main concern of Assumption II is that the norm has the desired property of promoting a structured sparse solution $\hat{\beta}$. We will later see, that the structured sparsity norms in Micchelli et al. (2010) and Micchelli et al. (2013) are all of this form. Thus, Assumption II is quite general.

3.1.1 Assumption I (overfitting)

If $\|\hat{\epsilon}\|_n = 0$, then $\hat{\beta}$ does the same thing as the Ordinary Least Squares (OLS) estimator β_{OLS}, namely it overfits. That is why we need a lower bound on $\|\hat{\epsilon}\|_n$. In order to achieve this lower bound we make the following assumptions:

$$P\left(Y \in \{\widetilde{Y} : \min_{\beta,\text{s.t.}X\beta=\widetilde{Y}}\Omega(\beta) \leq \|\epsilon\|_n\}\right) = 0.$$

$$\frac{\lambda^0}{\lambda}\big(1+2f\big) < 1.$$

The $\frac{\lambda^0}{\lambda}$ term makes sure that we introduce enough sparsity (no overfitting).

3.1.2 Assumption II (weak decomposability)

Assumption II is fulfilled for a set $S \subset \{1,...,p\}$ and a norm Ω on $\mathbb{R}^p$ if this norm is weakly decomposable, and S is an allowed set for this norm. This was used by van de Geer (2014) and goes back to Bach et al. (2012). It is an assumption on the structure of the sparsity inducing norm. By the triangle inequality we have:

$$\Omega(\beta_{S^c}) \geq \Omega(\beta) - \Omega(\beta_S).$$

But we will also need to lower bound this by another norm evaluated at β_{S^c}. This is motivated by relaxing the following decomposability property of the ℓ_1-norm:

$$\|\beta\|_1 = \|\beta_S\|_1 + \|\beta_{S^c}\|_1, \forall \text{ sets } S \subset 1,...,p \text{ and all } \beta \in \mathbb{R}^p.$$

This decomposability property is used to get oracle inequalities for the LASSO. But we can relax this property, and introduce weakly decomposable norms.

Definition 3 (Weak decomposability) *A norm Ω in $\mathbb{R}^p$ is called weakly decomposable for an index set $S \subset \{1,...,p\}$, if there exists a norm Ω^{S^c} on $\mathbb{R}^{|S^c|}$ such that*

$$\forall\beta \in \mathbb{R}^p \quad \Omega(\beta) \geq \Omega(\beta_S) + \Omega^{S^c}(\beta_{S^c}).$$

Furthermore we call a set S allowed if Ω is a weakly decomposable norm for this set.

Remark 4 *In order to get a good oracle bound, we will choose the norm Ω^{S^c} as large as possible. We will also choose the allowed sets S in such a way to reflect the active set S_0. Otherwise we would of course be able to choose as a trivial example the empty set $S = \varnothing$.*

Now that we have introduced the two main assumptions, we can introduce other definitions and concepts also used in van de Geer (2014).

Definition 5 *For S an allowed set of a weakly decomposable norm Ω, and $L > 0$ a constant, the $\Omega-$eigenvalue is defined as*

$$\delta_\Omega(L,S) := \min\left\{\|X\beta_S - X\beta_{S^c}\|_n : \Omega(\beta_S) = 1, \Omega^{S^c}(\beta_{S^c}) \leq L\right\}.$$

Then the $\Omega-$effective sparsity is defined as

$$\Gamma^2_\Omega(L,S) := \frac{1}{\delta^2_\Omega(L,S)}.$$

The $\Omega-$eigenvalue is the distance between the two sets (van de Geer and Lederer, 2013) $\{X\beta_S : \Omega(\beta_S) = 1\}$ and $\{X\beta_{S^c} : \Omega^{S^c}(\beta_{S^c}) \leq L\}$, see Figure 2. The additional discussion about these definitions will follow after the main theorem. The $\Omega-$eigenvalue generalizes the compatibility constant (van de Geer, 2007).
For the proof of the main theorem we need some small lemmas. For any vector $\beta \in \mathbb{R}^p$ the $(L,S)-$cone condition for a norm Ω is satisfied if $\Omega^{S^c}(\beta_{S^c}) \leq L\Omega(\beta_S)$, with $L > 0$ a constant and S an allowed set.
The proof of Lemma 6 can be found in van de Geer (2014). It shows the connection between the $(L,S)-$cone condition and the $\Omega-$eigenvalue. We bound $\Omega(\beta_S)$ by a multiple of $\|X\beta\|_n$.

Lemma 6 *Let S be an allowed set of a weakly decomposable norm Ω and $L > 0$ a constant. Then we have that the $\Omega-$eigenvalue is of the following form:*

$$\delta_\Omega(L,S) = \min\left\{\frac{\|X\beta\|_n}{\Omega(\beta_S)}, \beta \text{ satisfies the cone condition and } \beta_S \neq 0\right\}.$$

We have $\Omega(\beta_S) \leq \Gamma_\Omega(L,S)\|X\beta\|_n$.

We will also need a lower and an upper bound for $\|\hat\epsilon\|_n$, as already mentioned in Assumption I. The next Lemma 7 gives such bounds.

Lemma 7 *Suppose that Assumption I holds true. Then*

$$1 + f \geq \frac{\|\hat\epsilon\|_n}{\|\epsilon\|_n} \geq \frac{1 - \frac{\lambda^0}{\lambda}(1+2f)}{f+2} > 0.$$

Proof The upper bound is obtained by the definition of the estimator

$$\|Y - X\hat\beta\|_n + \lambda\Omega(\hat\beta) \leq \|Y - X\beta^0\|_n + \lambda\Omega(\beta^0).$$

Therefore we get

$$\|\hat\epsilon\|_n \leq \|\epsilon\|_n + \lambda\Omega(\beta^0).$$

Dividing by $\|\epsilon\|_n$ and by the definition of f we get the desired upper bound. The main idea for the lower bound is to use the triangle inequality

$$\|\hat\epsilon\|_n = \|\epsilon - X(\hat\beta - \beta^0)\|_n \geq \|\epsilon\|_n - \|X(\hat\beta - \beta^0)\|_n,$$

and then upper bound $\|X(\hat\beta - \beta^0)\|_n$. With Lemma 2 we get an upper bound for $\|X(\hat\beta - \beta^0)\|_n$,

$$\begin{aligned}
\|X(\hat\beta - \beta^0)\|_n^2 &\leq \epsilon^T X(\hat\beta - \beta^0)/n + \lambda\|\hat\epsilon\|_n(\Omega(\beta^0) - \Omega(\hat\beta)) \\
&\leq \lambda^0\|\epsilon\|_n\Omega(\hat\beta - \beta^0) + \lambda\|\hat\epsilon\|_n(\Omega(\beta^0) - \Omega(\hat\beta)) \\
&\leq \lambda^0\|\epsilon\|_n(\Omega(\hat\beta) + \Omega(\beta^0)) + \lambda\|\hat\epsilon\|_n(\Omega(\beta^0) - \Omega(\hat\beta)) \\
&\leq \lambda^0\|\epsilon\|_n\Omega(\hat\beta) + \Omega(\beta^0)(\lambda^0\|\epsilon\|_n + \lambda\|\hat\epsilon\|_n).
\end{aligned}$$

In the second line we used the definition of the dual norm, and the Cauchy-Schwartz inequality. Again by the definition of the estimator we have

$$\Omega(\hat\beta) \leq \frac{\|\epsilon\|_n}{\lambda} + \Omega(\beta^0).$$

And we are left with

$$\|X(\hat\beta - \beta^0)\|_n \leq \|\epsilon\|_n\sqrt{\frac{\lambda^0}{\lambda}\left(1 + 2\frac{\lambda\Omega(\beta^0)}{\|\epsilon\|_n} + \frac{\lambda}{\lambda^0}\cdot\frac{\|\hat\epsilon\|_n}{\|\epsilon\|_n}\cdot\frac{\lambda\Omega(\beta^0)}{\|\epsilon\|_n}\right)}.$$

By the definition of f we get

$$\begin{aligned}
\|X(\hat\beta - \beta^0)\|_n &\leq \|\epsilon\|_n\sqrt{\frac{\lambda^0}{\lambda}\left(1 + 2f + \frac{\lambda}{\lambda^0}\frac{\|\hat\epsilon\|_n}{\|\epsilon\|_n}f\right)}. \\
&= \|\epsilon\|_n\sqrt{\frac{\lambda^0}{\lambda} + 2\frac{\lambda^0}{\lambda}f + \frac{\|\hat\epsilon\|_n}{\|\epsilon\|_n}f}.
\end{aligned}$$

Now we get

$$\begin{aligned}
\|\hat\epsilon\|_n &\geq \|\epsilon\|_n - \|X(\hat\beta - \beta^0)\|_n \\
&\geq \|\epsilon\|_n - \|\epsilon\|_n\sqrt{\frac{\lambda^0}{\lambda} + 2\frac{\lambda^0}{\lambda}f + \frac{\|\hat\epsilon\|_n}{\|\epsilon\|_n}f}
\end{aligned} \tag{3.1}$$

Let us rearrange equation (3.1) further in the case $\frac{\|\hat\epsilon\|_n}{\|\epsilon\|_n} < 1$

$$\frac{\lambda^0}{\lambda} + 2\frac{\lambda^0}{\lambda}f + \frac{\|\hat\epsilon\|_n}{\|\epsilon\|_n}f \geq \left(1 - \frac{\|\hat\epsilon\|_n}{\|\epsilon\|_n}\right)^2$$

$$\frac{\|\hat\epsilon\|_n}{\|\epsilon\|_n}f \geq 1 - 2\frac{\|\hat\epsilon\|_n}{\|\epsilon\|_n} + \frac{\|\hat\epsilon\|_n^2}{\|\epsilon\|_n^2} - \frac{\lambda^0}{\lambda}(1+2f)$$

$$\frac{\|\hat{\epsilon}\|_n}{\|\epsilon\|_n} f + 2\frac{\|\hat{\epsilon}\|_n}{\|\epsilon\|_n} \geq 1 - \frac{\lambda^0}{\lambda}(1+2f)$$

$$\frac{\|\hat{\epsilon}\|_n}{\|\epsilon\|_n} \geq \frac{1-\frac{\lambda^0}{\lambda}(1+2f)}{f+2} \overset{\text{Assumption I}}{>} 0.$$

On the other hand if $\frac{\|\hat{\epsilon}\|_n}{\|\epsilon\|_n} > 1$, we already get a lower bound which is bigger than $\frac{1-\frac{\lambda^0}{\lambda}(1+2f)}{f+2}$. ■

3.2 Sharp Oracle Inequality

Finally we are able to present the main theorem. This theorem gives sharp oracle inequalities on the prediction error expressed in the ℓ_2-norm, and the estimation error expressed in the Ω and Ω^{S^c} norms.

Remark 8 *Let us first briefly remark that in the Theorem 9 we need to assure that $\lambda^* - \lambda^m > 0$. The assumption $\frac{\lambda^m}{\lambda} < 1/a$, with a chosen as in Theorem 9, together with the fact that $\lambda^0 \leq \lambda^m$ leads to the desired inequality*

$$\frac{\lambda^*}{\lambda} = \frac{1-\frac{\lambda^0}{\lambda}(1+2f)}{f+2} \geq \frac{1-\frac{\lambda^m}{\lambda}(1+2f)}{f+2} > \frac{\lambda^m}{\lambda}.$$

Theorem 9 *Assume that $0 \leq \delta < 1$, and also that $a\lambda^m < \lambda$, with the constant $a = 3(1+f)$. We invoke also Assumption I (overfitting) and Assumption II (weak decomposability) for S and Ω. Here the allowed set S is chosen such that the active set $S_\beta := \text{supp}(\beta)$ is a subset of S. Then it holds true that*

$$\begin{aligned}
&\|X(\hat{\beta}-\beta^0)\|_n^2 + 2\delta\|\epsilon\|_n \left[(\lambda^*+\lambda^m)\Omega(\hat{\beta}_S-\beta) + (\lambda^*-\lambda^m)\Omega^{S^c}(\hat{\beta}_{S^c})\right] \\
&\leq \|X(\beta-\beta^0)\|_n^2 + \|\epsilon\|_n^2 \left[(1+\delta)(\tilde{\lambda}+\lambda^m)\right]^2 \Gamma_\Omega^2(L_S, S), \qquad (3.2)
\end{aligned}$$

with $L_S := \frac{\tilde{\lambda}+\lambda^m}{\lambda^*-\lambda^m}\frac{1+\delta}{1-\delta}$ and

$$\lambda^* := \lambda\left(\frac{1-\frac{\lambda^0}{\lambda}(1+2f)}{f+2}\right), \qquad \tilde{\lambda} \quad := \lambda(1+f).$$

Furthermore we get the two oracle inequalities

$$\begin{aligned}
\|X(\hat{\beta}-\beta^0)\|_n^2 &\leq \|X(\beta_\star-\beta^0)\|_n^2 + \|\epsilon\|_n^2(1+\delta)^2(\tilde{\lambda}+\lambda^{S_\star^c})^2 \cdot \Gamma_\Omega^2(L_{S_\star}, S_\star) \\
\Omega(\hat{\beta}_{S_\star}-\beta_\star) + \Omega^{S_\star^c}(\hat{\beta}_{S_\star^c}) &\leq \frac{1}{2\delta\|\epsilon\|_n} \cdot \frac{\|X(\beta_\star-\beta^0)\|_n^2}{\lambda^*-\lambda^m} + ... \\
&\quad + \frac{(1+\delta)^2\|\epsilon\|_n}{2\delta} \cdot \frac{(\tilde{\lambda}+\lambda^m)^2}{\lambda^*-\lambda^m} \cdot \Gamma_\Omega^2(L_{S_\star}, S_\star).
\end{aligned}$$

For all fixed allowed sets S define

$$\beta_\star(S) := \underset{\beta:\ \text{supp}(\beta)\subseteq S}{\arg\min} \left(\|X(\beta-\beta^0)\|_n^2 + \|\epsilon\|_n^2 \left[(1+\delta)(\tilde{\lambda}+\lambda^m)\right]^2 \Gamma_\Omega^2(L_S, S)\right).$$

Then $S_\star$ is defined as

$$S_\star := \underset{S\ \text{allowed}}{\arg\min} \left(\|X(\beta_\star(S)-\beta^0)\|_n^2 + \|\epsilon\|_n^2 \left[(1+\delta)(\tilde{\lambda}+\lambda^m)\right]^2 \Gamma_\Omega^2(L_S, S)\right), \quad (3.3)$$

$$\beta_\star := \beta_\star(S_\star) \quad (3.4)$$

it attains the minimal right hand side of the oracle inequality (3.2). An improtant special case of equation (3.2) is to choose $\beta \equiv \beta^0$ with $S \supseteq S_0$ allowed. The term $\|X(\beta-\beta^0)\|_n^2$ vanishes in this case and only the $\Omega-$effective sparsity term remains for the upper bound. But it is not obvious in which cases and whether $\beta_\star$ leads to a substantially lower bound than β^0.

Proof Let $\beta \in \mathbb{R}^p$ and let S be an allowed set containing the active set of β. We need to distinguish 2 cases. The second case is the more substantial one.
Case 1: Assume that

$$\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n \leq -\delta\|\epsilon\|_n \left[(\lambda^*+\lambda^m)\Omega(\hat{\beta}_S-\beta) + (\lambda^*-\lambda^m)\Omega^{S^c}(\hat{\beta}_{S^c})\right].$$

Here $\langle u, v\rangle_n := v^T u/n$, for any two vectors $u, v \in \mathbb{R}^n$. In this case we can simply use the following calculations to verify the theorem.

$$\begin{aligned}
&\|X(\hat{\beta}-\beta^0)\|_n^2 - \|X(\beta-\beta^0)\|_n^2 + ... \\
&\quad + 2\delta\|\epsilon\|_n \left[(\lambda^*+\lambda^m)\Omega(\hat{\beta}_S-\beta) + (\lambda^*-\lambda^m)\Omega^{S^c}(\hat{\beta}_{S^c})\right] \\
&\quad = 2\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n - \|X(\beta-\hat{\beta})\|_n^2 \\
&\quad\quad + 2\delta\|\epsilon\|_n \left[(\lambda^*+\lambda^m)\Omega(\hat{\beta}_S-\beta) + (\lambda^*-\lambda^m)\Omega^{S^c}(\hat{\beta}_{S^c})\right] \\
&\quad \leq -\|X(\beta-\hat{\beta})\|_n^2 \\
&\quad \leq 0
\end{aligned}$$

Now we can turn to the more important case.
Case 2: Assume that

$$\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n \geq -\delta\|\epsilon\|_n \left[(\lambda^*+\lambda^m)\Omega(\hat{\beta}_S-\beta) + (\lambda^*-\lambda^m)\Omega^{S^c}(\hat{\beta}_{S^c})\right].$$

We can reformulate Lemma 2 with $Y - X\hat{\beta} = X(\beta^0-\hat{\beta}) + \epsilon$, then we get:

$$\frac{\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n}{\|\hat{\epsilon}\|_n} + \lambda\Omega(\hat{\beta}) \leq \frac{\langle \epsilon, X(\hat{\beta}-\beta)\rangle_n}{\|\hat{\epsilon}\|_n} + \lambda\Omega(\beta).$$

This is equivalent to

$$\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n + \|\hat{\epsilon}\|_n\lambda\Omega(\hat{\beta}) \leq \langle \epsilon, X(\hat{\beta}-\beta)\rangle_n + \|\hat{\epsilon}\|_n\lambda\Omega(\beta). \quad (3.5)$$

By the definition of the dual norm and the generalized Cauchy-Schwartz inequality we have

$$\begin{aligned}\langle \epsilon, X(\hat{\beta}-\beta)\rangle_n &\leq \|\epsilon\|_n \left(\lambda^S \Omega(\hat{\beta}_S-\beta)+\lambda^{S^c}\Omega^{S^c}(\hat{\beta}_{S^c})\right)\\ &\leq \|\epsilon\|_n \left(\lambda^m \Omega(\hat{\beta}_S-\beta)+\lambda^m\Omega^{S^c}(\hat{\beta}_{S^c})\right)\end{aligned}$$

Inserting this inequality into (3.5) we get

$$\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n + \|\hat{\epsilon}\|_n\lambda\Omega(\hat{\beta}) \leq \|\epsilon\|_n \left(\lambda^m\Omega(\hat{\beta}_S-\beta)+\lambda^m\Omega^{S^c}(\hat{\beta}_{S^c})\right) + \|\hat{\epsilon}\|_n\lambda\Omega(\beta). \tag{3.6}$$

Then by the weak decomposability and the triangle inequality in (3.6)

$$\begin{aligned}&\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n + \|\hat{\epsilon}\|_n\lambda\left(\Omega(\hat{\beta}_S)+\Omega^{S^c}(\hat{\beta}_{S^c})\right)\\ &\leq \|\epsilon\|_n \left(\lambda^m\Omega(\hat{\beta}_S-\beta)+\lambda^m\Omega^{S^c}(\hat{\beta}_{S^c})\right) + \|\hat{\epsilon}\|_n\lambda\left(\Omega(\hat{\beta}_S-\beta)+\Omega(\hat{\beta}_S)\right).\end{aligned} \tag{3.7}$$

By inserting the assumption of case 2

$$\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n \geq -\delta\|\epsilon\|_n\left[(\lambda^*+\lambda^m)\Omega(\hat{\beta}_S-\beta)+(\lambda^*-\lambda^m)\Omega^{S^c}(\hat{\beta}_{S^c})\right],$$

into (3.7) we get

$$\left(\lambda\|\hat{\epsilon}\|_n-\lambda^m\|\epsilon\|_n-\delta\|\epsilon\|_n(\lambda^*-\lambda^m)\right)\Omega^{S^c}(\hat{\beta}_{S^c}) \leq \left(\lambda\|\hat{\epsilon}\|_n+\lambda^m\|\epsilon\|_n+\delta\|\epsilon\|_n(\tilde{\lambda}+\lambda^m)\right)\Omega(\hat{\beta}_S-\beta).$$

By assumption $a\lambda^m < \lambda$ we have that $\lambda^* > \lambda^m$ (see Remark 8) and therefore

$$\Omega^{S^c}(\hat{\beta}_{S^c}) \leq \left(\frac{\tilde{\lambda}+\lambda^m}{\lambda^*-\lambda^m}\right)\cdot\frac{1+\delta}{1-\delta}\cdot\Omega(\hat{\beta}_S-\beta).$$

We have applied Lemma 7 in the last step, in order to replace the estimate $\|\hat{\epsilon}\|_n$ with $\|\epsilon\|_n$. By the definition of L_S we have

$$\Omega^{S^c}(\hat{\beta}_{S^c}) \leq L_S\Omega(\hat{\beta}_S-\beta). \tag{3.8}$$

Therefore with Lemma 6 we get

$$\Omega(\hat{\beta}_S-\beta) \leq \Gamma_\Omega(L_S,S)\|X(\hat{\beta}-\beta)\|_n. \tag{3.9}$$

Inserting (3.9) into (3.7), together with Lemma 7 and $\delta<1$, we get

$$\begin{aligned}&\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n + \delta\|\epsilon\|_n(\lambda^*-\lambda^m)\Omega^{S^c}(\hat{\beta}_{S^c})\\ &\leq (1+\delta-\delta)\|\epsilon\|_n(\lambda\|\hat{\epsilon}\|_n/\|\epsilon\|_n+\lambda^m)\Omega(\hat{\beta}_S-\beta)\\ &\leq (1+\delta)\|\epsilon\|_n(\tilde{\lambda}+\lambda^m)\Gamma_\Omega(L_S,S)\|X(\hat{\beta}-\beta)\|_n - \delta\|\epsilon\|_n(\lambda^*+\lambda^m)\Omega(\hat{\beta}_S-\beta)\end{aligned}$$

Because $\forall u,v\in\mathbb{R},\ 0\leq(u-v)^2$ it holds true that $uv \leq 1/2(u^2+v^2)$.

Therefore with $a=(1+\delta)\|\epsilon\|_n(\tilde{\lambda}+\lambda^m)\Gamma_\Omega(L_S,S)$ and $b=\|X(\hat{\beta}-\beta)\|_n$ we have

$$\begin{aligned}&\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n + \delta\|\epsilon\|_n(\lambda^*-\lambda^m)\Omega(\hat{\beta}_{S^c})^{S^c} + \delta\|\epsilon\|_n(\lambda^*+\lambda^m)\Omega(\hat{\beta}_S-\beta)\\ &\leq \frac{1}{2}(1+\delta)^2\|\epsilon\|_n^2(\tilde{\lambda}+\lambda^m)^2\Gamma_\Omega^2(L_S,S)+\frac{1}{2}\|X(\hat{\beta}-\beta)\|_n^2.\end{aligned}$$

Since

$$2\langle X(\hat{\beta}-\beta^0), X(\hat{\beta}-\beta)\rangle_n = \|X(\hat{\beta}-\beta^0)\|_n^2 - \|X(\beta-\beta^0)\|_n^2 + \|X(\hat{\beta}-\beta)\|_n^2,$$

we get

$$\begin{aligned}&\|X(\hat{\beta}-\beta^0)\|_n^2 + 2\delta\|\epsilon\|_n\left((\lambda^*-\lambda^m)\Omega(\hat{\beta}_{S^c})^{S^c}+(\lambda^*+\lambda^m)\Omega(\hat{\beta}_S-\beta)\right)\\ &\leq (1+\delta)^2\|\epsilon\|_n^2(\tilde{\lambda}+\lambda^m)^2\Gamma_\Omega^2(L_S,S)+\|X(\beta-\beta^0)\|_n^2.\end{aligned} \tag{3.10}$$

This gives the sharp oracle inequality. The two oracle inequalities mentioned are just a split up version of inequality (3.10), where for the second oracle inequality we need to see that $\lambda^*-\lambda^m \leq \lambda^*+\lambda^m$. ■

Remark that the sharpness in the oracle inequality of Theorem 9 is the constant one in front of the term $\|X(\beta-\beta^0)\|_n^2$. Because we measure a vector on $S_\star$ by Ω and on the inactive set $S_\star^c$ by the norm $\Omega^{S_\star^c}$, we take here $\Omega(\hat{\beta}_{S_\star}-\beta_\star)$ and $\Omega^{S_\star^c}(\hat{\beta}_{S_\star^c})$ as estimation errors.
If we choose λ of the same order as λ^m (i.e. $a\lambda=\lambda^m$, with $a>0$ a constant), then we can simplify the oracle inequalities. This is comparable to the oracle inequalities for the LASSO, see for example Bickel et al. (2009), Bunea et al. (2006), Bunea et al. (2007), van de Geer (2007) and further references can be found in Bühlmann and van de Geer (2011).

Corollary 10 *Take λ of the order of λ^m (i.e. $\lambda^m = C\lambda$, with $0<C<\frac{1}{3(f+1)}$ a constant). Invoke the same assumptions as in Theorem 9. Here we also use the same notation of an optimal $\beta_\star$ with $S_\star$ as in equation* (3.3) *and* (3.4)*. Then we have*

$$\begin{aligned}\|X(\hat{\beta}-\beta^0)\|_n^2 &\leq \|X(\beta_\star-\beta^0)\|_n^2 + C_1\lambda^2\cdot\Gamma_\Omega^2(L_{S_\star},S_\star)\\ \Omega(\hat{\beta}_{S_\star}-\beta_\star)+\Omega^{S_\star^c}(\hat{\beta}_{S_\star^c}) &\leq C_2\left(\frac{\|X(\beta_\star-\beta^0)\|_n^2}{\lambda}+C_1\lambda\cdot\Gamma_\Omega^2(L_{S_\star},S_\star)\right).\end{aligned}$$

Here C_1 and C_2 are the constants:

$$C_1 := (1+\delta)^2\cdot\|\epsilon\|_n^2(f+C+1)^2, \tag{3.11}$$

$$C_2 := \frac{1}{2\delta\|\epsilon\|_n}\cdot\frac{1}{\sqrt{1-2C(1+2f)}-C}. \tag{3.12}$$

First let us explain some of the parts of Theorem 9 in more detail. We can also study what happens to the bound if we additionally assume Gaussian errors, see Proposition 11.

3.3 On the two parts of the oracle bound

The oracle bound is a trade-off between two parts, which we will discuss now. Let us first remember that if we set $\beta \equiv \beta^0$ in the sharp oracle bound, only the term with the $\Omega-$effective sparsity will not vanish on the right hand side of the bound. But due to the minimization over β in the definition of $\beta_\star$ we might even do better than that bound.
The first part consisting of minimizing $\|X(\beta - \beta^0)\|_n^2$ can be thought of the error made due to approximation, hence we call it the approximation error. If we fix the support S, which can be thought of being determined by the second part, then minimizing $\|X(\beta - \beta^0)\|_n^2$ is just a projection onto the subspace spanned by S, see Figure 1. So if S has a similar structure than the true unknown support S_0 of β^0, this will be small.

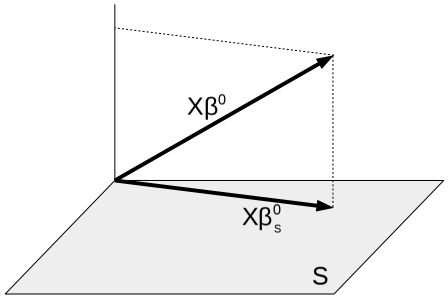


Figure 1: approximation error

The second part containing $\Gamma_\Omega^2(L_S, S)$ is due to estimation errors. There, minimizing over β will affect the set S. We have already mentioned that. It is one over the squared distance between the two sets $\{X\beta_S : \Omega(\beta_S) = 1\}$ and $\{X\beta_{S^c} : \Omega^{S^c}(\beta_{S^c}) \leq L\}$. Figure 2 shows this distance. This means that if the vectors in X_S and X_{S^c} show a high correlation the distance will shrink and the $\Omega-$effective sparsity will blow up, which we try to avoid. This distance depends also on the two chosen sparsity norms Ω and Ω^{S^c}. It is crucial to choose norms that reflect the true underlying sparsity in order to get a good bound. Also the constant L_S should be small.

3.4 On the randomness of the oracle bound

Until now, the bound still contains some random parts, for example in λ^m. In order to get rid of that random part we need to introduce the following sets

$$\mathcal{T} := \left\{ \max\left(\frac{\Omega^*((\epsilon^T X)_W)}{n\|\epsilon\|_n}, \frac{\Omega^{W^c*}((\epsilon^T X)_{W^c})}{n\|\epsilon\|_n} \right) \leq d \right\}, \text{ where } d \in \mathbb{R}, \text{ and any allowed set } W.$$

We need to choose the constant d in such a way, that we have a high probability for this set. In other words we try to bound the random part by a non random constant with a

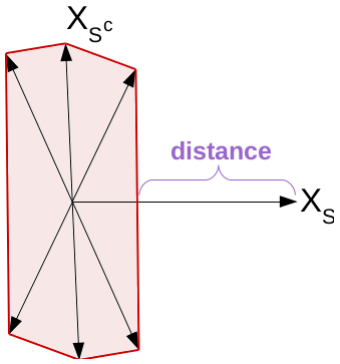


Figure 2: The Ω-eigenvalue

very high probability. In order to do this we need some assumptions on the errors. Here we assume Gaussian errors. Let us also remark that $\frac{\Omega^*((\epsilon^T X)_W)}{n\|\epsilon\|_n}$ is normalized by $\|\epsilon\|_n$. This normalization occurs due to the special form of the Karush-Kuhn-Tucker conditions. Thus the square root of the residual sum of squared errors is responsible for this normalization. In fact, this normalization is the main reason why λ does not contain the unknown variance. So the square root part of the estimator makes the estimator pivotal. Now in the case of Gaussian errors, we can use the concentration inequality from Theorem 5.8 in Boucheron et al. (2013) and get the following proposition. Define first:

$$\begin{array}{lllll} Z_1 := & \frac{\Omega^*((\epsilon^T X)_W)}{n\|\epsilon\|_n} & V_1 := & Z_1\|\epsilon\|_n/\sigma \\ Z_2 := & \frac{\Omega^{W^c*}((\epsilon^T X)_{W^c})}{n\|\epsilon\|_n} & V_2 := & Z_2\|\epsilon\|_n/\sigma \\ Z := & \max(Z_1, Z_2) & V := & \max(V_1, V_2) \end{array}$$

Proposition 11 *Suppose that we have i.i.d. Gaussian errors $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$, and that the following normalization $(X^T X/n)_{i,i} = 1, \forall i \in \{1, ..., p\}$ holds true. Let $B := \{z \in \mathbb{R}^p : \Omega(z) \leq 1\}$ be the unit $\Omega-$ball, and $B_2 := \sup_{b \in B} b^T b$ an $\Omega-$ball and ℓ_2-ball comparison. Then we have for all $d > \mathrm{E}V$ and $\Delta > 1$*

$$\mathrm{P}(\mathcal{T}) \geq 1 - 2e^{-\frac{(d - \mathrm{E}V)^2 \Delta^2}{2B_2/n}} - 2e^{-\frac{n}{4}(1-\Delta^2)^2}.$$

Proof Let us define $\Sigma^2 := \sup_{b\in B} \mathrm{E}\left(\frac{(\epsilon^T X)_W b}{n\sigma}\right)^2$ and calculate it

$$\begin{aligned}
\Sigma^2 &= \sup_{b\in B} \mathrm{Var}\left(\frac{(\epsilon^T X)_W b}{n\sigma}\right) \\
&= \sup_{b\in B} \mathrm{Var}\left(\sum_{w\in W}\sum_{i=1}^{n} \frac{\epsilon_i}{n} X_{wi} b_w\right) \frac{1}{\sigma^2} \\
&= \sup_{b\in B} \left(\sum_{w\in W} b_w^2 \sum_{i=1}^{n} X_{wi}^2 \,\mathrm{Var}\left(\frac{\epsilon_i}{n}\right)\right) \frac{1}{\sigma^2} \\
&= \sup_{b\in B} b_W^T \cdot b_W / n \sum_{i=1}^{n} X_{wi}^2 / n \\
&= \sup_{b\in B} b_W^T \cdot b_W / n \le B_2/n. \qquad (3.13)
\end{aligned}$$

These calculations hold true as well for W^c instead of W. Furthermore in the subsequent inequalities we can subsitute W with W^c and use Z_2, V_2 instead of Z_1, V_1 to get an analogous result. We have $\frac{(\epsilon^T X)_W b}{\sigma n} \sim \mathcal{N}(0, b_W^2/n)$. This is an almost surely continuous centred Gaussian process. Therefore we can apply Theorem 5.8 from Boucheron et al. (2013)

$$\mathrm{P}(V_1 - \mathrm{E}\, V_1 \ge c) \le e^{-\frac{c^2}{2B_2/n}}. \qquad (3.14)$$

Now to get to a probability inequality for Z_1 we use the following calculations

$$\begin{aligned}
\mathrm{P}\left(Z_1 - \mathrm{E}\, V_1 \ge d\right) &\le \mathrm{P}\left(\frac{V_1\sigma}{\|\epsilon\|_n} - \mathrm{E}\, V_1 \ge d \wedge \|\epsilon\|_n > \sigma\Delta\right) + \mathrm{P}(\|\epsilon\|_n \le \sigma\Delta) \\
&\le \mathrm{P}\left(V_1 - \mathrm{E}\, V_1 \Delta > d\Delta\right) + \mathrm{P}(\|\epsilon\|_n \le \sigma\Delta) \\
&\le \mathrm{P}\left(V_1 - \mathrm{E}\, V_1 > d\Delta\right) + \mathrm{P}(\|\epsilon\|_n \le \sigma\Delta) \\
&\le e^{-\frac{d^2\Delta^2}{2B_2/n}} + \mathrm{P}(\|\epsilon\|_n \le \sigma\Delta). \qquad (3.15)
\end{aligned}$$

The calculations above use the union bound and that a bigger set containing another set has a bigger probability. Furthermore we have applied equations (3.13) and (3.14). Now we are left to give a bound on $\mathrm{P}(\|\epsilon\|_n/\sigma \le \Delta)$. For this we use the corollary to Lemma 1 from Laurent and Massart (2000) together with the fact that $\|\epsilon\|_n/\sigma = \sqrt{R/n}$ with $R = \sum_{i=1}^n (\epsilon_i/\sigma)^2 \sim \chi^2(n)$. We obtain

$$\begin{aligned}
\mathrm{P}\left(R \le n - 2\sqrt{nx}\right) &\le \exp(-x) \\
\mathrm{P}\left(\sqrt{\frac{R}{n}} \le \sqrt{1 - 2\sqrt{\frac{x}{n}}}\right) &\le \exp(-x) \\
\mathrm{P}\left(\frac{\|\epsilon\|_n}{\sigma} \le \Delta\right) &\le e^{-\frac{n}{4}(1-\Delta^2)^2}. \qquad (3.16)
\end{aligned}$$

Combining equations (3.15) and (3.16) finishes the proof:

$$\begin{aligned}
\mathrm{P}(\mathcal{T}) &= \mathrm{P}(\max(Z_1, Z_2) \le d) \\
&= \mathrm{P}(Z_1 \le d \cap Z_2 \le d) \\
&\ge \mathrm{P}(Z_1 \le d) + \mathrm{P}(Z_2 \le d) - 1 \\
&\ge 1 - \mathrm{P}(Z_1 \ge d) - \mathrm{P}(Z_2 \ge d) \\
&\ge 1 - 2e^{-\frac{(d-\mathrm{E}\, V)^2\Delta^2}{2B_2/n}} - 2e^{-\frac{n}{4}(1-\Delta^2)^2}.
\end{aligned}$$

■

So the probability that the event $\mathcal{T}$ does not occur decays exponentially. This is what we mean by having a very high probability. Therefore we can take $d = t \cdot \sqrt{\frac{\frac{2}{n}B_2}{\Delta^2}} + \mathrm{E}\,[V]$ with $\Delta^2 = 1 - t\frac{2}{\sqrt{n}}$, where $t = \sqrt{\log\left(\frac{4}{\alpha}\right)}$ and $2e^{-n/2} < \alpha$ to ensure $\Delta^2 > 0$. With this we get

$$\mathrm{P}(\mathcal{T}) \ge 1 - \alpha. \qquad (3.17)$$

First remark that the term $\frac{\epsilon^T}{\sigma}$ is now of the right scaling, because $\epsilon_i/\sigma \sim \mathcal{N}(0,1)$. This is the whole point of the square root regularization.
Here B_2 can be thought of comparing the $\Omega-$ball in direction W to the ℓ_2-ball in direction W, because if the norm Ω is the ℓ_2-norm, then $B_2 = 1$. Moreover, for every norm there exists a constant D such that for all β it holds

$$\|\beta\|_2 \le D\Omega(\beta).$$

Therefore the B_2 of Ω satisfies

$$B_2 \le D^2 \sup_{b\in B} \Omega(b_W)^2 \le D^2.$$

Thus we can take

$$\boxed{d = t \cdot \frac{D}{\Delta}\sqrt{\frac{2}{n}} + \mathrm{E}\,[V]}$$

$$\boxed{\Delta^2 = 1 - t\sqrt{\frac{2}{n}},\ \text{with } t = \sqrt{\log\left(\frac{4}{\alpha}\right)}}.$$

What is left to be determined is $\mathrm{E}\,[V]$. In many cases we can use a adjusted version of the main theorem in Maurer and Pontil (2012) for Gaussian complexities to obtain this expectation. All the examples below can be calculated in this way. So, in the case of Gaussian errors, we have the following new version of Corollary 10.

Corollary 12 *Take $\lambda = t/\Delta \cdot D\sqrt{\frac{2}{n}} + \mathrm{E}\,[V]$, where t, δ, V and D are defined as above. Invoke the same assumptions as in Theorem 9 and additionally assume Gaussian errors. Use the*

notation from Corollary 10. Then with probability $1-\alpha$ *the following oracle inequalities hold true*

$$\|X(\hat{\beta}-\beta^0)\|_n^2 \leq \|X(\beta_\star-\beta^0)\|_n^2 + C_1\lambda^2\cdot\Gamma_\Omega^2(L_{S_\star},S_\star)$$
$$\Omega(\hat{\beta}_{S_\star}-\beta_\star)+\Omega^{S_\star^c}(\hat{\beta}_{S_\star^c}) \leq C_2\left(\frac{\|X(\beta_\star-\beta^0)\|_n^2}{\lambda}+C_1\lambda\cdot\Gamma_\Omega^2(L_{S_\star},S_\star)\right).$$

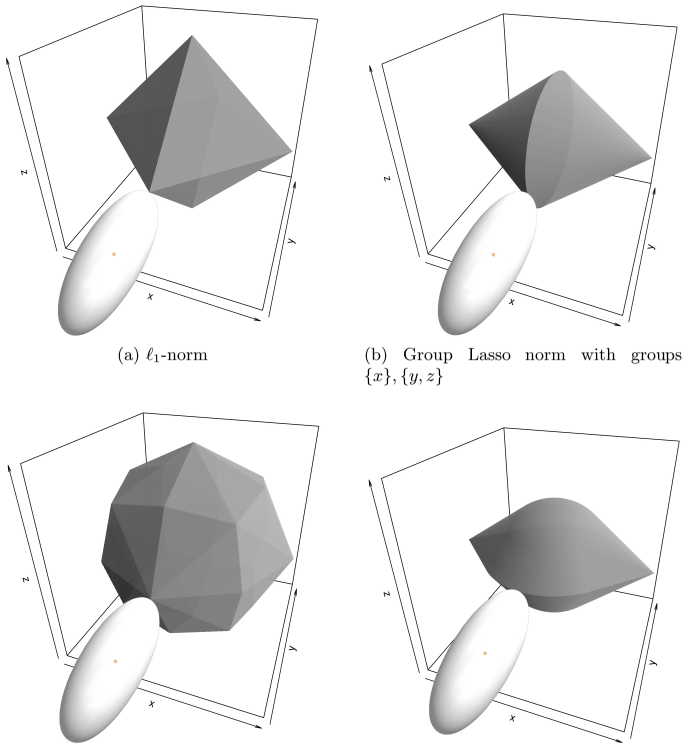
Now we still have a $\|\epsilon\|_n^2$ term in the constants (3.11), (3.12) of the oracle inequality. In order to handle this we need Lemma 1 from Laurent and Massart (2000). Which translates in our case to the probability inequality

$$\mathrm{P}\left(\|\epsilon\|_n^2 \leq \sigma^2\left(1+2x+2x^2\right)\right) \geq 1-\exp\left(-n\cdot x^2\right).$$

Here $x>0$ is a constant. Therefore we have that $\|\epsilon\|_n^2$ is of the order of σ^2 with exponentially decaying probability in n. We could also write this in the following form

$$\mathrm{P}\left(\|\epsilon\|_n^2 \leq \sigma^2\cdot C\right) \geq 1-\exp\left(-\frac{n}{2}\left(C-\sqrt{2C-1}\right)\right).$$

Here we can choose any constant $C>1$ big enough and take the bound $\sigma^2\cdot C$ for $\|\epsilon\|_n^2$ in the oracle inequality. A similar bounds can be found in Laurent and Massart (2000) for $1/\|\epsilon\|_n^2$. This takes care of the random part in the sharp oracle bound with the Gaussian errors.



(a) ℓ_1-norm

(b) Group Lasso norm with groups $\{x\},\{y,z\}$

(c) sorted ℓ_1-norm with a λ sequence $1>0.5>0.3$

(d) wedge norm

Figure 3: Pictorial description of how the estimator $\hat{\beta}$ works, with unit balls of different sparsity inducing norms.

4. Examples

Here we will give some examples of estimators where our sharp oracle inequalities hold. Figure 3 shows the unit balls of some sparsity inducing norms that we will use as examples. In order to give the theoretical λ for these examples we will again assume Gaussian errors. Theorem 9 still holds for all the examples even for non Gaussian errors. Some of the examples will introduce new estimators inspired by methods similar to the square root LASSO.

4.1 Square Root LASSO

First we examine the square root LASSO,

$$\hat{\beta}_{srL} := \arg\min_{\beta \in \mathbb{R}^p} \left\{ \|Y - X\beta\|_n + \lambda \|\beta\|_1 \right\}.$$

Here we use the ℓ_1-norm as a sparsity measure. We know that the ℓ_1-norm has the nice property to be able to set certain unimportant parameters individually to zero. As already mentioned the ℓ_1-norm has the following decomposability property for any set S

$$\|\beta\|_1 = \|\beta_S\|_1 + \|\beta_{S^c}\|_1, \forall \beta \in \mathbb{R}^p.$$

Therefore we also have weak decomposability for all subsets $S \subset \{1, ..., p\}$ with Ω^{S^c} being the ℓ_1-norm again. Thus Assumption II is fulfilled for all sets S and so we are able to apply Theorem 9.
Furthermore for the square root LASSO we have that $D = 1$. This is because the ℓ_2-norm is bounded by the ℓ_1-norm without any constant. So in order to get the value of λ we need to calculate the expectation of the dual norm of $\frac{\epsilon^T X}{\sigma n}$. The dual norm of ℓ_1 is the $\ell_\infty-$norm. By Maurer and Pontil (2012), we also have

$$\max\left(\mathrm{E}\left[\frac{\left\| (\epsilon^T X)_{S_\star^c} \right\|_\infty}{n\sigma} \right], \mathrm{E}\left[\frac{\left\| (\epsilon^T X)_{S_\star} \right\|_\infty}{n\sigma} \right] \right) \leq \sqrt{\frac{2}{n}} \left(2 + \sqrt{\log(|p|)} \right).$$

Therefore the theoretical λ for the square root LASSO can be chosen as

$$\boxed{\lambda = \sqrt{\frac{2}{n}} \left(t/\Delta + 2 + \sqrt{\log(|p|)} \right).}$$

Even though this theoretical λ is very close to being optimal, it is not optimal, see for example van de Geer (2016). In the special case of the ℓ_1-norm penalization, we can simplify Corollary 12:

Corollary 13 (Square Root LASSO) *Take* $\lambda = \sqrt{\frac{2}{n}} \left(t/\Delta + 2 + \sqrt{\log(|p|)} \right)$, *where* $t > 0$ *and* $\Delta > 1$ *are chosen as in* (3.17). *Invoke the same assumptions as in Corollary 12. Then for* $\Omega(\cdot) = \|\cdot\|_1$, *we have with probability* $1 - \alpha$ *that the following oracle inequalities hold true:*

$$\|X(\hat{\beta}_{srL} - \beta^0)\|_n^2 \leq \|X(\beta_\star - \beta^0)\|_n^2 + C_1 \lambda^2 \cdot \Gamma_\Omega^2(L_{S_\star}, S_\star)$$
$$\|\hat{\beta}_{srL} - \beta_\star\|_1 \leq C_2 \left(\frac{\|X(\beta_\star - \beta^0)\|_n^2}{\lambda} + C_1 \lambda \cdot \Gamma_\Omega^2(L_{S_\star}, S_\star) \right).$$

Remark that in Corollary 13 we have an oracle inequality for the estimation error $\|\hat{\beta}_{srL} - \beta_\star\|_1$ in ℓ_1. This is due to the decomposability of the ℓ_1-norm. In other examples we will have the sum of two norms.

4.2 Group Square Root LASSO

In order to set groups of variables simultaneously to zero, and not only individual variables, we will look at a different sparsity inducing norm. Namely a ℓ_1-type norm for grouped variables, called the group LASSO norm. The group square root LASSO was introduced by Bunea et al. (2014) as

$$\hat{\beta}_{gsrL} := \arg\min_{\beta \in \mathbb{R}^p} \left\{ \|Y - X\beta\|_n + \lambda \sum_{j=1}^{g} \sqrt{|G_j|} \|\beta_{G_j}\|_2 \right\}.$$

Here g is the total number of groups, and G_j is the set of variables that are in the jth group. Of course the ℓ_1-norm is a special case of the group LASSO norm, when $G_j = \{j\}$ and $g = p$.
The group LASSO penalty is also weakly decomposable with $\Omega^{S^c} = \Omega$, for any $S = \bigcup_{j \in \mathcal{J}} G_j$, with any $\mathcal{J} \subset \{1, ..., g\}$. So here the sparsity structure of the group LASSO norm induces the sets S to be of the same sparsity structure in order to fulfil Assumption II. Therefore the Theorem 9 can also be applied in this case.
How do we need to choose the theoretical λ? For the group LASSO norm we have $B_2 \leq 1$. One can see this due to the fact that $\sqrt{a_1} + ... + \sqrt{a_g} \geq \sqrt{a_1 + ... + a_g}$ for g positive constants. And also $|G_j| \geq 1$ for all groups. Therefore

$$\sum_{j=1}^{g} \sqrt{|G_j|} \|\beta_{G_j}\|_2 \geq \sqrt{\sum_{i=1}^{p} \beta_i^2}.$$

Remark that the dual norm is $\Omega^*(\beta) = \max_{1 \leq j \leq g} \|\beta_{G_j}\|_2 / \sqrt{|G_j|}$. With Maurer and Pontil (2012) we have

$$\max\left(\mathrm{E}\left[\frac{\Omega^*\left((\epsilon^T X)_{S_\star^c} \right)}{n\sigma} \right], \mathrm{E}\left[\frac{\Omega^*\left((\epsilon^T X)_{S_\star} \right)}{n\sigma} \right] \right) \leq \sqrt{\frac{2}{n}} \left(2 + \sqrt{\log(g)} \right).$$

That is why λ can be taken of the following form

$$\boxed{\lambda = \sqrt{\frac{2}{n}} \left(t/\Delta + 2 + \sqrt{\log(g)} \right).}$$

And we get a similar corollary for the group square root LASSO like the Corollary 13 for the square root LASSO. In the case of the group LASSO, there are better results for the theoretical penalty level available, see for example Theorem 8.1 in Bühlmann and van de Geer (2011). This takes the minimal group size into account.

4.3 Square Root SLOPE

Here we introduce a new method called the square root SLOPE estimator, which is also part of the square root regularization family. Let us thus take a look at the sorted ℓ_1 norm with some decreasing sequence $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_p > 0$,

$$J_\lambda(\beta) := \lambda_1 |\beta|_{(1)} + ... + \lambda_p |\beta|_{(p)}.$$

This was shown to be a norm by Zeng and Figueiredo (2014).
Let π be a permutation of $\{1,\dots,p\}$. The identity permutation is denoted by id. In order to show weak decomposability for the norm J_λ we need the following lemmas.

Lemma 14 (Rearrangement Inequality) *Let $\beta_1 \geq \cdots \geq \beta_p$ be a decreasing sequence of non-negative numbers. The sum $\sum_{i=1}^p \lambda_i \beta_{\pi(i)}$ is maximized over all permutations π at $\pi = id$.*

Proof. The result is obvious when $p = 2$. Suppose now that it is true for sequences of length $p-1$. We then prove it for sequences of length p as follows. Let π be an arbitrary permutation with $j := \pi(p)$. Then

$$\sum_{i=1}^{p} \lambda_i \beta_{\pi(i)} = \sum_{i=1}^{p-1} \lambda_i \beta_{\pi(i)} + \lambda_p \beta_j.$$

By induction

$$\begin{aligned}\sum_{i=1}^{p-1} \lambda_i \beta_{\pi(i)} &\leq \sum_{i=1}^{j-1} \lambda_i \beta_i + \sum_{i=j+1}^{p} \lambda_{i-1} \beta_i \\ &= \sum_{i \neq j} \lambda_i \beta_i + \sum_{i=j+1}^{p} (\lambda_{i-1} - \lambda_i) \beta_i \\ &= \sum_{i=1}^{p} \lambda_i \beta_i + \sum_{i=j+1}^{p} (\lambda_{i-1} - \lambda_i) \beta_i - \lambda_j \beta_j.\end{aligned}$$

Hence we have

$$\begin{aligned}\sum_{i=1}^{p} \lambda_i \beta_{\pi(i)} &\leq \sum_{i=1}^{p} \lambda_i \beta_i + \sum_{i=j+1}^{p} (\lambda_{i-1} - \lambda_i) \beta_i + (\lambda_j - \lambda_p) \beta_j \\ &= \sum_{i=1}^{p} \lambda_i \beta_i + \sum_{i=j+1}^{p} (\lambda_{i-1} - \lambda_i) \beta_i - \sum_{i=j+1}^{p} (\lambda_{i-1} - \lambda_i) \beta_j \\ &= \sum_{i=1}^{p} \lambda_i \beta_i + \sum_{i=j+1}^{p} (\lambda_{i-1} - \lambda_i)(\beta_i - \beta_j).\end{aligned}$$

Since $\lambda_{i-1} \geq \lambda_i$ for all $1 \leq i \leq p$ (defining $\lambda_0 = 0$) and $\beta_i \leq \beta_j$ for all $i > j$ we know that

$$\sum_{i=j+1}^{p} (\lambda_{i-1} - \lambda_i)(\beta_i - \beta_j) \leq 0.$$

□

Lemma 15 *Let*

$$\Omega(\beta) = \sum_{i=1}^{p} \lambda_i |\beta|_{(i)},$$

and

$$\Omega^{S^c}(\beta_{S^c}) = \sum_{l=1}^{r} \lambda_{p-r+l} |\beta|_{(l,S^c)},$$

where $r = p - s$ and $|\beta|_{(1,S^c)} \geq \cdots \geq |\beta|_{(r,S^c)}$ is the ordered sequence in β_{S^c}. Then $\Omega(\beta) \geq \Omega(\beta_S) + \Omega^{S^c}(\beta_{S^c})$. Moreover Ω^{S^c} is the strongest norm among all $\underline{\Omega}^{S^c}$ for which $\Omega(\beta) \geq \Omega(\beta_S) + \underline{\Omega}^{S^c}(\beta_{S^c})$

Proof. Without loss of generality assume $\beta_1 \geq \cdots \geq \beta_p \geq 0$. We have

$$\Omega(\beta_S) + \Omega^{S^c}(\beta_{S^c}) = \sum_{i=1}^{p} \lambda_i \beta_{\pi(i)}$$

for a suitable permutation π. It follows that

$$\Omega(\beta_S) + \Omega^{S^c}(\beta_{S^c}) \leq \Omega(\beta).$$

To show Ω^{S^c} is the strongest norm it is clear we need only to search among candidates of the form

$$\underline{\Omega}^{S^c}(\beta_{S^c}) = \sum_{l=1}^{r} \underline{\lambda}_{p-r+l} \beta_{\pi^{S^c}(l)}$$

where $\{\underline{\lambda}_{p-r+l}\}$ is a decreasing positive sequence and where $\pi^{S^c}(1),\dots,\pi^{S^c}(r)$ is a permutation of indices in S^c.
This is then maximized by ordering the indices in S^c in decreasing order. But then it follows that the largest norm is obtained by taking $\underline{\lambda}_{p-r+l} = \lambda_{p-r+l}$ for all $l = 1,\dots,r$. □
The SLOPE was introduced by Bogdan et al. (2015) in order to better control the false discovery rate, and is defined as:

$$\hat{\beta}_{SLOPE} := \arg\min_{\beta \in \mathbb{R}^p} \left\{ \|Y - X\beta\|_n^2 + \lambda J_\lambda(\beta) \right\}.$$

Now we are able to look at the square root SLOPE, which is the estimator of the form:

$$\hat{\beta}_{srSLOPE} := \arg\min_{\beta \in \mathbb{R}^p} \left\{ \|Y - X\beta\|_n + \lambda J_\lambda(\beta) \right\}.$$

The square root SLOPE replaces the squared ℓ_2-norm with a ℓ_2-norm. With Theorem 9 we have provided a sharp oracle inequality for this new estimator, the square root SLOPE. For the SLOPE penalty we have $B_2 \leq \frac{1}{\lambda_p}$, if $\lambda_p > 0$. This is because

$$\begin{aligned}\frac{J_\lambda(\beta)}{\lambda_p} &= \frac{\lambda_1}{\lambda_p} |\beta|_{(1)} + \ldots + \frac{\lambda_p}{\lambda_p} |\beta|_{(p)} \\ &\geq \sum_{i=1}^{p} |\beta_i| = \|\beta\|_1 \\ &\geq \|\beta\|_2.\end{aligned}$$

So the bound gets scaled by the smallest λ. The dual norm of the SLOPE is by Lemma 1 of Zeng and Figueiredo (2015)

$$J_{\lambda}^{*}(\beta) = \max_{k=1,...,p} \left\{ \left(\sum_{j=1}^{k} \lambda_j \right)^{-1} \cdot \|\beta^{(k)}\|_1 \right\},$$

Here $\beta^{(k)} := (\beta_{(1)}, ..., \beta_{(k)})^T$ is the vector which contains the k largest elements of β. Again by Maurer and Pontil (2012) we have

$$\max \left(\mathrm{E} \left[\frac{J_{\lambda}^{*}\left((\epsilon^T X)_{S_\star}\right)}{n\sigma} \right], \left[\frac{J_{\lambda}^{S_\star^c *}\left((\epsilon^T X)_{S_\star^c}\right)}{n\sigma} \right] \right) \leq \sqrt{\frac{2}{n}} \left(\frac{2\sqrt{2}+1}{\sqrt{2}} + \sqrt{\log(|R^2|)} \right).$$

Here we denote by $R^2 := \sum_i \frac{1}{\lambda_i^2}$. Therefore we can choose λ as

$$\boxed{\lambda = \sqrt{\frac{2}{n}} \left(\frac{t}{\lambda_p \Delta} + \frac{2\sqrt{2}+1}{\sqrt{2}} + \sqrt{\log(|R^2|)} \right).}$$

Let us remark that the asymptotic minimaxity of SLOPE can be found in Su and Candès (2016).

4.4 Sparse Group Square Root LASSO

The sparse group square root LASSO can be defined similarly to the sparse group LASSO, see Simon et al. (2013). This new method is defined as:

$$\hat{\beta}_{srSGLASSO} := \arg\min_{\beta \in \mathbb{R}^p} \left\{ \|Y - X\beta\|_n + \lambda \|\beta\|_1 + \eta \sum_{t=1}^{T} \|\beta_{I_t}\|_2 \sqrt{|G_t|} \right\},$$

where we have a partition as follows, $G_t \subset \{1, ..., p\}\ \forall t \in 1, ..., T$, $\bigcup_{t=1}^{T} G_t = \{1, ..., p\}$ and $G_i \cap G_j = \varnothing\ \forall i \neq j$. This penalty is again a norm and it not only chooses sparse groups by the group LASSO penalty, but also sparsity inside of the groups with the ℓ_1-norm. Define $R(\beta) := \lambda \|\beta\|_1 + \eta \sum_{t=1}^{T} \|\beta_{I_t}\|_2 \sqrt{|G_t|}$ and $R^{S^c}(\beta) := \lambda \|\beta\|_1$. Then we have weak decomposability for any set S

$$R(\beta_S) + R^{S^c}(\beta_{S^c}) \leq R(\beta).$$

This is due to the weak decomposability property of the ℓ_1-norm and $\|\beta_S\|_2 = \sqrt{\sum_{j \in S} \beta_j^2} \leq \sqrt{\sum_{j \in S} \beta_j^2 + \sum_{j \in S^c} \beta_j^2} = \|\beta\|_2$. Now in order to get the theoretical λ let us note that if we sum two norms, it is again a norm. Then the dual of this added norm is, because of the supremum taken over the unit ball, smaller than dual norm of each one of the two norms individually. So we can invoke the same theoretical λ as with the square root LASSO

$$\boxed{\lambda = \sqrt{\frac{2}{n}} \left(t/\Delta + 2 + \sqrt{\log(|p|)} \right).}$$

And also the theoretical η like the group square root LASSO

$$\boxed{\eta = \sqrt{\frac{2}{n}} \left(t/\Delta + 2 + \sqrt{\log(g)} \right).}$$

But of course we will not get the same Corollary, because the $\Omega-$effective sparsity will be different.

4.5 Structured Sparsity

Here we will look at the very general concept of structured sparsity norms. Let $\mathcal{A} \subset [0, \infty)^p$ be a convex cone such that $\mathcal{A} \cap (0, \infty)^p \neq \varnothing$. Then

$$\Omega(\beta) = \Omega(\beta; \mathcal{A}) := \min_{a \in \mathcal{A}} \frac{1}{2} \sum_{j=1}^{p} \left(\frac{\beta_j^2}{a_j} + a_j \right),$$

is a norm by Micchelli et al. (2010). Some special cases are for example the ℓ_1-norm or the wedge or box norm. Define

$$\mathcal{A}_S := \{a_S : a \in \mathcal{A}\}.$$

Then van de Geer (2014) also showed that for any $\mathcal{A}_S \subset \mathcal{A}$ we have that the set S is allowed and we have weak decomposability for the norm $\Omega(\beta)$ with $\Omega^{S^c}(\beta_{S^c}) := \Omega(\beta_{S^c}, \mathcal{A}_{S^c})$. Hence the estimator

$$\hat{\beta}_s = \arg\min_{\beta \in \mathbb{R}^p} \left\{ \|Y - X\beta\|_n + \lambda \min_{a \in \mathcal{A}} \frac{1}{2} \sum_{j=1}^{p} \left(\frac{\beta_j^2}{a_j} + a_j \right) \right\},$$

has also the sharp oracle inequality. The dual norm is given by

$$\Omega^{*}(\omega; \mathcal{A}) = \max_{a \in \mathcal{A}(1)} \sqrt{\sum_{j=1}^{p} a_j \omega_j^2}, \quad \omega \in \mathbb{R}^p,$$

$$\Omega^{S^c *}(\omega; \mathcal{A}_{S^c}) = \max_{a \in \mathcal{A}_{S^c}(1)} \sqrt{\sum_{j=1}^{p} a_j \omega_j^2},\ \omega \in \mathbb{R}^p.$$

Here $\mathcal{A}_{S^c}(1) := \{a \in \mathcal{A}_{S^c} : \|a\|_1 = 1\}$ and $\mathcal{A}(1) := \{a \in \mathcal{A} : \|a\|_1 = 1\}$. Then once again by Maurer and Pontil (2012) we have

$$\max \left(\mathrm{E} \left[\frac{\Omega^{*}\left((\epsilon^T X)_{S_\star}; \mathcal{A}_{S_\star}\right)}{n\sigma} \right], \mathrm{E} \left[\frac{\Omega^{*}\left((\epsilon^T X)_{S_\star^c}; \mathcal{A}_{S_\star^c}\right)}{n\sigma} \right] \right) \leq \sqrt{\frac{2}{n}} \widetilde{\mathcal{A}}_{S_\star} \left(2 + \sqrt{\log(|\,\mathrm{E}(\mathcal{A})|)} \right).$$

Here $\mathrm{E}(\mathcal{A})$ are the extreme points of the closure of the set $\left\{ \frac{a}{\|a\|_1} : a \in \mathcal{A} \right\}$. With the definition $\widetilde{\mathcal{A}}_{S_\star} := \max \left(\sqrt{\sum_{i=1}^{n} \Omega(X_{i,S_\star}; \mathcal{A}_{S_\star})}, \sqrt{\sum_{i=1}^{n} \Omega(X_{i,S_\star^c}; \mathcal{A}_{S_\star^c})} \right)$. That is why λ can be taken of the following form

$$\boxed{\lambda = \sqrt{\frac{2}{n}} \left(tD/\Delta + \widetilde{\mathcal{A}}_{S_\star} \left(2 + \sqrt{\log(|\,\mathrm{E}(\mathcal{A})|)} \right) \right).}$$

Since we do not know $S_\star$ we can either upper bound $\widetilde{\mathcal{A}}_{S_\star}$ for a given norm, or use the fact that $\Omega(\beta) \geq \|\beta\|_1$ and $\Omega^*(\beta) \leq \|\beta\|_\infty$ for all $\beta \in \mathbb{R}^p$. Therefore use the same λ as for the square root LASSO. And we get similar corollaries for the structured sparsity norms like the Corollary 13 for the square root LASSO.

5. Simulation: Comparison between srLASSO and srSLOPE

The goal of this simulation is to see how the estimation and prediction errors for the square root LASSO and the square root SLOPE behave under some Gaussian designs. We propose Algorithm 1 to solve the square root SLOPE:

Algorithm 1: srSLOPE

input : β^0 a starting parameter vector,
λ a desired penalty level with a decreasing sequence,
Y the response vector,
X the design matrix.

output: $\hat{\beta}_{srSLOPE} = \underset{\beta\in\mathbb{R}^p}{\arg\min}\left(\|Y - X\beta\|_n + \lambda J_\lambda(\beta)\right)$

1 **for** $i \leftarrow 0$ **to** i_{stop} **do**
2 $\quad \sigma_{i+1} \leftarrow \|Y - X\beta_i\|_n$;
3 $\quad \beta_{i+1} \leftarrow \underset{\beta\in\mathbb{R}^p}{\arg\min}\left(\|Y - X\beta\|_n^2 + \sigma_{i+1}\lambda J_\lambda(\beta)\right)$;
4 **end**

Note that in Algorithm 1 Line 3 we need to solve the usual SLOPE. To solve the SLOPE we have used the algorithm provided in Bogdan et al. (2015). For the square root LASSO we have used the R-Package flare by Li et al. (2014).
We consider a high-dimensional linear regression model:

$$Y = X\beta^0 + \epsilon,$$

with $n = 100$ response variables and $p = 500$ unknown parameters. The design matrix X is chosen with the rows being fixed i.i.d. realizations from $\mathcal{N}(0, \Sigma)$. Here the covariance matrix Σ has a Toeplitz structure

$$\Sigma_{i,j} = 0.9^{|i-j|}.$$

We choose i.i.d. Gaussian errors ϵ with a variance of $\sigma^2 = 1$. For the underlying unknown parameter vector β^0 we choose different settings. For each such setting we calculate the square root LASSO and the square root SLOPE with the theoretical λ given in this paper and the λ from a 8-fold Cross-validation on the mean squared prediction error. We use $r = 100$ repetitions to calculate the ℓ_1-estimation error, the sorted ℓ_1-estimation error and the ℓ_2-prediction error. As for the definition of the sorted ℓ_1-norm, we chose a regular decreasing sequence from 1 to 0.1 with length 500. The results can be found in Table 1,2,3 and 4.

Decreasing Case:
Here the active set is chosen as $S_0 = \{1, 2, 3, ..., 7\}$, and $\beta^0_{S_0} = (4,\ 3.\overline{6},\ 3.\overline{3},\ 3,\ 2.\overline{6},\ 2.\overline{3},\ 2)^T$ is a decreasing sequence.

Table 1: Decreasing β

	theoretical λ			Cross-validated λ		
	$\|\beta^0 - \hat{\beta}\|_{\ell_1}$	$J_\lambda(\beta^0 - \hat{\beta})$	$\|X(\beta^0 - \hat{\beta})\|_{\ell_2}$	$\|\beta^0 - \hat{\beta}\|_{\ell_1}$	$J_\lambda(\beta^0 - \hat{\beta})$	$\|X(\beta^0 - \hat{\beta})\|_{\ell_2}$
srSLOPE	2.06	0.21	4.12	2.37	0.26	3.88
srLASSO	1.85	0.19	5.51	1.78	0.19	5.05

Decreasing Random Case:
The active set was randomly chosen to be $S_0 = \{154, 129, 276, 29, 233, 240, 402\}$ and again $\beta^0_{S_0} = (4,\ 3.\overline{6},\ 3.\overline{3},\ 3,\ 2.\overline{6},\ 2.\overline{3},\ 2)^T$.

Table 2: Decreasing Random β

	theoretical λ			Cross-validated λ		
	$\|\beta^0 - \hat{\beta}\|_{\ell_1}$	$J_\lambda(\beta^0 - \hat{\beta})$	$\|X(\beta^0 - \hat{\beta})\|_{\ell_2}$	$\|\beta^0 - \hat{\beta}\|_{\ell_1}$	$J_\lambda(\beta^0 - \hat{\beta})$	$\|X(\beta^0 - \hat{\beta})\|_{\ell_2}$
srSLOPE	4.50	0.49	7.74	7.87	1.09	7.68
srLASSO	8.48	0.89	29.47	7.81	0.85	9.19

Grouped Case:
Now in order to see if the square root SLOPE can catch grouped variables better than the square root LASSO we look at an active set $S_0 = \{1, 2, 3, ..., 7\}$ together with $\beta^0_{S_0} = (4, 4, 4, 3, 3, 2, 2)^T$.

Table 3: Grouped β

	theoretical λ			Cross-validated λ		
	$\|\beta^0 - \hat{\beta}\|_{\ell_1}$	$J_\lambda(\beta^0 - \hat{\beta})$	$\|X(\beta^0 - \hat{\beta})\|_{\ell_2}$	$\|\beta^0 - \hat{\beta}\|_{\ell_1}$	$J_\lambda(\beta^0 - \hat{\beta})$	$\|X(\beta^0 - \hat{\beta})\|_{\ell_2}$
srSLOPE	2.81	0.29	6.43	1.71	0.18	3.65
srLASSO	3.02	0.31	8.37	1.83	0.19	4.25

Grouped Random Case:
Again we take the same randomly chosen set $S_0 = \{154, 129, 276, 29, 233, 240, 402\}$ with $\beta^0_{S_0} = (4, 4, 4, 3, 3, 2, 2)^T$.

Table 4: Grouped Random β

	theoretical λ			Cross-validated λ		
	$\|\beta^0 - \hat{\beta}\|_{\ell_1}$	$J_\lambda(\beta^0 - \hat{\beta})$	$\|X(\beta^0 - \hat{\beta})\|_{\ell_2}$	$\|\beta^0 - \hat{\beta}\|_{\ell_1}$	$J_\lambda(\beta^0 - \hat{\beta})$	$\|X(\beta^0 - \hat{\beta})\|_{\ell_2}$
srSLOPE	6.05	0.66	12.84	5.80	0.66	5.78
srLASSO	16.90	1.77	66.68	6.14	0.67	6.67

The random cases usually lead to larger errors for both estimators. This is due to the correlation structure of the design matrix. The square root SLOPE seems to outperform

the square root LASSO in the cases where β^0 is somewhat grouped (grouped in the sense that amplitudes of same magnitude appear). This is due to the structure of the sorted ℓ_1-norm, which has some of the sparsity properties of ℓ_1 as well as some of the grouping properties of ℓ_∞, see Zeng and Figueiredo (2014). Therefore the square root SLOPE reflects the underlying sparsity structure in the grouped cases. What is also remarkable is that the square root SLOPE always has a better mean squared prediction error than the square root LASSO. This is even in cases, where square root LASSO has better estimation errors. The estimation errors seem to be better for the square root LASSO in the decreasing cases.

6. Discussion

Sparsity inducing norms different from ℓ_1 may be used to facilitate the interpretation of the results. Depending on the sparsity structure we have provided sharp oracle inequalities for square root regularization. Due to the square root regularizing we do not need to estimate the variance, the estimators are all pivotal. Moreover, because the penalty is a norm the optimization problems are all convex, which is a practical advantage when implementing the estimation procedures. For these sharp oracle inequalities we only needed the weak decomposability and not the decomposability property of the ℓ_1-norm. The weak decomposability generalizes the desired property of promoting an estimated parameter vector with a sparse structure. The structure of the $\Omega-$ and $\Omega^{S^c}-$norms influence the oracle bound. Therefore it is useful to use norms that reflect the true underlying sparsity structure.

Acknowledgments

We would like to acknowledge financial support from the SNSF (Swiss National Science Foundation) with Grant number 200021 149145. We thank the editor and the referees for their very helpful suggestions and improvements.

References

F.R. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends® in Machine Learning*, 4(1):1–106, 2012.

H. H. Bauschke and P. L. Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC. Springer, New York, 2011. With a foreword by Hédy Attouch.

A. Belloni, V. Chernozhukov, and L. Wang. Square-root lasso: pivotal recovery of sparse signals via conic programming. *Biometrika*, 98(4):791–806, 2011.

A. Belloni, V. Chernozhukov, and L. Wang. Pivotal estimation via square-root Lasso in nonparametric regression. *The Annals of Statistics*, 42(2):757–788, 2014.

P. J. Bickel, Y. Ritov, and A. B. Tsybakov. Simultaneous analysis of lasso and Dantzig selector. *The Annals of Statistics*, 37(4):1705–1732, 2009.

M. Bogdan, E. van den Berg, C. Sabatti, W. Su, and E.J. Candès. SLOPE—adaptive variable selection via convex optimization. *The Annals of Applied Statistics*, 9(3):1103–1140, 2015. URL `http://statweb.stanford.edu/~candes/SortedL1/`.

S. Boucheron, M. Ledoux, G. Lugosi, and P. Massart. *Concentration inequalities : a nonasymptotic theory of independence*. Oxford university press, 2013. ISBN 978-0-19-953525-5.

F. Bunea, A. B. Tsybakov, and M. H. Wegkamp. Aggregation and sparsity via l_1 penalized least squares. In *Learning theory*, volume 4005 of *Lecture Notes in Comput. Sci.*, pages 379–391. Springer, Berlin, 2006.

F. Bunea, A. B. Tsybakov, and M. H. Wegkamp. Aggregation for Gaussian regression. *The Annals of Statistics*, 35(4):1674–1697, 2007.

F. Bunea, J. Lederer, and Y. She. The group square-root lasso: Theoretical properties and fast algorithms. *IEEE Transactions on Information Theory*, 60(2):1313–1325, 2014.

P. Bühlmann and S. van de Geer. *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer Publishing Company, Incorporated, 1st edition, 2011. ISBN 3642201911, 9783642201912.

O. Klopp. Noisy low-rank matrix completion with general sampling distribution. *Bernoulli*, 20(1):282–303, 2014.

V. Koltchinskii. *Oracle inequalities in empirical risk minimization and sparse recovery problems*, volume 2033 of *Lecture Notes in Mathematics*. Springer, Heidelberg, 2011. Lectures from the 38th Probability Summer School held in Saint-Flour, 2008, École d'Été de Probabilités de Saint-Flour. [Saint-Flour Probability Summer School].

V. Koltchinskii, K. Lounici, and A. B. Tsybakov. Nuclear-norm penalization and optimal rates for noisy low-rank matrix completion. *Ann. Statist.*, 39(5):2302–2329, 2011.

B. Laurent and P. Massart. Adaptive estimation of a quadratic functional by model selection. *The Annals of Statistics*, 28(5):pp. 1302–1338, 2000.

X. Li, T. Zhao, L. Wang, X. Yuan, and H. Liu. *flare: Family of Lasso Regression*, 2014. URL `https://CRAN.R-project.org/package=flare`. R package version 1.5.0.

A. Maurer and M. Pontil. Structured sparsity and generalization. *Journal of Machine Learning Research*, 13(1):671–690, 2012.

C.A. Micchelli, J. Morales, and M. Pontil. A family of penalty functions for structured sparsity. In J.D. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1612–1623. Curran Associates, Inc., 2010.

C.A. Micchelli, J. Morales, and M. Pontil. Regularizers for structured sparsity. *Advances in Computational Mathematics*, 38(3):455–489, 2013.

A. B. Owen. A robust hybrid of lasso and ridge regression. In *Prediction and discovery*, volume 443 of *Contemp. Math.*, pages 59–71. Amer. Math. Soc., Providence, RI, 2007.

N. Simon, J. Friedman, T. Hastie, and R. Tibshirani. A sparse-group lasso. *Journal of Computational and Graphical Statistics*, 2013.

W. Su and E. Candès. SLOPE is adaptive to unknown sparsity and asymptotically minimax. *Ann. Statist.*, 44(3):1038–1068, 2016.

T. Sun and C. Zhang. Scaled sparse linear regression. *Biometrika*, 99(4):879–898, 2012.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B. Methodological*, 58(1):267–288, 1996.

S. van de Geer. The deterministic lasso. *JSM proceedings*, 2007.

S. van de Geer. Weakly decomposable regularization penalties and structured sparsity. *Scand. J. Stat.*, 41(1):72–86, 2014.

S. van de Geer. *Estimation and Testing under Sparsity*. École d'Éte de Saint-Flour XLV. Springer (to appear), 2016.

S. van de Geer and J. Lederer. The Lasso, correlated design, and improved oracle inequalities. 9:303–316, 2013.

X. Zeng and M. A. T. Figueiredo. Decreasing weighted sorted l1 regularization. *IEEE Signal Processing Letters*, 21(10):1240–1244, June 2014.

X. Zeng and M. A. T. Figueiredo. The ordered weighted l1 norm: Atomic formulation and conditional gradient algorithm. In *Workshop on Signal Processing with Adaptive Sparse Structured Representations - SPARS*, July 2015.

Journal of Machine Learning Research 18 (2017) 1-43 Submitted 11/15; Revised 6/17; Published 8/17

Soft Margin Support Vector Classification as Buffered Probability Minimization

Matthew Norton mdnorto@gmail.com
Alexander Mafusalov sasha.mafusalov@gmail.com
Stan Uryasev uryasev@ufl.edu
Risk Management and Financial Engineering Lab
Department of Industrial and Systems Engineering
University of Florida
Gainesville, FL 32611, USA.

Editor: Corinna Cortes

Abstract

In this paper, we show that the popular C-SVM, soft-margin support vector classifier is equivalent to minimization of Buffered Probability of Exceedance (bPOE), a recently introduced characterization of uncertainty. To show this, we introduce a new SVM formulation, called the EC-SVM, which is derived from a simple bPOE minimization problem that is easy to interpret with a meaningful free parameter, optimal objective value, and probabilistic derivation. Over the range of its free parameter, the EC-SVM has both a convex and non-convex case which we connect to existing SVM formulations. We first show that the C-SVM, formulated with any regularization norm, is equivalent to the convex EC-SVM. Similarly, we show that the Eν-SVM is equivalent to the EC-SVM over its entire parameter range, which includes both the convex and non-convex case. These equivalences, coupled with the interpretability of the EC-SVM, allow us to gain surprising new insights into the C-SVM and fully connect soft margin support vector classification with superquantile and bPOE concepts. We also show that the EC-SVM can easily be cast as a robust optimization problem, where bPOE is minimized with data lying in a fixed uncertainty set. This reformulation allows us to clearly differentiate between the convex and non-convex case, with convexity associated with pessimistic views of uncertainty and non-convexity associated with optimistic views of uncertainty. Finally, we address some practical considerations. First, we show that these new insights can assist in making parameter selection more efficient. Second, we discuss optimization approaches for solving the EC-SVM. Third, we address the issue of generalization, providing generalization bounds for both bPOE and misclassification rate.

Keywords: Support Vector Machines, Buffered Probability of Exceedance, Conditional Value-at-Risk, Binary Classification, Robust Optimization

1. Introduction

In the machine learning community, the Soft Margin Support Vector Machine (C-SVM) has proven to be an extremely popular tool for classification, spawning generalizations for regression, robust optimization, and a host of other applications. With its connection to statistical learning theory, intuitive geometric interpretation, and efficient extensions to non-linear classification, the C-SVM has proven to be a flexible tool based on sound theory and intuition. Still, there are insights left to be gained with regard to soft margin support vector classification.

One such insight was revealed by Takeda and Sugiyama (2008), where it was shown that the Eν-SVM, an extension of the ν-SVM, is equivalent to superquantile minimization. Superquantiles, popularized in the financial engineering literature under the name Conditional Value-at-Risk (CVaR), were developed by Rockafellar and Uryasev (2002) as means for dealing with optimization of quantiles. Utilizing the popular calculation formula for superquantiles, Takeda and Sugiyama (2008) showed that the Eν-SVM was equivalent to superquantile minimization, with the free parameter of the Eν-SVM being equivalent to the free choice of probability level in superquantile minimization.

In this paper, we provide insights in a similar direction by utilizing the inverse of the superquantile, so called Buffered Probability of Exceedance (bPOE). More specifically, bPOE is a generalization of *buffered Probability of Failure* (bPOF) which was introduced by Rockafellar (2009) and further studied in Rockafellar and Royset (2010). This generalization, recently studied by Mafusalov and Uryasev (2015); Norton and Uryasev (2014); Norton et al. (2015); Davis and Uryasev (2014); Uryasev (2014), has shown a great deal of promise as generating numerically tractable methods for probability minimization.

Utilizing the bPOE concept, we introduce a new SVM formulation called the Extended Soft Margin Support Vector Machine (EC-SVM). Being derived as a bPOE minimization problem, the EC-SVM is simple to interpret. First, we show that the EC-SVM has a free parameter interpretable as a specific statistical quantity related to the optimal loss distribution. Second, we show that the value of the optimal objective function (divided by sample size) is equal to a probability level. Lastly, we show that the EC-SVM can be interpreted as having a hard-margin criterion. Additionally, with the EC-SVM formulated with any general norm, we show that the choice of norm implies a distance metric which defines the hard-margin criterion.

After introducing the EC-SVM, we then connect it to existing SVM formulations. In our main result, we show that the C-SVM and EC-SVM, when formulated with any general norm and non-negative parameter values, produce the same set of optimal hyperplanes. This result implies that the original soft-margin SVM formulation, derived in great part from geometric intuition, is equivalent to minimization of bPOE, a probabilistic concept. This result also implies that the interpretation of the EC-SVM's parameter, optimal objective, and hard-margin criterion can be applied to the C-SVM. This includes the surprising result that the optimal objective value of the C-SVM, divided by sample size, equals a probability level.

We also connect the EC-SVM and Eν-SVM, showing that these SVM formulations produce the same set of optimal hyperplanes over their entire parameter range. With bPOE being the inverse of the superquantile, this relationship follows immediately from the derivation of the EC-SVM as a bPOE minimization problem and the results of Takeda and Sugiyama (2008). This result also makes it clear that the EC-SVM is an extension of the C-SVM in the same way that the Eν-SVM is an extension of the ν-SVM.

Additionally, we provide a novel interpretation of the EC-SVM as a robust optimization (RO) problem, where bPOE is minimized with data points lying in a fixed uncertainty set. With the EC-SVM having both a convex and non-convex case, depending on the value of the free parameter, we find that the RO representation reveals a unique interpretation to

©2017 Norton, Mafusalov and Uryasev.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-566.html.

help distinguish these cases. Specifically, the RO representation reveals that the convex case corresponds to bPOE minimization with worst-case *pessimistic* data uncertainty, while the non-convex case corresponds to bPOE minimization with best-case *optimistic* data uncertainty. The RO representation also provides other insights. For the convex case, the RO representation is similar to existing equivalences for the C-SVM presented in Xu et al. (2009), but with a much simpler proof, definition of uncertainty set, and correspondence between equivalent parameter values. For the non-convex case, this new representation suggests an efficient alternating minimization method for finding a local minimum. Furthermore, the RO representation reveals that this seemingly heuristic optimization method is related to the DC Algorithm, a popular algorithm for finding the local minimum of DC (difference of convex functions) optimization problems.

Finally, we consider some practical implications of these new theoretical insights. We show that the new interpretation of the C tradeoff parameter suggests a range that should be used to select the C parameter. This helps to improve grid selection for cross validation so that one can partially avoid solving the EC-SVM for values of C that yield trivial or redundant optimal solutions. Furthermore, using the RO representation of the EC-SVM, we show that there may be situations where prior knowledge about data uncertainty suggests, at-best, a fixed C or, at-worst, much tighter bounds for the range of interesting C values for cross-validation.

We also address the practical issue of generalization. In classification, generalization bounds are typically provided for misclassification rate because classification algorithms are viewed as attempts to find classifiers which minimize this performance metric. In this regard, we utilize results from Takeda and Sugiyama (2008); Schölkopf et al. (2000) to show that generalization bounds for misclassification rate of EC-SVM classifiers can be posed in terms of empirical estimates of bPOE, the quantile, and the superquantile. However, with the new insight that the EC-SVM and C-SVM directly minimize bPOE, we also provide generalization bounds for this metric. We apply the stability arguments of Bousquet and Elisseeff (2002) to provide tight generalization bounds on the true bPOE of EC-SVM and C-SVM classifiers and show that bPOE threshold plays an important role in these bounds.

This paper is organized as follows. Note that in order to make this paper as self-contained as possible, we include a significant amount of review in the first three sections. Section 2 reviews some existing SVM formulations relevant to our discussion, specifically the C-SVM, ν-SVM, and Eν-SVM. Section 3 briefly reviews the concept of a superquantile and the results of Takeda and Sugiyama (2008), which show that the Eν-SVM is equivalent to superquantile minimization. Section 4 reviews the bPOE concept, which is critical to our contribution. Additionally, we present a new formulation for minimizing bPOE in the presence of Positive Homogenous (PH) random functions. The necessity of this new formulation is discussed in more detail in Appendix A. Section 5 introduces the EC-SVM as a bPOE minimization problem and discusses the properties of the EC-SVM and its interpretation as a hard-margin optimization problem. Section 6 connects the C-SVM and EC-SVM. Section 7 connects the EC-SVM and Eν-SVM and presents the results of this and previous papers in a cohesive framework connecting soft margin support vector classification and superquantile concepts. Section 8 presents dual formulations of the C-SVM and EC-SVM formulated with any general norm, discusses application of the kernel trick, and shows that the optimal objective of the C-SVM, divided by sample size, equals a probability level.

Section 9 shows that the EC-SVM has a novel interpretation as a RO problem, with a distinction between the convex and non-convex cases. Section 10.1 discusses the issue of parameter selection. Section 10.2 discusses optimization of the EC-SVM. Section 10.3 provides generalization bounds for misclassification rate and bPOE.

2. C-SVM, ν-SVM, Eν-SVM

In this section, we review three existing SVM formulations; C-SVM, ν-SVM, and Eν-SVM. We begin with a review of the C-SVM and ν-SVM, reviewing the fact that they share the same optimal solution sets. We then review the interpretation of the ν-SVM parameter, its limitations, and the Eν-SVM formulation which serves to resolve these limitations.

2.1 The C-SVM

Consider the task of binary classification where we have a set of N feature vectors $X_i \in \mathbb{R}^n$ and associated class labels $y_i \in \{-1, +1\}$ and we need to choose a hyperplane $w \in \mathbb{R}^n$ with intercept $b \in \mathbb{R}$ to properly classify feature vectors via the linear decision function $d(w, b, X_i) = sign(w^T X_i + b)$.

One of the most successful algorithms for accomplishing this task is the soft-margin SVM from Cortes and Vapnik (1995), also referred to as the C-SVM. The C-SVM is formulated as (1), where $C \geq 0 \in \mathbb{R}$ is chosen as a fixed tradeoff parameter and the norm is typically the L_2, $\|\cdot\|_2$, or L_1, $\|\cdot\|_1$, norm.[1] Below, we present it with the general norm, $\|\cdot\|$.

$$\begin{aligned} \min_{w,b,\xi} \quad & C\|w\| + \sum_{i=1}^{N} \xi_i \\ s.t. \quad & \xi_i \geq -y_i(w^T X_i + b) + 1, \quad \forall i \in \{1, ..., N\}, \\ & \xi \geq 0. \end{aligned} \tag{1}$$

2.2 The ν-SVM

After introduction of the C-SVM, the ν-SVM was introduced by Schölkopf et al. (2000) as an equivalent formulation with more intuitive parameter choices. C-SVM and ν-SVM, with the L_2 norm, are equivalent in that they provide the same set of optimal solutions over the space of all possible parameter choices, see Chang and Lin (2001). These algorithms are different, though, in the meaning of the value of the free parameter. For the C-SVM, there was no direct interpretation for the meaning of the C-parameter other than as a trade-off between margin size and classification errors. The ν-SVM, on the other hand, provided a more interpretable parameter.

1. Often, the squared norm is utilized. We use non-squared for ease of presentation. Many of the ideas in this paper are also valid for the squared case, however, the proofs are more involved and the correspondence between formulations is less straight-forward.

The ν-SVM is traditionally formulated as (2) with the L_2 norm, where $\nu \in [0,1]$ instead of $C \in [0,+\infty)$ is chosen as a fixed tradeoff parameter.

$$\begin{aligned} \min_{w,b,\rho,\xi} \quad & \frac{1}{2}\|w\|_2^2 - \nu\rho + \sum_{i=1}^{N} \xi_i \\ s.t. \quad & \xi_i \geq -y_i(w^T X_i + b) + \rho, \quad \forall i \in \{1,...,N\}, \\ & \xi \geq 0. \end{aligned} \tag{2}$$

As already mentioned, the ν-SVM advantageously gives us a free parameter, $\nu \in [0,1]$, with implied meaning. The meaning of ν is shown in Schölkopf et al. (2000) by proving a variant of Property 1.

Property 1 *Assume that there exists a feasible solution (w,b,ρ,ξ) for (2) for parameter choice $\nu \in [0,1]$. Then the following bounds apply:*

- *The choice of ν acts as an upper bound on the fraction of errors in the margin:*

$$1-\alpha < \nu, \quad \textit{where} \quad 1-\alpha = \frac{1}{N}\left|\{i : y_i(w^T X_i + b) < \rho\}\right|.$$

- *The choice of ν acts as a lower bound on the fraction of support vectors (SV's), support vectors being errors that lie in the margin or on the margin boundary:*

$$\%SV's > \nu, \quad \textit{where} \quad \%SV's = \frac{1}{N}\left|\{i : y_i(w^T X_i + b) \leq \rho\}\right|.$$

2.3 The Eν-SVM

Given the natural interpretation for the meaning of the ν-parameter, it would seem normal to assume that all the values of $\nu \in [0,1]$ will yield non-trivial, feasible solutions satisfying the bounds stated in Property 1. This, though, is not the case. In Chang and Lin (2001), it was shown that the ν-parameter has a limited range. Specifically, a variant of Property 2 is proved in Chang and Lin (2001).

Property 2

- *There exists a minimum and maximum value such that $\nu \in (\nu_{\min}, \nu_{\max}]$ yields feasible (2) with non-trivial solutions, $\nu \leq \nu_{\min}$ yields (2) with trivial optimal solution $w = b = 0$, and $\nu > \nu_{\max}$ yields infeasible (2).*

- *Furthermore, this limitation in parameter range applies to the C-SVM as well. Specifically, there exists a correspondence in allowable range such that $\nu \to \nu_{\min}$ corresponds to $C \to \infty$ and $\nu \to \nu_{\max}$ corresponds to $C \to 0$.*

To solve this issue, extending the valid range of the ν-parameter to be the entire $[0,1]$ interval, Pérez-Cruz et al. (2003) developed an extended ν-SVM formulation called Eν-SVM (3). The Eν-SVM is traditionally formulated as follows with the L_2 norm. We

present it with the general norm as follows:

$$\begin{aligned} \min_{w,b,\rho,\xi} \quad & -\nu\rho + \sum_{i=1}^{N} \xi_i \\ s.t. \quad & \xi_i \geq -y_i(w^T X_i + b) + \rho, \quad \forall i \in \{1,...,N\}, \\ & \xi \geq 0, \\ & \|w\| = 1. \end{aligned} \tag{3}$$

One can view (3) as an **extension** of (2) in that Pérez-Cruz et al. (2003); Chang and Lin (2001) showed that the optimal solution to (2), formulated with L_2 norm and any $\nu_0 \in (\nu_{\min}, v_{\max}]$, is also an optimal solution to (3), formulated with L_2 norm, for some $\nu_1 \leq \nu_0$. Problem (3), though, can achieve solutions that (2) cannot because of its extended range of the ν-parameter.

3. Superquantiles and the Eν-SVM

In this section, we first give a brief review of the superquantile concept as introduced by Rockafellar and Uryasev (2002). We then review the results of Takeda and Sugiyama (2008), showing that the Eν-SVM is equivalent to superquantile minimization.

3.1 Superquantiles and Tail Probabilities

When working with optimization of tail probabilities, one frequently works with constraints or objectives involving *probability of exceedance* (POE), $p_z(Z) = P(Z > z)$, or its associated quantile $q_\alpha(Z) = \min\{z | P(Z \leq z) \geq \alpha\}$, where $\alpha \in [0,1]$ is a probability level and $z \in \mathbb{R}$ is a fixed threshold level. The quantile is a popular measure of tail probabilities in financial engineering, called within this field Value-at-Risk by its interpretation as a measure of tail risk. The quantile, though, when included in optimization problems via constraints or objectives is quite difficult to treat with continuous (linear or non-linear) optimization techniques.

A significant advancement was made by Rockafellar and Uryasev (2002) in the development of an approach to combat the difficulties raised by the use of the quantile function in optimization. They explored a replacement for the quantile, called CVaR within the financial literature and called the superquantile in a general context. The superquantile is a measure of uncertainty similar to the quantile, but with superior mathematical properties. Formally, the superquantile (CVaR) for a continuously distributed real valued random variable Z is defined as,

$$\bar{q}_\alpha(Z) = E\left[Z | Z > q_\alpha(Z)\right]. \tag{4}$$

For general distributions, the superquantile can be defined by the following formula,

$$\bar{q}_\alpha(Z) = \min_{\gamma} \quad \gamma + \frac{E[Z-\gamma]^+}{1-\alpha}, \tag{5}$$

where $[\cdot]^+ = \max\{\cdot, 0\}$. For a discretely distributed random variable Z with equally probable realizations $\{Z_1, Z_2, ..., Z_N\}$ we can write this formula as the following Linear Program-

ming problem,

$$\begin{aligned} \bar{q}_\alpha(Z) = \min_{\gamma,\xi} \quad & \gamma + \frac{1}{N(1-\alpha)} \sum_{i=1}^{N} \xi_i \\ s.t. \quad & \xi_i \geq Z_i - \gamma, \forall i \in \{1, ..., N\}, \\ & \xi \geq 0. \end{aligned} \tag{6}$$

Similar to $q_\alpha(Z)$, the superquantile can be used to assess the tail of the distribution. The superquantile, though, is far easier to handle in optimization contexts. It also has the important property that it considers the magnitude of events within the tail. Therefore, in situations where a distribution may have a heavy tail, the superquantile accounts for magnitudes of low-probability large-loss tail events while the quantile does not account for this information.

3.2 Eν-SVM as Superquantile Minimization

In Takeda and Sugiyama (2008), the meaning of the ν-parameter was solidified by showing that the Eν-SVM, (3), is equivalent to superquantile minimization. Specifically, they proved a variant of Property 3.

Property 3 *Consider optimization problem (3). Let $\alpha = 1 - \nu$ and $\gamma = -\rho$. Also, let $L(w, b, X, y) = -y(w^T X + b)$ be a discretely distributed random variable with equally probable realizations $\{-y_1(w^T X_1 + b), ..., -y_N(w^T X_N + b)\}$. With this notation, (3) can be rewritten as (7), which is equivalent to (8), minimization of the α-superquantile:*

$$\begin{aligned} \min_{w,b,\gamma,\xi} \quad & (1-\alpha)\left(\gamma + \frac{1}{N}\sum_{i=1}^{N} \xi_i\right) \\ s.t. \quad & \xi_i \geq -y_i(w^T X_i + b) - \gamma, \quad \forall i \in \{1, ..., N\}, \\ & \xi \geq 0, \\ & \|w\| = 1. \end{aligned} \tag{7}$$

$$\begin{aligned} \min_{w,b} \quad & (1-\alpha)\bar{q}_\alpha\left(-y(w^T X + b)\right) \\ s.t. \quad & \|w\| = 1. \end{aligned} \tag{8}$$

With this, one can see that the Eν-SVM is simply minimization of the value (5) multiplied by $1 - \alpha$ with the real valued discretely distributed random loss $L(w, b, X, y) = -y(w^T X + b)$ in place of the real valued random variable Z.

4. Buffered Probability of Exceedance (bPOE)

In this section, we first review the concept of bPOE. We show how it is simply one minus the inverse of the superquantile and review its surprising calculation formula. We then review how minimization of bPOE integrates quite nicely into optimization frameworks. Finally, we present a slightly altered formulation for minimization of bPOE in the presence of Positive

Homogenous (PH) random functions. For the interested reader, we discuss the necessity of this alteration and derive the formulation in Appendix A. We move this discussion to the appendix, as it is slightly distracting from the discussion related to support vector machines.

4.1 bPOE: Inverse of the Superquantile

As mentioned in Section 3, when working with optimization of tail probabilities, one frequently works with constraints or objectives involving POE, $p_z(Z) = P(Z > z)$, or its associated quantile $q_\alpha(Z) = \min\{z | P(Z \leq z) \geq \alpha\}$, where $\alpha \in [0, 1]$ is a probability level and $z \in \mathbb{R}$ is a fixed threshold level. The superqunatile was developed to alleviate difficulties associated with optimization problems involving quantiles. Working to extend this concept, bPOE was developed as the inverse of the superquantile in the same way that POE is the inverse of the quantile.

Specifically, there exist two slightly different variants of bPOE, namely Lower and Upper bPOE. Mafusalov and Uryasev (2015) mainly work with so called Lower bPOE while Norton and Uryasev (2014) work with so called Upper bPOE. These definitions do not differ dramatically and a discussion of these differences is beyond the scope of this paper. Thus, for the remainder of this paper when we refer to bPOE, we are utilizing Upper bPOE. With this in mind, bPOE is defined in the following way, where $\sup Z$ denotes the essential supremum of random variable Z. In this paper we assume all random variables to be L_1-finite, $Z \in \mathcal{L}^1(\Omega)$, i.e. $E|Z| < \infty$.

Definition 1 *Upper bPOE for a random variable Z at a threshold z equals,*

$$\bar{p}_z(Z) = \begin{cases} \max\{1 - \alpha | \bar{q}_\alpha(Z) \geq z\}, & \text{if } z \leq \sup Z, \\ 0, & \text{otherwise.} \end{cases}$$

In words, for any threshold $z \in (E[Z], \sup Z)$, bPOE can be interpreted as one minus the probability level at which the superquantile equals z. Although bPOE seems troublesome to calculate, Norton and Uryasev (2014) provide the following calculation formula for bPOE.

Proposition 1 *Given a real valued random variable Z and a fixed threshold z, bPOE for random variable Z at z equals,*

$$\bar{p}_z(Z) = \inf_{\gamma < z} \frac{E[Z-\gamma]^+}{z-\gamma} = \begin{cases} \lim\limits_{\gamma \to -\infty} \frac{E[Z-\gamma]^+}{z-\gamma} = 1, & \text{if } z \leq E[Z], \\ \min\limits_{\gamma < z} \frac{E[Z-\gamma]^+}{z-\gamma}, & \text{if } z \in (E[Z], \sup Z), \\ \lim\limits_{\gamma \to z^-} \frac{E[Z-\gamma]^+}{z-\gamma} = P(Z = \sup Z), & \text{if } z = \sup Z, \\ \min\limits_{\gamma < z} \frac{E[Z-\gamma]^+}{z-\gamma} = 0, & \text{if } \sup Z < z. \end{cases} \tag{9}$$

It is also important to note that formula (9) has the following property, Property 4, proved in Mafusalov and Uryasev (2015) and Norton and Uryasev (2014). This property will become important in later sections when we begin to interpret the EC-SVM. Note that for Property 4, we must distinguish between the *lower* quantile $q_\alpha(Z)$ and *upper* quantile $q^+_\alpha(Z) = \inf\{z | P(Z \leq z) > \alpha\}$. However, the difference in these quantities will likely

be small or zero. For continuously distributed Z, we will have $q_\alpha(Z) = q^+_\alpha(Z)$ or if Z is discretely distributed with N equally probably events, and N is reasonably large, the difference $q^+_\alpha(Z) - q_\alpha(Z)$ will likely be small, possibly zero.

Property 4 *If* $z \in (E[Z], \sup Z)$ *and* $\min_{\gamma<z} \frac{E[Z-\gamma]^+}{z-\gamma} = \frac{E[Z-\gamma^*]^+}{z-\gamma^*} = 1-\alpha^*$*, then:*

$$\bar{p}_z(Z) = 1-\alpha^*, \qquad \bar{q}_{\alpha^*}(Z) = z, \qquad \gamma^* \in [q_{\alpha^*}(Z), q^+_{\alpha^*}(Z)],$$

where $[q_{\alpha^*}(Z), q^+_{\alpha^*}(Z)]$ *is the entire set of minimizers.*

Thus, using formula (9), bPOE can be efficiently calculated. Additionally, Property 4 shows that we can recover quantile and superquantile information. As we demonstrate in the next section, formula (9) also allows for convenient optimization of bPOE.

4.2 Optimization of bPOE for Random PH Functions

Norton and Uryasev (2014) considered the following optimization setup to demonstrate the ease with which bPOE can be minimized directly. Assume we have a real valued positive homogenous (PH) random function $f(w, X)$ determined by a vector of control variables $w \in \mathbb{R}^n$ and a random vector X. By definition, a function $f(w, X)$ is PH w.r.t. w if it satisfies the following condition: $af(w, X) = f(aw, X)$ for any $a \geq 0, a \in \mathbb{R}$.

Now, assume that we would like to find the vector of control variables, $w \in \mathbb{R}^n$, that minimizes the probability of $f(w, X)$ exceeding a threshold $z \in \mathbb{R}$. We would like to solve the POE optimization problem,

$$\min_{w\in\mathbb{R}^n} \quad p_z(f(w, X)). \tag{10}$$

Here we have a discontinuous and non-convex objective function (assuming a discretely distributed X) that is numerically difficult to minimize. Consider alternatively minimization of bPOE instead of POE at the same threshold z. This is posed as the optimization problem,

$$\min_{w\in\mathbb{R}^n} \quad \bar{p}_z(f(w, X)). \tag{11}$$

Given Proposition 1, (11) can be transformed into the following:

$$\min_{w\in\mathbb{R}^n,\gamma<z} \quad \frac{E[f(w, X)-\gamma]^+}{z-\gamma}. \tag{12}$$

Paper Norton and Uryasev (2014), though, limit consideration to only a threshold of $z = 0$, in which case formulation (12) reduces to,

$$\min_{w\in\mathbb{R}^n} \quad E[f(w, X)+1]^+. \tag{13}$$

As we discuss in Appendix A, formulation (12) has shortcomings for nonzero thresholds. Specifically, it fails to achieve varying optimal solutions for varying threshold levels. To address these issues, the next section provides an alternative formulation for bPOE minimization with PH functions $f(w, X)$ that allows effective variation of threshold levels.

4.3 An Altered Formulation

With (12) failing to achieve varying optimal solutions as the threshold z varies, we find that adding a constraint on the norm of w remedies this situation (because w can no longer rescale as the threshold changes). Here $\|\cdot\|$ denotes any general norm. This gives us

$$\begin{array}{ll} \min\limits_{w\in\mathbb{R}^n,\gamma<z} & \dfrac{E[f(w, X)-\gamma]^+}{z-\gamma} \\ s.t. & \|w\| = 1 \end{array} \quad \equiv \quad \min_{w\in\mathbb{R}^n} \bar{p}_z\left(\frac{f(w, X)}{\|w\|}\right). \tag{14}$$

Furthermore, the following Proposition 2 shows that (14) can be simplified, yielding

$$\min_{w\in\mathbb{R}^n} \quad E[f(w, X) - z\|w\| + 1]^+. \tag{15}$$

Proposition 2 *Assume* $f(w, X)$ *is PH with respect to* w*. If* (w^*, γ^*) *is optimal to (14) with optimal objective* $1-\alpha^*$*, then* $w = \frac{w^*}{z-\gamma^*}$ *is optimal to (15) with optimal objective* $1-\alpha^*$*.*

Proof For this, we show that (15) is formed only by making a change of variable in (14). We start with (14). Since $\gamma < z$ is an explicit constraint and thus $z - \gamma > 0$, we bring the denominator into the expectation in the numerator to get

$$\begin{array}{ll} \min\limits_{w\in\mathbb{R}^n,\gamma<z} & E\left[f\left(\dfrac{w}{z-\gamma}, X\right) - \left(\dfrac{\gamma}{z-\gamma}\right)\right]^+ \\ s.t. & \|w\| = 1. \end{array} \tag{16}$$

Now, make the change of variable $w_{new} = \frac{w}{z-\gamma}$. Since we have the explicit constraint $\|w\| = 1$, we have that $\|w_{new}\| = \left\|\frac{w}{z-\gamma}\right\| = \frac{\|w\|}{z-\gamma} = \frac{1}{z-\gamma}$. We can then make the change of variable to get

$$\min_{w_{new}\in\mathbb{R}^n,\gamma<z} \quad E\left[f(w_{new}, X) - \|w_{new}\|\gamma\right]^+. \tag{17}$$

We can also rearrange $\|w_{new}\| = \frac{1}{z-\gamma}$ to get $\gamma = z - \frac{1}{\|w_{new}\|}$ and thus that $\|w_{new}\|\gamma = \|w_{new}\|\left(z - \frac{1}{\|w_{new}\|}\right) = z\|w_{new}\| - 1$. Plugging this into our formulation, we arrive at

$$\min_{w_{new}\in\mathbb{R}^n} \quad E[f(w_{new}, X) - z\|w_{new}\| + 1]^+, \tag{18}$$

where due to our change in variable, we see that if we have optimal solution w^*_{new}, then $\left(w = \frac{w^*_{new}}{\|w_{new}\|}, \gamma = z - \frac{1}{\|w^*_{new}\|}\right)$ is optimal to (14) before the change of variable. ■

Thus, we can turn to (15) as our formulation for bPOE minimization of a PH function for varying threshold choices. Notice from the proof of Proposition 2 that if w^* is optimal to (15) then $(w = \frac{w^*}{\|w^*\|}, \gamma = z - \frac{1}{\|w^*\|})$ is optimal to (14). Also, let us call $f\left(\frac{w^*}{\|w^*\|}, X\right)$ the normalized loss distribution at w^*. Given Property 4, if

$$E[f(w^*, X) - z\|w^*\| + 1]^+ = 1-\alpha^*,$$

we know the following about the normalized loss distribution at the optimal point w^*:

$$\bar{p}_z(F) = 1-\alpha^*, \quad \bar{q}_{\alpha^*}(F) = z, \quad z-\frac{1}{\|w^*\|} \in [q_{\alpha^*}(F), q^+_{\alpha^*}(F)], \quad \text{where } F := f\left(\frac{w^*}{\|w^*\|}, X\right).$$

5. Extended C-SVM

In this section, we use formula (15) to introduce the EC-SVM. We approach the classification problem via a natural bPOE minimization problem. Utilizing the interpretability of optimization problem (15), we show that the EC-SVM is also simple to interpret. Specifically, we show that application of Property 4 allows us to interpret the choice of parameter and the value of the optimal objective in interesting, purely statistical ways.

We also point out that the traditional *hinge loss function* naturally occurs when minimizing bPOE, meaning that we do not need to explicitly specify it as a loss function. This can be seen clearly by considering minimization of bPOE at threshold $C = 0$. Additionally, it is interesting to notice that we do not explicitly attempt to regularize via use of norms, as the use of norms naturally arises from (14).

5.1 bPOE Minimization with SVM Loss

Consider the formula for bPOE minimization (15) with discretely distributed random loss $L(w, b, X, y) = -y(w^T X+b)$ with equally probable realizations $\{-y_1(w^T X_1+b), ..., -y_N(w^T X_N+b)\}$ and threshold $z \in \mathbb{R}$. If we let $C = -z$ and multiply the objective function by N, this gives us the following optimization problem, which we call the EC-SVM:

$$\begin{array}{ll} \min\limits_{w,b,\xi} & \sum\limits_{i=1}^{N} \xi_i \\ s.t. & \xi_i \geq -y_i(w^T X_i + b) + C\|w\| + 1, \quad \forall i \in \{1, \ldots, N\}, \\ & \xi \geq 0. \end{array} \tag{19}$$

Notice that the norm $\|\cdot\|$, just as in (15), is an arbitrary norm in $\mathbb{R}^n$. Notice also that the EC-SVM is a very natural formulation. It is simply a *buffered* way of minimizing the probability that misclassification errors exceed our threshold $-C$. Put specifically, instead of finding the classifier (w, b) that minimizes $p_{-C}\left(-y(w^T X + b)\right) = P\left(-y(w^T X + b) > -C\right)$, we are minimizing its buffered variant, $\bar{p}_{-C}\left(-y(w^T X + b)\right)$.

5.2 Occurrence of Hinge Loss

When discussing SVM's, it is traditional for people to say that the C-SVM minimizes a *hinge loss function* plus a regularization term on the vector w. For the EC-SVM, though, we see that this is not an accurate description, as we do not need to explicitly specify a hinge loss function. We see that it naturally arises when minimizing bPOE. Specifically, minimizing bPOE of the loss function $L(w, b, X, y) = -y(w^T X + b)$ at threshold $C = 0$ yields the following problem, which is exactly minimization of hinge losses:

$$\min_{w,b} \quad \sum_{i=1}^{N} \left[-y_i(w^T X_i + b) + 1\right]^+. \tag{20}$$

One can see this more generally by looking at (13), where we are minimizing bPOE of a PH loss function $f(w, X)$ at threshold zero.

5.3 Interpretability of EC-SVM

Since the EC-SVM is simply bPOE minimization, we can utilize Property 4 to interpret the C-parameter and the value of the optimal objective in an exact way. Specifically, let us put Property 4 in terms of the EC-SVM.

Property 5 *Suppose (19), with any general norm, yields optimal hyperplane (w^*, b^*) and the optimal objective value equal to obj^*. Considering $L(w^*, b^*, X, y) = -y(w^{*T} X + b^*)$ as a discretely distributed random loss, we know the following about the normalized loss distribution.*

- $\bar{p}_{-C}\left(\frac{L(w^*,b^*,X,y)}{\|w^*\|}\right) = 1 - \alpha^* = \frac{obj^*}{N}$,
- $\bar{q}_{\alpha^*}\left(\frac{L(w^*,b^*,X,y)}{\|w^*\|}\right) = -C$,
- $q_{\alpha^*}\left(\frac{L(w^*,b^*,X,y)}{\|w^*\|}\right) \leq -C - \frac{1}{\|w^*\|} \leq q^+_{\alpha^*}\left(\frac{L(w^*,b^*,X,y)}{\|w^*\|}\right)$.

5.3.1 The C-parameter as Superquantile Threshold Choice

For the C-SVM, the non-negative C-parameter is typically discussed as being a tradeoff between errors and margin size. In the broad scheme of Empirical Risk Minimization (ERM), this parameter is discussed as the tradeoff between risk and regularization. For the EC-SVM, we provide a much more concrete interpretation. With the EC-SVM being exactly bPOE minimization, the C-parameter is a choice of threshold $z = -C$. Specifically, looking at Property 5, we have that

$$\bar{q}_{\alpha^*}\left(\frac{L(w^*, b^*, X, y)}{\|w^*\|}\right) = -C,$$

where $\alpha^* = 1 - \frac{obj^*}{N} = 1 - \bar{p}_{-C}\left(\frac{L(w^*,b^*,X,y)}{\|w^*\|}\right)$.

5.3.2 The Optimal Objective Value as bPOE

The EC-SVM also has the surprising property that the optimal objective value, divided by the number of samples, is a probability level. More specifically, as shown in Property 5, we have that $\frac{obj^*}{N} = \bar{p}_{-C}\left(\frac{L(w^*,b^*,X,y)}{\|w^*\|}\right)$. In words, the optimal objective value divided by the number of samples equals bPOE of the optimal normalized loss distribution at threshold $-C$.

5.4 Norm Choice and Margin Interpretation

In this section, we show that the EC-SVM has a clear interpretation of 'margin,' which is dependent on the choice of norm. This interpretation shows, in an exact way, how the C parameter determines the margin of the separating hyperplane.

Looking back to formula (14), using loss function $f(w, X) = -y(w^T X + b)$, and making the change of variable $w \to \frac{w}{C}, b \to \frac{b}{C}$ we are able to formulate the following equivalent problem (for the full derivation, see Appendix E):

$$\begin{aligned} \min_{\gamma<-C,w,b} \quad & \sum_{i=1}^{n} \left[\left(\frac{1}{-C-\gamma} \right) \left(\frac{-y_i(w^T X_i + b) + 1}{\|w\|} \right) + 1 \right]^+ \\ s.t. \quad & \frac{1}{\|w\|} = C. \end{aligned} \tag{21}$$

From this formulation, we can form an interpretation of the EC-SVM within the context of selecting an optimal hyperplane under a 'hard margin' criterion. To make this interpretation clear, we can start by analyzing (21) formulated with the L_2 norm, traditional to the C-SVM and discussions of 'maximal margin hyperplanes.' In this context, we see that the constraint $\frac{1}{\|w\|_2} = C$ is fixing the *euclidean* distance between hyperplanes $w^T X + b = 1$ and $w^T X + b = -1$, i.e. fixing the margin to be equal to C. Also, we see that the directional distance from X_i to the corresponding "separating" hyperplane is equal to $\frac{-y_i(w^T X_i + b) + 1}{\|w\|_2}$. The optimization problem above, therefore, fixes the margin between "separating" hyperplanes and minimizes the buffered probability of margin violations. If the classes are linearly separable with a margin of at least C, then the optimal objective is 0, meaning that "separating" hyperplanes are indeed separating. However, when classes are not separable with a margin of at least C, optimization problem (21) finds the number of worst classified objects such that the average of their directional distances to corresponding hyperplanes equals to 0. This number of worst classified objects is then minimized subject to the fixed margin size.

This interpretation, though, extends to any general norm within formulation (21). Using euclidean distance as our metric for measuring distances in $\mathbb{R}^n$, geometry tells us that the distance between hyperplanes is given by $\frac{2}{\|w\|_2}$ and that the directional distance from X_i to the corresponding "separating" hyperplane equals to $\frac{-y_i(w^T X_i + b) + 1}{\|w\|_2}$. But what if we were to use a different metric to measure distances within $\mathbb{R}^n$? For example, what if we were to say that the 'distance' between two points X_1, X_2 was $\|X_1 - X_2\|_1$ instead of $\|X_1 - X_2\|_2$? In this case, the distance between hyperplanes is given by $\frac{2}{\|w\|_\infty}$. This follows from the concept of the dual norm.

Denote by $\|\cdot\|^*$ the norm dual to the norm $\|\cdot\|$. In the general case, we know that if the 'distance' between two points X_1, X_2 is $\|X_1 - X_2\|^*$, then the distance between hyperplanes $w^T X + b = 1$ and $w^T X + b = -1$ is equal to $\frac{2}{\|w\|}$ and that the directional distance from X_i to the corresponding "separating" hyperplane equals to $\frac{-y_i(w^T X_i + b) + 1}{\|w\|}$. Thus, for the EC-SVM formulated with general norm $\|\cdot\|$, the constraint $\frac{1}{\|w\|} = C$ is fixing the 'margin' equal to C in $\mathbb{R}^n$ under the implied distance metric defined by the dual norm, $\|\cdot\|^*$.

Note that the problem above is equivalent to the EC-SVM, (19), having the same optimal objective and (up to some scaling factor) equivalent optimal hyperplane when parameter C is the same for both problems. Therefore, the interpretation above is also valid for the EC-SVM. In particular, if a certain norm $\|\cdot\|$ is used in the optimization problem setting, then it is implied that distances between objects represent object similarities in a better fashion when measured according to the dual norm. The correspondence between C-SVM and EC-SVM, described further by Theorems 1 and 2, is not as direct as the correspondence

between (21) and the EC-SVM. However, the presence of this correspondence should also imply that the use of $\|\cdot\|$ in optimization problem is closely related to the choice of norm $\|\cdot\|^*$ for the considered space.

Thus, as opposed to the C-SVM formulation, the EC-SVM formulation has a clear interpretation as hard margin separation problem for the case of non-separable classes.

6. Connecting the EC-SVM and C-SVM

In this section, we prove the equivalence of the EC-SVM and C-SVM when formulated with any general norm. Theorems 1 and 2 present the main result, providing a direct correspondence between parameter choices and optimal solutions. We emphasize the critical implication, which is that solving the C-SVM with any parameter $\hat{C} \geq 0$ is equivalent to solving the EC-SVM (i.e. minimizing bPOE) for some parameter $C \geq 0$.

Below be prove the equivalence of C-SVM and EC-SVM with two theorems. We leave the lengthy proofs to Appendix B. Theorem 1 begins with the assumption that one has solved the C-SVM and then provides the proper parameter value for which the EC-SVM will yield the same optimal hyperplane, up to a specific scaling factor, which we also provide. Theorem 2 is analogous to Theorem 1, but begins with the assumption that one has solved the EC-SVM. It should be noted that the theorems reference dual variables, which are discussed more explicitly (via KKT conditions) within the proofs of the theorems.

Theorem 1 *Assume that the data set is not linearly separable and suppose optimization problem (1) is formulated with any general norm $\|\cdot\|$ and some parameter $\hat{C} \geq 0$ and that it has optimal primal variables (w^*, b^*, ξ^*) and optimal dual variables (α^*, β^*). Then (19), formulated with corresponding norm and parameter $C = \frac{\hat{C}}{\sum_{i=1}^N \beta_i^*}$, will have optimal primal variables $(w = \mu w^*, b = \mu b^*, \xi = \mu \xi^*)$ and optimal dual variables $(\alpha = \alpha^*, \beta = \beta^*)$, where $\mu = \frac{\sum_{i=1}^N \beta_i^*}{\sum_{i=1}^N \beta_i^* - \hat{C}\|w^*\|} > 0$.*

Theorem 2 *Suppose optimization problem (19) is formulated with any general norm $\|\cdot\|$ and some parameter $C \geq 0$ and that it has optimal primal variables (w^*, b^*, ξ^*) and optimal dual variables (α^*, β^*). Then (1), formulated with corresponding norm and parameter $\hat{C} = C \sum_{i=1}^N \beta_i^*$, will have optimal primal variables $(w = \mu w^*, b = \mu b^*, \xi = \mu \xi^*)$ and optimal dual variables $(\alpha = \alpha^*, \beta = \beta^*)$, where $\mu = \frac{1}{1 + C\|w^*\|} > 0$.*

7. Presentation as Cohesive Structure

Here, we show exactly why it is appropriate to call formulation (19) the EC-SVM. Specifically, we show that the EC-SVM is equivalent to the Eν-SVM, producing the same set of optimal solutions. This follows directly from the fact that bPOE is the inverse of the superquantile. This fact helps solidify the idea that the EC-SVM is an **extension** of the C-SVM in the same way that the Eν-SVM is an extension of the ν-SVM. First, as was proved in Section 6, the optimal solution set produced by the C-SVM over $C \in [0, \infty)$ is contained in the optimal solution set produced by the EC-SVM over $C \in (-\infty, \infty)$. Second, the EC-SVM extends the allowable range of the C parameter to negative C-values. Thus,

just as the Eν-SVM extends the parameter range and optimal solution set of the ν-SVM, the EC-SVM does the same for the C-SVM.

Proposition 3 *The EC-SVM and Eν-SVM, formulated with the same general norm, produce the same set of optimal hyperplanes.*

Proof First, recall that the EC-SVM is equivalent to bPOE minimization. Second, recall that the Eν-SVM is equivalent to superquantile minimization. Using these two facts, the equivalence follows immediately from Mafusalov and Uryasev (2015), which shows that (w, b) is a minimizer of bPOE at some threshold level if and only if (w, b) is a minimizer of the superquantile at some probability level. ■

To help gather all of the results we have discussed, we can present them as a cohesive structure in the following table:

$\hat{C} \in [0, +\infty)$		$C \in (-\infty, +\infty)$	$=$	$-z$
$\uparrow$		$\uparrow$		$\uparrow$
C-SVM	$\subset$	EC-SVM	$\equiv$	$\min\limits_{w,b} \ \bar{p}_z\left(-y(w^T X + b)\right)$
$\Updownarrow$		$\Updownarrow$		$\Updownarrow$
ν-SVM	$\subset$	Eν-SVM	$\equiv$	$\min\limits_{w,b} \ \bar{q}_\alpha\left(-y(w^T X + b)\right)$
$\downarrow$		$\downarrow$		$\downarrow$
$\hat{\nu} \in (\nu_{\min}, \nu_{\max}]$		$\nu \in [0, 1]$	$=$	$1 - \alpha$

Key:

- $\Updownarrow$ Formulations generate the same set of optimal solutions.
- $\subset$ The right hand side formulation is an "extension" of the left hand formulation. (i.e. in the way that Eν-SVM is an extension of ν-SVM)
- $\equiv$ Formulations are objectively equivalent.
- $\uparrow$ or $\downarrow$ Arrow points to parameter values for the formulation.

8. Dual Formulations and Kernalization

Typically, the C-SVM is presented first in its primal form, then in its dual form, as the latter provides insights into selection of support vectors via dual variables while additionally providing a quadratic optimization problem for solving the C-SVM (in the case of the L_2 regularization norm). This quadratic optimization problem also allows one to use the 'kernel trick,' utilizing the presence of dot products to introduce a non-linear mapping to a high dimensional feature space.

In this section, we present the dual formulations of the C-SVM, (1), and EC-SVM, (19), formulated with any general norm. We leave the derivation of these dual formulations to Appendix C. We use these formulations to enlighten our perspective in two ways. First, in conjunction with Theorems 1 and 2, we use the dual formulations to show that the optimal objective value of the C-SVM and EC-SVM coincide when yielding the same optimal hyperplane. This effectively yields the surprising result that the optimal value of the C-SVM objective function, divided by sample size, equals a probability level. Second, we use these dual formulations to present kernalization of the EC-SVM when formulated with the L_2 norm.

Assume the EC-SVM, (19), is formulated with some norm $\|\cdot\|$ and $C \geq 0$. Let $\|\cdot\|^*$ be the corresponding dual norm, then the dual formulation is as follows:

$$\begin{aligned} \max_{\beta} \quad & \sum_{i=1}^{N} \beta_i \\ s.t. \quad & \left\| \sum_{i=1}^{N} \beta_i y_i X_i \right\|^* \leq C \sum_{i=1}^{N} \beta_i, \\ & \sum_{i=1}^{N} \beta_i y_i = 0, \\ & 0 \leq \beta_i \leq 1, \quad \forall i \in \{1, \ldots, N\}. \end{aligned} \tag{22}$$

Assume the C-SVM, (1), is formulated with some norm $\|\cdot\|$ and $C \geq 0$. Let $\|\cdot\|^*$ be the corresponding dual norm, then the dual formulation is as follows:

$$\begin{aligned} \max_{\beta} \quad & \sum_{i=1}^{N} \beta_i \\ s.t. \quad & \left\| \sum_{i=1}^{N} \beta_i y_i X_i \right\|^* \leq C, \\ & \sum_{i=1}^{N} \beta_i y_i = 0, \\ & 0 \leq \beta_i \leq 1, \quad \forall i \in \{1, \ldots, N\}. \end{aligned} \tag{23}$$

Given the dual formulations, it follows immediately from Theorems 1 and 2 that the optimal dual objective solutions coincide when parameters are chosen so that the EC-SVM and C-SVM produce the same optimal hyperplane. Additionally, this result applies to the primal formulations via strong duality. Thus, since the EC-SVM objective, divided by sample size, equals a probability level (as seen in Property 6), we can conclude that the optimal objective value of the C-SVM primal formulation, divided by sample size, also equals a probability level.

8.1 Kernalization of EC-SVM

Here we briefly present various methods for utilizing the kernel trick with the EC-SVM. First, the most obvious way is for the L_2 norm case and the dual EC-SVM formulation. To effectively apply the kernel trick[2], we need only to square the constraint in the dual formulation

$$\left\| \sum_{i=1}^{N} \beta_i y_i X_i \right\|^* \leq C \sum_{i=1}^{N} \beta_i \ ,$$

2. Recall that the L_2 norm is self-dual.

forming the following convex quadratically constrained quadratic program (QCQP) (24), where $\phi(X_i)$ represents a non-linear kernel mapping of the i_{th} data vector. Note that this can be efficiently solved with most convex optimization software.

$$\begin{aligned} \max_{\beta} \quad & \sum_{i=1}^{N} \beta_i \\ s.t. \quad & \sum_{i=1}^{N}\sum_{i=1}^{N} \beta_i\beta_j \left(y_i y_j \phi(X_i)^T\phi(X_j) - C^2\right) \leq 0, \\ & \sum_{i=1}^{N} \beta_i y_i = 0, \\ & 0 \leq \beta_i \leq 1, \quad \forall i \in \{1,\ldots,N\}. \end{aligned} \tag{24}$$

The application of the kernel trick, though, is not limited to the dual formulation. Mimicking the work of Chapelle (2007), we can apply the kernel in the primal formulated with the L_2 norm by using the representers theorem of Schölkopf et al. (2001). Since the representers theorem applies to the C-SVM, it indeed can be applied to the equivalent convex EC-SVM.[3] Thus, we know that in the high dimensional feature space we can represent the optimal solution of the convex EC-SVM as $w = \sum_i \theta_i y_i \phi(X_i)$. Therefore, using this representation, we can write the primal EC-SVM with $C \geq 0$ and the L_2 norm as the following unconstrained non-smooth convex program, where $K_{ij} = \phi(X_i)^T\phi(X_j)$.

$$\min_{\theta,b} \sum_{i=1}^{N}[-y_i(\sum_j \theta_j y_j K_{ij} + b) + C\left(\sum_i\sum_j \theta_i\theta_j y_i y_j K_{ij}\right)^{\frac{1}{2}} + 1]^+ . \tag{25}$$

Furthermore, this can be reformulated as the following convex QCQP.

$$\begin{aligned} \min_{\theta,b,\xi,t} \quad & \sum_{i=1}^{N} \xi_i \\ s.t. \quad & \xi_i \geq -y_i(\sum_j \theta_j y_j K_{ij} + b) + Ct + 1, \quad \forall i \in \{1,\ldots,N\}. \\ & \xi \geq 0 \\ & t^2 \geq \sum_i\sum_j \theta_i\theta_j y_i y_j K_{ij}. \end{aligned} \tag{26}$$

So far, we have discussed the case of the L_2 norm and $C \geq 0$. Here, for the interest of the reader, we briefly mention how to apply a standard technique which allows approximate kernelization of the EC-SVM with any norm and $C \in \mathbb{R}$. Here, we explicitly map the data vectors into the high dimensional feature space by approximating the kernel map. Let $K \in \mathbb{R}^{N\times N}$ be the positive semi-definite (PSD) matrix such that $K_{ij} = \phi(X_i)^T\phi(X_j)$ and, with abuse of notation, let $X = [X_1, ..., X_N]$ be the matrix of data vectors. Let

3. In other words, because the EC-SVM and C-SVM produce the same set of optimal solutions for $C \geq 0$, we can say that their optimal solutions can be represented in the same way.

$V = [V_1, ..., V_N]$ and $\Lambda = diag\{\lambda_1, ..., \lambda_n\}$ be the matrix of eigenvectors and associated eigenvalues of K with $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$. With K being symmetric PSD, we then have the eigendecomposition,

$$K = \phi(X)^T\phi(X) = V\Lambda V^T = (\Lambda^{\frac{1}{2}}V^T)^T(\Lambda^{\frac{1}{2}}V^T) .$$

Thus, we can approximate $\phi(X)$ by taking the top d eigenvectors and eigenvalues to get

$$\hat{X} = diag\{\lambda_1, ..., \lambda_d\}^{\frac{1}{2}}[V_1, ..., V_d]^T \approx \Lambda^{\frac{1}{2}}V^T = \phi(X) .$$

Then, we can simply use $\hat{X}$ as our data in the linear EC-SVM, (19), with any norm or parameter value and solve using the standard methods that will be discussed in Section 10.2.

9. Interpretation as Robust Optimization, Pessimistic vs. Optimistic Uncertainty

In this section, we present an equivalent representation of the convex and non-convex case of the EC-SVM as robust optimization (RO) problems. This equivalence is enlightening for both the convex and non-convex case. In general, we find that the RO representation reveals a unique interpretation, showing that the convex case corresponds to bPOE minimization with worst-case *pessimistic* data uncertainty, while the non-convex case corresponds to bPOE minimization with best-case *optimistic* data uncertainty. For the convex case, the RO representation is similar to existing equivalences for the C-SVM in Xu et al. (2009), but with a much simpler proof, definition of uncertainty set, and correspondence between equivalent parameter values. For the non-convex case, this new representation reveals a connection with the Total Support Vector Classifier (T-SVC) proposed in Bi and Zhang (2005) and suggests an efficient heuristic for the optimization. Furthermore, the RO representation reveals that this seemingly heuristic optimization method, which was shown to work well in the T-SVC case, is related to DCA, a popular algorithm for locally minimizing DC (difference of convex) functions.

9.1 Convex Case and Pessimism

For the convex case of the EC-SVM, (19), when we have $C \geq 0$, it is simple to show that it is equivalent to (27) and (28). Notice that we are minimizing bPOE, as we did before, but with threshold equal to zero and data subject to *pessimistic* disturbances δ_i. We call this *pessimism*, because we are maximizing w.r.t. the disturbance and looking at the worst case. This is the traditional viewpoint of robust optimization, leading to a special case of the Wald's minimax model.

$$\begin{aligned} \min_{w,b} \quad & \max_{\delta_i} \quad \bar{p}_0\left(-y(w^T(X+\delta)+b)\right) \\ & s.t. \quad \|\delta_i\|^* \leq C \, , i = 1, ..., N \end{aligned} \tag{27}$$

$$\begin{aligned} \min_{w,b} \quad & \max_{\delta_i} \quad \sum_i [-y_i(w^T(X_i+\delta_i)+b)+1]^+ \\ & s.t. \quad \|\delta_i\|^* \leq C, \; i = 1, ..., N \end{aligned} \tag{28}$$

To see the equivalence between (19) and (28), one only needs to observe that for any $w \in \mathbb{R}^n$, if $C \geq 0$, we have that

$$C\|w\| = C \max_{\|\delta\|^* \leq 1} w^T\delta = \max_{\|\delta\|^* \leq C} w^T\delta = \max_{\|\delta\|^* \leq C} -yw^T\delta \ .$$

Furthermore, to see the equivalence between (28) and (27), a simple application of the bPOE formula (9) and multiplication by N transforms (27) into (28).

This provides us with three things. First, we see that the bPOE threshold can be interpreted as controlling the size of the uncertainty set. Following from the transformation of (19) to (28) via the simple algebra, we see that this correspondence is exact, meaning that a threshold of C in the EC-SVM implies an uncertainty set with "size" C in (28) (i.e. optimal solutions are the same when parameters are the same). Second, we see that regularization with norm $\|\cdot\|$ implies uncertainty w.r.t. the corresponding dual norm $\|\cdot\|^*$. Third, we see that the convex case can be interpreted as pessimistic, or risk averse, following the traditional RO framework.

9.2 Nonconvex Case and Optimism

For the non-convex case, we find a similar robust reformulation, but instead of pessimistic disturbances we have *optimistic* disturbances. Specifically, for the EC-SVM (19), when we have $C < 0$, it is simple to show that it is equivalent to (29) and (30). Notice again that we are minimizing bPOE, as we did before, but with threshold equal to zero and data subject to *optimistic* disturbances δ_i. We call this *optimistic*, because we are minimizing w.r.t. the disturbance and looking at the best case.

$$\begin{aligned} \min_{w,b,\delta_i} \quad & \bar{p}_0\left(-y(w^T(X+\delta)+b)\right) \\ s.t. \quad & \|\delta_i\|^* \leq -C \ , i = 1, ..., N \end{aligned} \tag{29}$$

$$\begin{aligned} \min_{w,b,\delta_i} \quad & \sum_i [-y_i(w^T(X_i+\delta_i)+b)+1]^+ \\ s.t. \quad & \|\delta_i\|^* \leq -C \ , i = 1, ..., N \end{aligned} \tag{30}$$

To see the equivalence between (19) and (30), one only needs to observe that for any $w \in \mathbb{R}^n$, if $C < 0$, we have that

$$\begin{aligned} C\|w\| &= C \max_{\|\delta\|^* \leq 1} w^T\delta \\ &= -\max_{\|\delta\|^* \leq 1} -Cw^T\delta \\ &= -\max_{\|\delta\|^* \leq -C} w^T\delta \\ &= \min_{\|\delta\|^* \leq -C} w^T(-\delta) \\ &= \min_{\|\delta\|^* \leq -C} w^T\delta \quad = \min_{\|\delta\|^* \leq -C} -yw^T\delta \ . \end{aligned}$$

Furthermore, to see the equivalence between (30) and (29), a simple application of the bPOE formula (9) and multiplication by N transforms (29) into (30).

This formulation is enlightening for two reasons. First, as already mentioned, we see that the non-convex EC-SVM can be viewed from the RO lens as minimizing bPOE with an optimistic view of the data set. The optimistic perspective makes sense if one assumes, for example, that your data observations have been contaminated by noise. This view is supported by Bi and Zhang (2005), which proposes the Total Support Vector Classifier (T-SVC). Specifically, as revealed by the RO formulation, we see that the T-SVC is equivalent to the non-convex EC-SVM with L_2 norm. In their paper, they show that by using an alternating minimization strategy to solve this problem, a special case of the non-convex EC-SVM performs better than the C-SVM if data have been contaminated by noise. Second, as we will discuss in the next section, the RO formulation suggests that the simple alternating minimization algorithm for finding the local minimum is equivalent to DCA in a certain sense.

10. Practical Considerations

In this section, we consider some practical implications of the theoretical insights connecting soft margin SVM's and bPOE minimization. We first discuss the most obvious use, which is for the selection of the threshold parameter. We show that the new interpretation of the C tradeoff parameter suggests a range that should be used to select the C parameter. This helps to improve grid selection for cross validation so that one can partially avoid solving the EC-SVM for values of C that yield trivial or redundant optimal solutions. Furthermore, using the robust interpretation of the EC-SVM, we show that there may be situations where prior knowledge about data uncertainty suggests, at-best, a fixed C or, at-worst, much tighter bounds for the range of interesting C values for cross-validation.

Another practical consideration is the optimization of the EC-SVM for the convex and non-convex case. For the convex case, we discuss alternative formulations that can be efficiently solved by standard convex optimization tools. With some of these approaches not well suited for large scale problems, we briefly mention recent work on subgradient methods that can be efficiently applied to the EC-SVM. For the non-convex case, we suggest a simple alternating minimization algorithm and show that this procedure has appealing connections with DCA, the Difference of Convex Functions Algorithm; see Tao and An (1997) or Dinh and Le Thi (2014).

Finally, with most learning applications, the true distribution of (X, y) is unknown and thus we must deal with only a finite sample. For example, throughout this paper, we assumed that we had N observations. Thus, in this case, we work with empirical estimates of bPOE, the quantile, the superquantile, and misclassification rate of the true distribution. A practical question, therefore, is regarding generalization. Given an optimal classifier by solving the EC-SVM (or equivalently the C-SVM), what can we say about the true rate of misclassification, $P(-y(w^TX+b) > 0)$, or the true value of bPOE, $\bar{p}_z(-y(w^TX+b))$? With regard to misclassification rate, we utilize results from Takeda and Sugiyama (2008); Schölkopf et al. (2000) to show that generalization bounds can be posed in terms of empirical estimates of bPOE, the quantile, and the superquantile, i.e. the information provided by Property 5. With regard to bPOE, we apply the stability arguments of Bousquet and Elisseeff (2002) to provide tight generalization bounds on the true bPOE at any non-positive threshold and show that the SVM parameter, C, plays an important role in these bounds.

10.1 Parameter Selection

10.1.1 Bounds for the Threshold

With the EC-SVM having a parameter interpretable as the bPOE threshold, we use this to select bounds on the admissible range of the parameter C. From (9), we see that for a random variable Z, the only interesting values of the parameter are $-C = z \in [E[Z], \sup Z]$. Finding this range, or a range containing it, can assist in performing grid search for model selection where we are trying to find the best choice of C without unnecessary iterations over meaningless or redundant parameter values (e.g. different parameter values all yielding bPOE equal to 1). First, we consider the EC-SVM with unbiased linear classifier, where $b = 0$. Then, using this result and some geometric intuition, we are able to formulate a similar bound for biased classifiers, where b is a free variable.

Since the EC-SVM is minimizing bPOE of $\frac{-y(w^T X+b)}{\|w\|}$, we know from (9) that the threshold parameter C achieves varying solutions only for

$$-C \in \left[\inf_{w,b} E\left[\frac{-y(w^T X+b)}{\|w\|}\right], \inf_{w,b}\left\{\sup_{X,y}\left(\frac{-y(w^T X+b)}{\|w\|}\right)\right\}\right] .$$

For the case where $b = 0$, we can use simple algebraic arguments to find data dependent upper and lower bounds for this range. First, we have the lower bound,

$$-C \geq \min_w \frac{1}{N}\sum_i \frac{-y_i(w^T X_i)}{\|w\|} \geq -\max_{i,w} \frac{w^T X_i}{\|w\|} = -\max_i \|X_i\|^* .$$

Next, we find an upper bound for this range.

$$-C \leq \min_w \max_i \frac{-y_i(w^T X_i)}{\|w\|} \leq \max_{i,w} \frac{w^T X_i}{\|w\|} = \max_i \|X_i\|^* .$$

Therefore, when the EC-SVM is formulated with unbiased hyperplane $w^T X$, we only need to do a grid search for $-C \in [-\max_i \|X_i\|^*, \max_i \|X_i\|^*]$. Now, consider the case of the biased hyperplane. To find a lower bound of this range, first note, importantly, that by having a biased hyperplane (i.e. by including the b term), we have that the lower bound can be trivially unbounded with $\inf_{w,b} E\left[\frac{-y(w^T X+b)}{\|w\|}\right] = -\infty$. But this is unreasonable since this hyperplane trivially classifies all points as belonging to the same class. We know that any reasonable hyperplane should at least intersect the norm ball supporting the data set, i.e. the norm ball with radius $\max_i \|X_i\|^*$. Therefore, any non-trivial hyperplane should have an expected margin error larger than the margin error of the best classified point. Because the hyperplane at least intersects the norm ball at one point, the best classified point will have margin error no smaller than $-2\max_i \|X_i\|^*$. Therefore, a reasonable lower bound for $-C$ is,

$$-C \geq -2\max_i \|X_i\|^* .$$

For the upper bound, the exact same logic applies but w.r.t. the worst classified point for any hyperplane intersecting the norm ball containing all data points. Therefore, a reasonable upper bound for $-C$ is,

$$-C \leq 2\max_i \|X_i\|^* .$$

Thus, when the EC-SVM is formulated with biased hyperplane $w^T X + b$, we only need to do a grid search for $-C \in [-2\max_i \|X_i\|^*, 2\max_i \|X_i\|^*]$.

10.1.2 Using the Robust Interpretation

For specific situations, one can actually use the RO representation of the EC-SVM to select the proper parameter value. Since the parameter does not change value, along with the optimal (w, b), when moving from formulation (19) to (28) or (19) to (30), we can use the RO interpretation, exactly, to select parameter values for (19).

A situation where this would be ideal is if each data point X_i is given by some measurement and we know the maximum amount by which this measurement can deviate from the true value. Consider the following two dimensional example, where we use some equipment to measure blood pressure $X^{(1)}$ and weight $X^{(2)}$ where $X = [X^{(1)}, X^{(2)}] \in \mathbb{R}^2$. Assume that we know that the equipment to measure blood pressure is off by at most ± 2 and that the equipment to measure weight is off by at most ± 2. In this case, for any specific measurement X_i, we know that a representative uncertainty set would be $\|\delta_i\|_\infty \leq 2$. This immediately implies that $C = 2$ is desirable if one chooses to be *pessimistic* about the uncertainty or that $C = -2$ is desirable if one chooses to be *optimistic* about the uncertainty. Furthermore, this implies that one should use the EC-SVM formulated with the L_1 norm $\|w\|_1$ since its dual norm is the L_∞ norm.

Furthermore, in less idealistic situations when knowledge of the deviation of measurement is not so exact, the same intuition can be used to, at least, select a range of C values for cross validation that is smaller than the interval $[-2\max_i \|X_i\|^*, 2\max_i \|X_i\|^*]$ provided in the previous section.

10.2 Optimization of the EC-SVM

10.2.1 Convex Case

The primal and dual EC-SVM formulated with any general norm and $C \geq 0$ are convex optimization problems and, thus, there are many options for efficient optimization. Essentially, since from the optimization viewpoint the difference between C-SVM and EC-SVM is minor, the complexity of convex EC-SVM is exactly the same as that of C-SVM. For a general norm, due to convexity, gradient methods may be successfully implemented. For particular norms, we may reformulate the problem as a specific type of convex optimization problem to target available solvers. Below, we detail some problem reformulations into Quadratically Constrained Quadratic Programs (QCQP), Second-Order Cone Programs (SOCP), and Linear Programs (LP). If we assume that the norm is the euclidean norm, the EC-SVM with $C \geq 0$ can be written as the convex QCQP,

$$\begin{aligned} \min_{w,b,\xi,t} \quad & \sum_{i=1}^N \xi_i \\ s.t. \quad & \xi_i \geq -y_i(w^T X_i + b) + Ct + 1, \quad \forall i \in \{1,\dots,N\}. \\ & \xi \geq 0 \\ & t^2 \geq w^T w. \end{aligned} \tag{31}$$

Note that if we simply write $t \geq \|w\|_2$, it is a convex SOCP. If the norm is LP representable (e.g. the L_1 or L_∞ norm), the problem can be reformulated as an LP. For example, with the L_1 norm, the EC-SVM with $C \geq 0$ can be written as,

$$\begin{aligned} \min_{w,\hat{w},b,\xi,t} \quad & \sum_{i=1}^{N} \xi_i \\ s.t. \quad & \xi_i \geq -y_i(w^T X_i + b) + Ct + 1, \quad \forall i \in \{1,\ldots,N\}. \\ & \xi \geq 0 \\ & t \geq \sum_{j=1}^{n} \hat{w}_j \\ & \hat{w}_j \geq w_j \\ & \hat{w}_j \geq -w_j. \end{aligned} \tag{32}$$

Note that the convex dual EC-SVM can also be written as a QCQP, SOCP, or LP in a similar manner depending on the choice of norm.

For large-scale problems, though, these reformulations may suffer. For the C-SVM, this issue has garnered much attention. Although we leave in-depth exploration to future work, we note that many techniques used to solve large scale SVM's are directly applicable to the convex EC-SVM. Note that the primal EC-SVM with $C \geq 0$ can be formulated as the following convex, non-smooth optimization problem.

$$\min_{w,b} \sum_{i=1}^{N} [-y_i(w^T X_i + b) + C\|w\| + 1]^+ \tag{33}$$

This formulation lends itself to subgradient methods, particularly stochastic variants which have been shown to work well for large-scale SVMs. In fact, methods mirroring Pegasos Shalev-Shwartz et al. (2011), a solver using a subgradient method which finds an ϵ-accurate solution, with high probability, for the primal C-SVM in runtime $O(\frac{1}{C\epsilon})$, have been successfully applied to large-scale robust SVM's taking a similar form to (33) in Wang et al. (2016).[4] Note, also, that the EC-SVM can be formulated as a decomposable consensus problem which can be solved in a distributed way via the Alternating Direction Method of Multipliers; see Chapter 7 and 8 of Boyd et al. (2011).

10.2.2 Non-Convex Case

For the EC-SVM with $C < 0$, solving for a global optimum is much more difficult. We can, though, solve for a local minimum by using the optimistic RO formulation (30) and implementing the simple, seemingly heuristic, approach of alternating minimization. While this approach was shown to be effective for the T-SVC special case of the EC-SVM presented in Bi and Zhang (2005), we show here that its effectiveness may be explained, on the surface at least, by its connection with DCA. Specifically, we have the following alternating algorithm, where at each step one solves an LP followed by a convex optimization problem:

4. Ignoring some details for simplicity, the robust SVM of similar form is formulated as $\min_{w,b} \sum_i [-y_i(w^T X_i + b) + C\|\Sigma^{\frac{1}{2}} w\| + 1]^+$, where Σ is a particular covariance matrix.

- **Initialize:** Minimize (30) w.r.t. (w, b), fixing $\delta_i = 0$ for all $i = 1, ..., N$. Denote optimal point as (w^*, b^*).
- **Step 1:** Let $\hat{\delta}_i = \underset{\|\delta_i\|^* \leq -C}{\operatorname{argmax}} w^{*T}\delta_i$ for all $i = 1, ..., N$.
- **Step 2:** Minimize (30) w.r.t. (w, b), fixing $\delta_i = \hat{\delta}_i$ for all $i = 1, ..., N$. Denote the optimal point as (w^*, b^*).
- **Step 3:** If iteration limit reached or convergence criteria is satisfied, STOP. Else, return to Step 1.

This algorithm, at worst, seems like a simple heuristic approach to a non-convex problem with intuitive appeal. In Step 1, we start with the data $X+\delta$, which is an optimistic version of the original data X if $\delta \neq 0$. We then find a classifier (w, b) that minimizes bPOE at threshold zero on the optimistic version of the data set. Then, in Step 2, we look for a new, optimistic view of the data that is optimistic w.r.t. the new classifier found in Step 1.

In addition to its intuitive appeal, however, this alternating minimization has theoretical justification. We can show that this alternating minimization algorithm is essentially implementing DCA, a commonly used algorithm for solving DC optimization problems. In short, a DC function h, is such that $h(v) = h_1(v) - h_2(v)$ where h_1, h_2 are both convex. To find a local minimum for the problem $\min_v h(v)$, DCA performs the following steps[5] :

- **Initialize:** $\hat{v} = \underset{v}{\operatorname{argmin}}\ h_1(v)$.
- **Step 1:** Set δ equal to a subgradient of h_2 at $\hat{v}$: $\delta \in \partial h_2(\hat{v})$.
- **Step 2:** Set $\hat{v} = \underset{v}{\operatorname{argmin}}\ h_1(v) - v^T\delta$.
- **Step 3:** If iteration limit reached or convergence criteria is satisfied, STOP. Else, return to Step 1.

Thus, DCA can be seen as first linearizing the concave term at the current solution via a subgradient and then solving a convex subproblem which replaces the concave term with the linear approximation.

To see how the alternating minimization procedure is related to DCA, first notice that when $C < 0$, the function $-y_i(w^T X_i + b) + C\|w\| + 1$ is DC; the sum of the convex function $-y_i(w^T X_i + b) + 1$ and the concave function $C\|w\|$. Therefore, $\sum_i [-y_i(w^T X_i + b) + C\|w\| + 1]^+$ is also DC.[6] Second, notice that $-C\|w\|$ can be viewed as the support function of the convex set $S = \{\delta | \|\delta\|^* \leq -C\}$, meaning that $-C\|w\| = \max_{\delta \in S} w^T\delta$. By the properties of support functions[7] we then know that if $\delta \in \underset{\|\delta\|^* \leq -C}{\operatorname{argmax}} w^T\delta$, then $\delta \in \partial(-C\|w\|)$.

Therefore, we see that we initialize the alternating minimization algorithm by setting the concave part to zero and minimizing the convex part. Then, in Step 1, because of the

5. Note that for brevity, we present a very simplified view of DCA and ignore the underlying duality results which produce this algorithm.
6. This follows from the property that the maximum of DC functions is DC and that the sum of DC functions is DC.
7. See e.g. Chapter 13 of Rockafellar (2015).

properties of the support function, we are equivalently solving for $\hat{\delta}_i \in \partial(-C\|w\|)$. Finally, in Step 2, we linearize the concave part using this subgradient and solve the resulting convex problem. Thus, not only is the alternating minimization algorithm simple, but it is also theoretically appealing with its relation to DCA.

10.3 Generalization of POE and bPOE for SVM Classifiers

In machine learning, particularly in the context of empirical risk minimization, statistical learning theory is a popular tool for analyzing the predictive ability of a learning algorithm $A : S \to \mathcal{F}$ which maps a sample S onto a function $A_S \in \mathcal{F}$, with $\mathcal{F}$ being some class of functions such that the score $A_S(X) \in \mathbb{R}$ is used to predict y given X. For example, in classification, we can consider the prediction of y given X to be $sign\{A_S(X)\}$. Statistical learning theory addresses the fact that the true distribution of (X, y) is usually unknown and we are only given access to a finite sample $S = \{(X_1, y_1), ..., (X_N, y_N)\}$. Thus, if we have, for example, a real valued random loss $V(A_S, (X, y))$ for the output A_S of the learning algorithm trained on S, we need to estimate the true risk

$$\mathcal{R}_{true}(A, S) = E_{X,y}[V(A_S, (X, y))] \,,$$

given only the empirical estimate,

$$\mathcal{R}_{emp}(A, S) = \frac{1}{N} \sum_i V(A_S, (X_i, y_i)) \,.$$

The risk can be viewed as a type of performance metric we would like to minimize, like bPOE or misclassification rate. Given a training algorithm and a training set S, statistical learning theory helps to determine the expected value of the difference $|\mathcal{R}_{true}(A, S) - \mathcal{R}_{emp}(A, S)|$.

In the context of binary classification, we consider two different types of risk (or performance metrics) for the EC-SVM (or equivalently C-SVM) learning algorithm. We first consider the true misclassification rate, which is the typical object of study for classification algorithms in statistical learning theory. Drawing on results from Takeda and Sugiyama (2008), we show that, given a solution to the EC-SVM, the true misclassification rate can be upper bounded by a function of the empirical estimates of bPOE, the superquantile, and the quantile of the random loss $\left(\frac{-y(w^T X+b)}{\|w\|}\right)$. Furthermore, given Property 5, we find that we can state these bounds in terms of the value of the EC-SVM objective function, C, and $\frac{1}{\|w\|}$.

Next, instead of misclassification rate, we consider generalization bounds for the true bPOE of the random loss $\left(\frac{-yw^T X}{\|w\|}\right)$. Using the stability arguments of Bousquet and Elisseeff (2002), given a solution to the EC-SVM or C-SVM, we provide tight upper bounds for the true bPOE at any non-positive threshold. In general, for binary classification, we find that using these stability arguments to bound bPOE is more intuitive than the use of the clipped loss function from Bousquet and Elisseeff (2002). Note that we only consider the unbiased case with $b = 0$, as it is unclear whether this bound holds for the biased case. Furthermore, in Section 10.3.2 we consider only the L_2 norm and in Section 10.3.3 we only consider norms possessing a variant of strong convexity.

In the following sections, given a sample $S = \{(X_1, y_1), ..., (X_N, y_N)\}$, we consider $L(w, b, S)$ to be a discretely distributed random loss with equally probable realizations $\{-y_1(w^T X_1 + b), ..., -y_N(w^T X_N + b)\}$. Therefore, we have that

$$\bar{p}_{-C}\left(\frac{L(w,b,S)}{\|w\|}\right), \bar{q}_\alpha\left(\frac{L(w,b,S)}{\|w\|}\right), \text{ and } q_\alpha\left(\frac{L(w,b,S)}{\|w\|}\right)$$

are empirical estimates of the *true* bPOE, superquantile, and quantile

$$\bar{p}_{-C}\left(\frac{-y(w^T X+b)}{\|w\|}\right), \bar{q}_\alpha\left(\frac{-y(w^T X+b)}{\|w\|}\right), q_\alpha\left(\frac{-y(w^T X+b)}{\|w\|}\right) \,.$$

10.3.1 Generalization of POE

Takeda and Sugiyama (2008) provide bounds on the true misclassification rate of the Eν-SVM classifier. Given the equivalence between the Eν-SVM and EC-SVM provided in Section 7, we can apply these results directly to the EC-SVM. Although these results are pulled directly from Takeda and Sugiyama (2008), hence we do not provide the proofs, we find that we can pose their bounds on the true misclassification rate strictly in terms of the empirical estimates of bPOE, the superquantile, and the quantile of the random loss $\left(\frac{-y(w^T X+b)}{\|w\|_2}\right)$. Furthermore, given Property 5, we find that we can state these bounds in terms of the value of the EC-SVM objective function, C, and $\frac{1}{\|w\|_2}$. Note also that we are able to provide bounds for the non-convex case where $C < 0$.

For the following theorems, let $P_{true} = P\left(-y(w^T X + b) > 0\right)$, i.e. the true misclassification rate for the classifier (w, b), and let

$$G(\tau) = \sqrt{\frac{2}{N}\left(\frac{4k^2(R^2+1)^2}{\tau^2}\log_2(2N) - 1 + \ln\frac{2}{\rho}\right)} \,.$$

Theorem 3 *Suppose that (w, b) is an optimal solution to the EC-SVM trained on sample S, that the optimal objective value divided by sample size N equals $1 - \alpha$, and that either $C > 0$ or $C < 0$ and $-C - \frac{1}{\|w\|_2} < 0$. If the support $\mathcal{X}$ of X is contained in a ball of radius R, then there exists a positive constant k such that the following bound holds with probability at least $1 - \rho$:*

$$P_{true} \le \bar{p}_{-C}\left(\frac{L(w,b,S)}{\|w\|_2}\right) + G\left(q_\alpha\left(\frac{L(w,b,S)}{\|w\|_2}\right)\right) \,.$$

Furthermore, given Property 5, this can be written simply as:

$$P_{true} \le 1 - \alpha + G\left(-C - \frac{1}{\|w\|_2}\right) \,.$$

Theorem 4 *Make the same assumptions as in Theorem 3, but suppose that $C < 0$ and $-C - \frac{1}{\|w\|_2} > 0$. Then there exists a positive constant k such that the following bound holds with probability at least $1 - \rho$:*

$$P_{true} \ge \bar{p}_{-C}\left(\frac{L(w,b,S)}{\|w\|_2}\right) - G\left(q_\alpha\left(\frac{L(w,b,S)}{\|w\|_2}\right)\right) \,.$$

Furthermore, given Property 5, this can be written simply as:

$$P_{true} \geq 1 - \alpha - G\left(-C - \frac{1}{\|w\|_2}\right) .$$

Given these theorems, we see that the EC-SVM is directly minimizing the the lower (or upper) bound on P_{true}, while simultaneously trading off w.r.t. the choice in parameter C.

10.3.2 Generalization of Buffered Probability

Here, we use the stability arguments of Bousquet and Elisseeff (2002) to provide generalization bounds on bPOE for optimal solutions of the C-SVM and convex EC-SVM. We provide tight bounds for bPOE of the normalized loss distribution of the classifier given by an SVM trained on a finite sample. This shows that while the C-SVM and EC-SVM directly minimize bPOE, this statistic does, in fact, generalize to unseen samples.

For brevity, we avoid a full introduction of uniform stability and the associated generalization theorems and refer readers to Bousquet and Elisseeff (2002). Thus, we confine most details to the proof. Theorem 5 considers the C-SVM, but with squared norm, and shows that we can provide tight bounds on the true bPOE of the loss $\left(\frac{-yw^TX}{\|w\|}\right)$ at any non-positive threshold. We present the C-SVM with squared norm in the main theorem, as opposed to the EC-SVM, because the bound expression is much more straightforward. Corollary 1 shows how this result applies to the case of the EC-SVM with L_2 norm.

In general, both bound expressions provide the same intuitive result. The uncertainty regarding the estimate of bPOE will grow as the bPOE threshold considered in the SVM minimization shrinks. As the SVM parameter grows larger, the threshold for the equivalent bPOE minimization problem decreases. Thus, we are considering a growing portion of the tail of the loss distribution in our minimization. This means that the SVM objective function will get larger, but the uncertainty will grow smaller since a larger portion of the tail will be easier to estimate than a smaller portion (it contains more samples). The opposite holds true as the SVM parameter shrinks, getting closer to zero. This implies that we are considering a shrinking portion of the tail of the loss distribution in our minimization. Thus, the SVM objective function will grow smaller, but the uncertainty will grow larger since a smaller portion of the tail will be more difficult to empirically estimate (it contains less samples).

We note that we assume that the classifier is unbiased with $b = 0$ and that the squared norm is differentiable and m-strongly convex w.r.t. itself.[8] For example, the squared L_2 norm is strongly convex w.r.t. itself with $m = 2$. A bias term can be included by concatenating a feature equal to one onto the data, but this would then be included in the regularization norm. It is unclear whether these bounds hold for the truly unbiased case. Although we do not present it, we also point out that this can easily be generalized to consider functions in a Reproducing Kernel Hilbert Space such that the kernel function $K(\cdot,\cdot)$ is bounded with $\sup_{X\in\mathcal{X}} K(X,X) \leq R$, as is done in Bousquet and Elisseeff (2002). For the following proposition, recall that $\bar{p}_{-C}\left(\frac{-yw^TX}{\|w\|}\right)$ denotes the true bPOE while $\bar{p}_{-C}\left(\frac{L(w,S)}{\|w\|}\right)$ denotes empirical bPOE.

8. A differentiable function f is m-strongly convex w.r.t. the norm $\|\cdot\|$ if for any w_1, w_2 we have that $\frac{m}{2}\|w_1 - w_2\|^2 \leq f(w_1) - f(w_2) - \langle \nabla f(w_2), w_1 - w_2\rangle$.

Theorem 5 *Assume that the squared norm $\|\cdot\|^2$ is differentiable and m-strongly convex w.r.t. itself and that $\sup_{x\in\mathcal{X}}\|x\|^* \leq R$, meaning that the support $\mathcal{X}$ of X is contained in a $\|\cdot\|^*$-norm ball of radius R. Assume we solve the C-SVM with squared norm on training set $S = \{(X_1, y_1), ..., (X_N, y_N)\}$, yielding*

$$w_S \in \underset{w}{argmin}\ \frac{1}{N}\sum_i [-y_i w^T X_i + 1]^+ + \lambda\|w\|^2 ,$$

where $\lambda > 0$. Then for any $a, C \geq 0$, with probability $1-\rho$,

$$\bar{p}_{-C}\left(\frac{-yw_S^TX}{\|w_S\|}\right) \leq \frac{1}{N}\sum_i [-y_i a w_S^T X_i + C\|aw_S\| + 1]^+ + \frac{U_{\frac{1}{2}}(a,\lambda,C)\sqrt{\ln\frac{1}{\rho}}}{\sqrt{N}} + \frac{U_1(a,\lambda,C)}{N} ,$$

where $U_{\frac{1}{2}}(a,\lambda,C) = \frac{1}{\sqrt{2}} + \frac{a(R+C)}{\sqrt{2\lambda}}\left(1 + \frac{4R}{m\sqrt{\lambda}}\right)$, $U_1(a,\lambda,C) = \frac{2aR(R+C)}{m\lambda}$. In particular,

$$\bar{p}_{-C}\left(\frac{-yw_S^TX}{\|w_S\|}\right) \leq \bar{p}_{-C}\left(\frac{L(w_S,S)}{\|w_S\|}\right) + \frac{U_{\frac{1}{2}}(\bar{a}_S,\lambda,C)\sqrt{\ln\frac{1}{\rho}}}{\sqrt{N}} + \frac{U_1(\bar{a}_S,\lambda,C)}{N},$$

with probability $1-\rho$, when $\bar{a}_S \in \underset{a\geq 0}{argmin}\ \frac{1}{N}\sum_i [-y_i a w_S^T X_i + C\|aw_S\| + 1]^+$.

Proof Let $V(w,(X,y),a,C) = [-yaw^TX + C\|aw\| + 1]^+$. First, note that, for any $C \geq 0$,

$$\begin{aligned}
\bar{p}_{-C}(\frac{-yw_S^TX}{\|w_S\|}) &= \min_{a\geq 0} E[a\left(\frac{-yw_S^TX}{\|w_S\|} + C\right) + 1]^+ \\
&= \min_{a\geq 0} E[\frac{a}{\|w_S\|}\left(-yw_S^TX + C\|w_S\|\right) + 1]^+ \\
&= \min_{a\geq 0} E[-yaw_S^TX + C\|aw_S\| + 1]^+ \\
&\leq E[V(w_S,(X,y),a,C)] \ , \forall\ a \geq 0 .
\end{aligned}$$

Given any $a \geq 0$, we can upper bound $E[V(w_S,(X,y),a,C)]$ by using stability arguments of Bousquet and Elisseeff (2002).

We first prove that the C-SVM with squared norm is uniformly stable, as defined in Bousquet and Elisseeff (2002), w.r.t. loss function $V(w,(X,y),a,C)$, showing that there exists θ such that for any sample S,

$$\sup_{X,y}|V(w_S,(X,y),a,C) - V(w_{S^j},(X,y),a,C)| \leq \theta$$

where

$$w_S \in \operatorname*{argmin}_w \frac{1}{N}\sum_i [-y_i w^T X_i + 1]^+ + \lambda\|w\|^2$$

$$w_{S^j} \in \operatorname*{argmin}_w \frac{1}{N}\sum_{i\neq j} [-y_i w^T X_i + 1]^+ + \lambda\|w\|^2 .$$

First, note that for any $w_1, w_2 \in \mathbb{R}^n$, $X, y \in \mathcal{X} \times \{-1, 1\}$ we have that

$$\begin{aligned}
|V(w_1,(X,y),a,C) - V(w_2,(X,y),a,C)| &= |[-y(aw_1^TX) + C\|aw_1\| + 1]^+ \\
&\quad - [-y(aw_2^TX) + C\|aw_2\| + 1]^+| \\
&\le |-y(aw_1^TX) + C\|aw_1\| + 1 \\
&\quad - \left(-y(aw_2^TX) + C\|aw_2\| + 1\right)| \\
&= \left|-ya(w_1-w_2)^TX + Ca(\|w_1\| - \|w_2\|)\right| \\
&\le \left|-ya(w_1-w_2)^TX\right| + Ca\,|\|w_1\| - \|w_2\|| \\
&\le \left|-ya(w_1-w_2)^TX\right| + Ca\|w_1 - w_2\| \\
&\le a\left|(w_1-w_2)^TX\right| + Ca\|w_1 - w_2\| \\
&\le \sup_{X\in\mathcal{X}} a\left|(w_1-w_2)^TX\right| + Ca\|w_1 - w_2\| \\
&\le \sup_{\|X\|^*\le R, X\in\mathbb{R}^n} a\left|(w_1-w_2)^TX\right| + Ca\|w_1 - w_2\| \\
&= Ra\|w_1 - w_2\| + Ca\|w_1 - w_2\| \\
&= a(R+C)\|w_1 - w_2\|
\end{aligned}$$

We show now that $\|w_S - w_{S^j}\| \le \frac{R}{mN\lambda}$. Let $d_f, \bar{d}_f$ denote the Bregman divergence and the generalized Bregman divergence of a convex function f with subderivatives at any w denoted as $\nabla f(w) \in \partial f(w)$. (See Appendix F for definition and properties of divergence). Let $g(w) = \|w\|^2$, $h_S(w) = \frac{1}{N}\sum_i [-y_i w^T X_i + 1]^+$, and $h_{S^j}(w) = \frac{1}{N}\sum_{i\neq j}[-y_i w^T X_i + 1]^+$. First, since $g(w)$ is assumed to be m-strongly convex w.r.t. itself, by definition we have $\frac{m}{2}\|w_S - w_{S^j}\|^2 \le d_g(w_S, w_{S^j})$. Next, we have that

$$\begin{aligned}
\lambda m\|w_S - w_{S^j}\|^2 &\le \lambda\left(d_g(w_{S^j}, w_S) + d_g(w_S, w_{S^j})\right) \\
&= d_{\lambda g}(w_{S^j}, w_S) + d_{\lambda g}(w_S, w_{S^j}) \\
\text{(following from (44))} &= \bar{d}_{\lambda g}(w_{S^j}, \nabla\lambda g(w_S)) + \bar{d}_{\lambda g}(w_S, \nabla\lambda g(w_{S^j}))
\end{aligned}$$

Since generalized divergence is linear and non-negative (see Appendix F and (45)), we have for any $\nabla(h_S(w_S) + \lambda g(w_S)) \in \partial(h_S(w_S) + \lambda g(w_S))$ and any $\nabla(h_{S^j}(w_{S^j}) + \lambda g(w_{S^j})) \in \partial(h_{S^j}(w_{S^j}) + \lambda g(w_{S^j}))$, that

$$\begin{aligned}
\bar{d}_{\lambda g}(w_{S^j}, \nabla\lambda g(w_S)) + \bar{d}_{\lambda g}(w_S, \nabla\lambda g(w_{S^j})) &\le \bar{d}_{(h_S+\lambda g)}(w_{S^j}, \nabla(h_S(w_S) + \lambda g(w_S))) \\
&\quad + \bar{d}_{(h_{S^j}+\lambda g)}(w_S, \nabla(h_{S^j}(w_{S^j}) + \lambda g(w_{S^j}))) .
\end{aligned}$$

Now, since $0 \in \partial(h_S(w_S) + \lambda g(w_S))$ and $0 \in \partial(h_{S^j}(w_{S^j}) + \lambda g(w_{S^j}))$, we have

$$\begin{aligned}
\bar{d}_{\lambda g}(w_{S^j}, \nabla\lambda g(w_S)) + \bar{d}_{\lambda g}(w_S, \nabla\lambda g(w_{S^j})) &\le \bar{d}_{(h_S+\lambda g)}(w_{S^j}, 0) + \bar{d}_{(h_{S^j}+\lambda g)}(w_S, 0) \\
\text{(following from (46))} &= h_S(w_{S^j}) + \lambda g(w_{S^j}) - h_S(w_S) - \lambda g(w_S) \\
&\quad + h_{S^j}(w_S) + \lambda g(w_S) - h_{S^j}(w_{S^j}) - \lambda g(w_{S^j}) \\
&= h_S(w_{S^j}) - h_S(w_S) + h_{S^j}(w_S) - h_{S^j}(w_{S^j}) \\
&= \frac{1}{N}\left(V(w_{S^j},(X_j,y_j),1,0) - V(w_S,(X_j,y_j),1,0)\right) \\
&\le \frac{R\|w_{S^j} - w_S\|}{N}
\end{aligned}$$

This then implies that $\|w_S - w_{S^j}\|^2 \le \|w_S - w_{S^j}\|\frac{R}{mN\lambda}$ which further implies that $\|w_S - w_{S^j}\| \le \frac{R}{N\lambda m}$. Together with the previous result, we have that

$$|V(w_S,(X,y),a,C) - V(w_{S^j},(X,y),a,C)| \le a(R+C)\|w_S - w_{S^j}\| \le \frac{aR(R+C)}{N\lambda m} .$$

Thus, the learning algorithm is uniformly θ-stable w.r.t. loss function V with $\theta = \frac{aR(R+C)}{N\lambda m}$.

To apply Theorem 12 of Bousquet and Elisseeff (2002), which gives us the final generalization bound, we need to show that $V(w_S,(X,y),a,C)$ is bounded. To see this, first notice that

$$\lambda\|w_S\|^2 \le h_S(w_S) + \lambda\|w_S\|^2 \le h_S(0) + \lambda\|0\|^2 = \frac{1}{N}\sum_i V(0,(X_i,y_i),1,0) = 1 .$$

Second, combining this with the first result of the proof yields,

$$\begin{aligned}
V(w_S,(X,y),a,C) &\le |V(w_S,(X,y),a,C) - V(0,(X,y),a,C)| + V(0,(X,y),a,C) \\
&\le a(R+C)\|w_S - 0\| + 1 \\
&\le \frac{a(R+C)}{\sqrt{\lambda}} + 1 .
\end{aligned}$$

Thus, applying Theorem 12 of Bousquet and Elisseeff (2002), for any $a \ge 0$ yields the first generalization bound. The second generalization bound follows from the fact that if

$$\bar{a} \in \operatorname*{argmin}_{a\ge 0} \frac{1}{N}\sum_i [-y_i a w_S^T X_i + C\|aw_S\| + 1]^+ ,$$

then,

$$\frac{1}{N}\sum_i [-y_i \bar{a} w_S^T X_i + C\|\bar{a}w_S\| + 1]^+ = \bar{p}_{-C}\left(\frac{L(w_S,S)}{\|w_S\|}\right) .$$

■

Corollary 1 *Make the same assumptions as in Theorem 5, but solve the EC-SVM with* $\|\cdot\| = \|\cdot\|_2$ *and* $\hat{C} > 0$*, which yields,*

$$w_S \in \operatorname*{argmin}_w \frac{1}{N}\sum_i [-y_i(w^TX_i) + 1 + \hat{C}\|w\|_2]^+ .$$

Then, for any $a, C \ge 0$*, with probability* $1-\rho$*,*

$$\bar{p}_{-C}\left(\frac{-yw_S^TX}{\|w_S\|}\right) \le \frac{1}{N}\sum_i [-y_i a\mu w_S^T X_i + C\|a\mu w_S\| + 1]^+ + \frac{U_{\frac{1}{2}}(a,\lambda_S,C)\sqrt{\ln\frac{1}{\rho}}}{\sqrt{N}} + \frac{U_1(a,\lambda_S,C)}{N},$$

where $\lambda_S = \frac{\hat{C}\sum_i \beta_i}{2\|w_S\|_2}$, $\mu = \frac{1}{1+\hat{C}\|w_S\|_2}$, *and* β_i *are the optimal dual variables of the EC-SVM. In particular, when* $\bar{a}_S \in \underset{a\geq 0}{argmin}\ \frac{1}{N}\sum_i [-y_i a\mu w_S^T X_i + C\|a\mu w_S\| + 1]^+$, *with probability* $1-\rho$,

$$\bar{p}_{-C}\left(\frac{-yw_S^T X}{\|w_S\|}\right) \leq \bar{p}_{-C}\left(\frac{L(w_S,S)}{\|w_S\|}\right) + \frac{U_{\frac{1}{2}}(\bar{a}_S,\lambda_S,C)\sqrt{\ln\frac{1}{\rho}}}{\sqrt{N}} + \frac{U_1(\bar{a}_S,\lambda_S,C)}{N}.$$

Proof From Theorem 2, we know that if w is optimal for the EC-SVM with $\hat{C} \geq 0$, then solving the EC-SVM is equivalent to solving the C-SVM with parameter $\hat{C}\sum_i \beta$ and will yield optimal solution μw_S. Furthermore, it is easy to see by looking at the KKT system (34) that if w is optimal for the C-SVM with parameter $\hat{C} \geq 0$, then solving the C-SVM is equivalent to solving the C-SVM with squared norm with parameter $\frac{\hat{C}}{2\|w\|_2}$ and will yield optimal solution w. Therefore, if w_S is optimal for the EC-SVM with $\hat{C} \geq 0$, then solving the EC-SVM is equivalent to solving the C-SVM with squared norm with parameter $\frac{\hat{C}\sum_i \beta_i}{2\|w_S\|_2}$ and will yield optimal solution μw. Thus, we can apply Theorem 5 with $\lambda = \frac{\hat{C}\sum_i \beta_i}{2\|w_S\|_2}$ and μw_S. ■

11. Conclusion

In this paper we have introduced a new SVM formulation called the EC-SVM to help provide theoretical insights into the nature of the C-SVM, soft margin support vector classifier. Much like the Eν-SVM, this new formulation acts as an extension of the C-SVM. The main contribution of this paper, though, is not a new SVM formulation with computational or generalization benefits.

The main contribution of this paper is proof that soft margin support vector classification is equivalent to simple bPOE minimization. Additionally, we show that the C-SVM, EC-SVM, ν-SVM, and Eν-SVM fit nicely into the general framework of superquantile and bPOE minimization problems. This allows us to gain interesting and surprising insights, interpreting soft margin support vector optimization with newly developed statistical tools. For example, we were able to show that the C-parameter of the C-SVM has a statistical interpretation and that the optimal objective value, divided by sample size, equals a probability level.

Additionally, we were able to provide two useful interpretations of the EC-SVM. First, we showed that it can be considered to be a hard-margin optimization problem, showing that the choice of regularization norm implies a metric used to define the margin. Second, we showed that the convex and non-convex case can be interpreted as Robust Optimization problems, with convexity implying pessimistic views of data uncertainty and non-convexity implying optimistic views of data uncertainty.

We also address some practical implications of these theoretical insights. We show that the new interpretations imply that the interesting values of C lie in a limited range. We also suggest methods for solving the convex and non-convex case of the EC-SVM, using the robust interpretation in particular for solving the non-convex case in an efficient and theoretically justifiable way. Finally, we also show that we can provide generalization bounds

for both misclassification rate and bPOE. We provide tight bounds for bPOE of the loss distribution of the classifier given by the C-SVM and EC-SVM trained on a finite sample. This shows that while the C-SVM and EC-SVM minimize bPOE, this statistic does, in fact, generalize to unseen samples.

In the broad scheme, we were able to show that the C-SVM formulation, derived traditionally from geometric intuition, can also be derived from purely statistical tools, with little geometric intuition involved. Specifically, we show that these statistical tools are superquantiles and the related bPOE.

Acknowledgment

Authors would like to thank Dr. Akiko Takeda, Dr. Junya Gotoh, Dr. Anand Rangarajan, and Prof. R. Tyrrell Rockafellar, for the productive discussions and valuable comments.

This work was partially supported by the USA AFOSR grants: "Design and Redesign of Engineering Systems", FA9550-12-1-0427, and "New Developments in Uncertainty: Linking Risk Management, Reliability, Statistics and Stochastic Optimization", FA9550-11-1-0258. This material is based upon work supported by the Air Force Research Laboratory (AFRL) Mathematical Modeling and Optimization Institute.

Appendix A. Ineffective Variation of Threshold Levels

In application, it may be desirable that bPOE is minimized for different thresholds $z \in \mathbb{R}$ yielding a selection of optimal distributions $f(w_z^*, X)$, where $w_z^* = \arg\min \bar{p}_z(f(w,X))$ for some chosen value of threshold z. This way, one could do some type of model selection or analysis based upon the behavior of the optimal distribution over different thresholds. In doing so, one would expect to achieve different solutions for different threshold choices. As shown in the following propositions, the naive construction of formulation (12) combined with the positive homogeneity of $f(w,X)$ causes formulation (12) to achieve only two possible optimal solutions.

Proposition 4 and Corollary 2 show that for any threshold $z \leq 0$, formulation (12) becomes equivalent to

$$\min_{w\in\mathbb{R}^n} \quad \bar{p}_0(f(w,X)),$$

effectively yielding the solution for threshold $z = 0$. Proposition 5 shows that for any threshold $z > 0$, formulation (12) yields a trivial solution.

Proposition 4 *If* $f(w,X)$ *is PH w.r.t.* w *and minimizing bPOE at* $z \leq 0$ *yields*

$$1-\alpha^* = \min_{w,\gamma<z} \quad \frac{E[f(w,X)-\gamma]^+}{z-\gamma},$$

with optimal solution vector (w^*,γ^*), *then for any* $a \geq 1$, $\bar{z} = az$ *we have*

$$1-\alpha^* = \min_{w,\gamma<\bar{z}} \quad \frac{E[f(w,X)-\gamma]^+}{\bar{z}-\gamma},$$

with optimal solution vector $(aw^*, a\gamma^*)$.

Proof Assume that for $z \leq 0$,

$$1 - \alpha^* = \min_{w \in \mathbb{R}^n} \quad \bar{p}_z(f(w, X)) = \min_{w \in \mathbb{R}^n, \gamma < z} \quad \frac{E[f(w, X) - \gamma]^+}{z - \gamma} = \frac{E[f(w^*, X) - \gamma^*]^+}{z - \gamma^*}.$$

This means that $\bar{p}_z(f(w^*, X)) \leq \bar{p}_z(f(w, X))$ for every $w \in \mathbb{R}^n$. Now, notice that for $\bar{z} = az$, where $a \geq 1$,

$$\begin{aligned}
\bar{p}_{\bar{z}}(f(aw^*, X)) &= \frac{E[f(aw^*, X) - a\gamma^*]^+}{\bar{z} - a\gamma^*} \\
&= \frac{a}{a} \frac{E[f(w^*, X) - \gamma^*]^+}{z - \gamma^*} \\
&= 1 - \alpha^* \\
&= \bar{p}_z(f(w^*, X)).
\end{aligned}$$

Since $\bar{p}_z(f(w, X))$ is a monotonically decreasing function w.r.t. z, we also know that $\bar{p}_z(f(w^*, X)) \leq \bar{p}_{\bar{z}}(f(w^*, X))$ for every $\bar{z} = az$, $a \geq 1$. Therefore, if $\min_{w \in \mathbb{R}^n} \bar{p}_z(f(w, X)) = 1 - \alpha^*$ at (w^*, γ^*), then for any $\bar{z} = az$, $a \geq 1$ we have that $\min_{w \in \mathbb{R}^n} \bar{p}_{\bar{z}}(f(w, X)) = 1 - \alpha^*$ at $(aw^*, a\gamma^*)$. ■

Corollary 2 *Given Proposition 4, we can say that if $z \leq 0$, then*

$$\min_{w \in \mathbb{R}^n, \gamma < z} \quad \frac{E[f(w, X) - \gamma]^+}{z - \gamma} = \min_{w \in \mathbb{R}^n} \quad \bar{p}_0(f(w, X)).$$

Proof Let (z_n) be a strictly decreasing sequence such that $z_0 < 0$ and $\lim_{n \to \infty} z_n = 0$. Proposition 2 implies that

$$\min_{w \in \mathbb{R}^n} \bar{p}_{z_0}(f(w, X)) = \min_{w \in \mathbb{R}^n} \bar{p}_{z_1}(f(w, X)) = ... = \min_{w \in \mathbb{R}^n} \bar{p}_0(f(w, X)).$$

Intuitively, this corollary simply follows from application of Proposition 2 to any $z < 0$ arbitrarily close to zero. ■

Proposition 5 *If $z > 0$, then*

$$\min_{w \in \mathbb{R}^n, \gamma < z} \quad \frac{E[f(w, X) - \gamma]^+}{z - \gamma} = 0.$$

Proof Objective is a non-negative function. If $z > 0$, then for $\gamma \in (0, z)$ the objective is 0, hence, optimal. ■

Appendix B. Proofs for Theorems 1 and 2

B.1 Theorem 1

Proof To prove Theorem 1, we compare the KKT systems of (1) and (19) formulated with the same general norm, $\|\cdot\|$. We assume that the C-SVM with parameter $\hat{C} \geq 0$ yields optimal primal variables (w^*, b^*, ξ^*) and optimal dual variables (α^*, β^*). Thus, this is the same as assuming that $(w^*, b^*, \xi^*, \alpha^*, \beta^*)$ satisfies the following KKT system of (1):

$$\xi^* \geq 0, \tag{34a}$$
$$\beta^* \geq 0, \tag{34b}$$
$$\alpha^* \geq 0, \tag{34c}$$
$$-\alpha^* - \beta^* + 1 = 0, \tag{34d}$$
$$\xi_i^* \geq -y_i({w^*}^T x_i + b^*) + 1, \tag{34e}$$
$$0 \in -\hat{C}\partial\|w^*\| + \sum_{i=1}^N y_i \beta_i^* x_i, \tag{34f}$$
$$\sum_{i=1}^N -y_i \beta_i^* = 0, \tag{34g}$$
$$\alpha_i^* \xi_i^* = 0 = (1 - \beta_i^*)\xi_i^*, \tag{34h}$$
$$\beta_i^* \left[-y_i({w^*}^T x_i + b^*) + 1 - \xi_i^*\right] = 0. \tag{34i}$$

Now, we show that with $\mu = \frac{\sum_{i=1}^N \beta_i^*}{\sum_{i=1}^N \beta_i^* - \hat{C}\|w^*\|}$, the variables $(\mu w^*, \mu b^*, \mu \xi^*, \alpha^*, \beta^*)$ satisfy the KKT system of (19). We then show that indeed $\mu > 0$ when the data set is not linearly separable. The KKT system of (19) formulated with parameter $C \geq 0$ is as follows:

$$\xi \geq 0, \tag{35a}$$
$$\beta \geq 0, \tag{35b}$$
$$\alpha \geq 0, \tag{35c}$$
$$-\alpha - \beta + 1 = 0, \tag{35d}$$
$$\xi_i \geq -y_i(w^T x_i + b) + 1 + C\|w\|, \tag{35e}$$
$$0 \in -C\partial\|w\| \sum_{i=1}^N \beta_i + \sum_{i=1}^N y_i \beta_i x_i, \tag{35f}$$
$$\sum_{i=1}^N -y_i \beta_i = 0, \tag{35g}$$
$$\alpha_i \xi_i = 0 = (1 - \beta_i)\xi_i, \tag{35h}$$
$$\beta_i \left[-y_i(w^T x_i + b) + 1 + C\|w\| - \xi_i\right] = 0. \tag{35i}$$

When now show that $\left(w = \mu w^*, b = \mu b^*, \xi = \mu \xi^*, \alpha = \alpha^*, \beta = \beta^*, C = \frac{\hat{C}}{\sum_{i=1}^N \beta_i^*}\right)$ satisfy all of these conditions and is thus a solution to the KKT system (35).

1. $\xi \geq 0$: True, since $\xi = \mu\xi^*$, $\xi^* \geq 0$, $\mu \geq 0$.
2. $\beta \geq 0$: True, since $\beta = \beta^* \geq 0$.
3. $\alpha \geq 0$: True, since $\alpha = \alpha^*$.
4. $-\alpha - \beta + 1 = 0$: True, since $-\alpha - \beta + 1 = -\alpha^* - \beta^* + 1 = 0$.
5. $\xi_i \geq -y_i(w^T x_i + b) + 1 + C\|w\| \longrightarrow \mu\xi_i^* \geq -y_i(w^{*T}x_i + b^*)\mu + 1 + \mu C\|w^*\|$
   $\longrightarrow \xi^* \geq -y_i(w^{*T}x_i + b^*) + \left(\frac{1}{\mu} + C\|w^*\|\right) = -y_i(w^{*T}x_i + b^*)$
   $+\left(1 - \frac{\hat{C}}{\sum_{i=1}^N \beta^*}\|w^*\| + \frac{\hat{C}}{\sum_{i=1}^N \beta^*}\|w^*\|\right) = -y_i(w^{*T}x_i + b^*) + 1$: True, following from (34e).
6. $0 \in -C\partial\|w\| \sum_{i=1}^N \beta_i + \sum_{i=1}^N y_i\beta_i x_i$: True, following from
   $C\partial\|w\| \sum_{i=1}^N \beta_i = \hat{C}\partial\|\mu w^*\| = \hat{C}\partial\|w^*\|$, $\sum_{i=1}^N y_i\beta_i x_i = \sum_{i=1}^N y_i\beta_i^* x_i$, and (34f).
7. $\sum_{i=1}^N -y_i\beta_i = 0$: True, since $\beta = \beta^*$ and (34g).
8. $\alpha_i\xi_i = 0$: True, since $\alpha = \alpha^*$, $\xi = \mu\xi^*$, $\mu > 0$, and (34h).
9. $\beta_i\left[-y_i(w^T x_i + b) + 1 + C\|w\| - \xi_i\right] = 0 \iff \beta_i\xi_i = -\beta_i y_i(w^T x_i + b) + \beta_i + \beta_i C\|w\|$.
   Notice then that (35h) $\implies \beta_i\xi_i = \xi_i$. This gives us $\xi_i = \mu\xi^* = -\mu\beta_i^* y_i(w^{*T}x_i + b^*) + \beta_i^* + \beta_i^* C\|\mu w^*\| = \beta_i^* + \beta_i^* \frac{\hat{C}\mu}{\sum_{i=1}^N \beta_i^*}\|w^*\| + \mu\left[\xi_i^* - \beta_i^*\right]$ where $-\mu\beta_i^* y_i(w^{*T}x_i + b^*) = \mu\left[\xi_i^* - \beta_i^*\right]$ is implied by (35i). Furthermore, notice that this last equality is true $\iff$ $\beta_i^*\left[1 + \frac{\hat{C}\mu}{\sum_{i=1}^N \beta_i^*}\|w^*\| - \mu\right] = 0 \iff 1 + \frac{\hat{C}\|w^*\|}{\sum_{i=1}^N \beta_i^* - \hat{C}\|w^*\|} - \frac{\sum_{i=1}^N \beta_i^*}{\sum_{i=1}^N \beta_i^* - \hat{C}\|w^*\|} = 0$. Clearly, the last equality is true.

Now we show that for non-linearly separable data sets, $\mu > 0$. We use the KKT system (34) to form the dual via the Lagrangian. We then show that $\sum_{i=1}^N \beta_i^* > \hat{C}\|w^*\|$, which proves that $\mu > 0$.

We first have that the Lagrangian of (1) is the following:

$$L(w, b, \xi, \alpha, \beta) = \hat{C}\|w\| + \sum_{i=1}^N \xi_i + \sum_{i=1}^N \beta_i\left[-y_i(w^T x_i + b) + 1 - \xi_i\right] - \sum_{i=1}^N \alpha_i\xi_i$$

Using (34d,g), we can then simplify, also maximizing w.r.t. the dual variables and minimizing w.r.t. the primal variables, to form the following dual. Here, the constraints are implied by (34d,g).

$$\begin{aligned} \max_{\beta} \quad & \sum_{i=1}^N \beta_i + \left(\inf_{w,b} \hat{C}\|w\| + \sum_{i=1}^N \beta_i\left[-y_i(w^T x_i + b)\right]\right) \\ s.t. \quad & 0 \leq \beta \leq 1. \\ & \sum_{i=1}^N y_i\beta_i = 0. \end{aligned} \tag{36}$$

Notice that since $w = 0, b = 0$ is a feasible solution, we know that

$$\inf_{w,b} \quad \hat{C}\|w\| + \sum_{i=1}^N \beta_i\left[-y_i(w^T x_i + b)\right] \leq 0.$$

Finally, noting that with the optimal variables assumed to be $(w^*, b^*, \xi^*, \alpha^*, \beta^*)$, we see that

$$\sum_{i=1}^N \beta_i^* + \left(\inf_{w,b} \quad \hat{C}\|w\| + \sum_{i=1}^N \beta_i^*\left[-y_i(w^T x_i + b)\right]\right) = \hat{C}\|w^*\| + \sum_{i=1}^N \xi^*.$$

But since the data set is not linearly separable $\sum_{i=1}^N \xi^* > 0$. This implies that $\sum_{i=1}^N \beta_i^* > \hat{C}\|w^*\|$, which shows that $\mu = \frac{\sum_{i=1}^N \beta_i^*}{\sum_{i=1}^N \beta_i^* - \hat{C}\|w^*\|} > 0$. ■

B.2 Theorem 2

Proof To prove Theorem 2, we compare the KKT systems of (1) and (19) formulated with the same general norm, $\|\cdot\|$. We assume that the EC-SVM with parameter $C \geq 0$ yields optimal primal variables (w^*, b^*, ξ^*) and optimal dual variables (α^*, β^*). Thus, this is the same as assuming that $(w^*, b^*, \xi^*, \alpha^*, \beta^*)$ satisfies the following KKT system of (19):

$$\xi^* \geq 0, \tag{37a}$$
$$\beta^* \geq 0, \tag{37b}$$
$$\alpha^* \geq 0, \tag{37c}$$
$$-\alpha^* - \beta^* + 1 = 0, \tag{37d}$$
$$\xi_i^* \geq -y_i(w^{*T}x_i + b^*) + 1 + C\|w^*\|, \tag{37e}$$
$$0 \in -C\partial\|w^*\| \sum_{i=1}^N \beta_i^* + \sum_{i=1}^N y_i\beta_i^* x_i, \tag{37f}$$
$$\sum_{i=1}^N -y_i\beta_i^* = 0, \tag{37g}$$
$$\alpha_i^*\xi_i^* = 0 = (1 - \beta_i^*)\xi_i^*, \tag{37h}$$
$$\beta_i^*\left[-y_i(w^{*T}x_i + b^*) + 1 + C\|w^*\| - \xi_i^*\right] = 0. \tag{37i}$$

Now, we show that with $\mu = \frac{1}{1 + C\|w^*\|} > 0$, the variables $(\mu w^*, \mu b^*, \mu\xi^*, \alpha^*, \beta^*)$ satisfy the KKT system of (1). We then show that indeed $\mu > 0$ when the data set is not linearly

separable. The KKT system of (1) formulated with parameter $\hat{C} \geq 0$ is as follows:

$$\xi \geq 0, \tag{38a}$$
$$\beta \geq 0, \tag{38b}$$
$$\alpha \geq 0, \tag{38c}$$
$$-\alpha - \beta + 1 = 0, \tag{38d}$$
$$\xi_i \geq -y_i(w^T x_i + b) + 1, \tag{38e}$$
$$0 \in -\hat{C}\partial\|w\| + \sum_{i=1}^{N} y_i \beta_i x_i, \tag{38f}$$
$$\sum_{i=1}^{N} -y_i \beta_i = 0, \tag{38g}$$
$$\alpha_i \xi_i = 0 = (1 - \beta_i)\xi_i, \tag{38h}$$
$$\beta_i \left[-y_i(w^T x_i + b) + 1 - \xi_i\right] = 0. \tag{38i}$$

When now show, one by one, that $(w = \mu w^*, b = \mu b^*, \xi = \mu\xi^*, \alpha = \alpha^*, \beta = \beta^*, \hat{C} = C\sum_{i=1}^{N} \beta_i^*)$ satisfy all of these conditions and is thus a solution to the KKT system (38):

1. $\xi \geq 0$: True, since $\xi = \mu\xi^*$, $\xi^* \geq 0$, $\mu \geq 0$.
2. $\beta \geq 0$: True, since $\beta = \beta^* \geq 0$.
3. $\alpha \geq 0$: True, since $\alpha = \alpha^*$.
4. $-\alpha - \beta + 1 = 0$: True, since $-\alpha - \beta + 1 = -\alpha^* - \beta^* + 1 = 0$.
5. $\xi_i \geq -y_i(w^T x_i + b) + 1 \longrightarrow \mu\xi_i^* \geq -y_i(w^{*T} x_i + b^*)\mu + 1 \iff \xi^* \geq -y_i(w^{*T} x_i + b^*) + (1 + C\|w^*\|)$: True, following from (37e).
6. $0 \in -\hat{C}\partial\|w^*\| + \sum_{i=1}^{N} y_i\beta_i x_i$: True, following from $\hat{C}\partial\|w^*\| = C\partial\|w\| \sum_{i=1}^{N} \beta_i$, $\sum_{i=1}^{N} y_i\beta_i x_i = \sum_{i=1}^{N} y_i\beta_i^* x_i$, and (37f).
7. $\sum_{i=1}^{N} -y_i\beta_i = 0$: True, since $\beta = \beta^*$ and (37g).
8. $\alpha_i\xi_i = 0$: True, since $\alpha = \alpha^*$, $\xi = \mu\xi^*$, $\mu > 0$, and (37h).
9. $\beta_i \left[-y_i(w^T x_i + b) + 1 - \xi_i\right] = \beta_i^* \left[-\mu y_i(w^{*T} x_i + b^*) + 1 - \mu\xi_i^*\right] = \beta_i^* + \frac{\beta_i^*\left(-y_i(w^{*T} x_i + b^*) - \xi_i^*\right)}{1 + C\|w^*\|} = 0 \iff \beta_i^* \left[-y_i(w^{*T} x_i + b^*) + 1 + C\|w^*\| - \xi_i^*\right] = 0$, which is True, following from (37i).

Now we point out that $\mu > 0$. This follows immediately from the assumption that $C \geq 0$ and the fact that $\|w^*\| \geq 0$. ■

Appendix C. Derivation of EC-SVM Dual Formulation

The dual of the EC-SVM is formulated as follows, via the Lagrangian:

$$\begin{aligned}
&\max_{\substack{\alpha \geq 0 \\ \beta \geq 0}} \min_{w,b} \quad \sum_{i=1}^{N} \xi_i + \sum_{i=1}^{N} \beta_i \left[-y_i(w^T x_i + b) + 1 - \xi_i + C\|w\|\right] - \sum_{i=1}^{N} \alpha_i \xi_i \\
&= \max_{\substack{0 \leq \beta \leq 1 \\ \sum_{i=1}^{N} \beta_i y_i = 0}} \quad \sum_{i=1}^{N} \beta_i + \left[\min_{w} \quad C\|w\| \sum_{i=1}^{N} \beta_i + \sum_{i=1}^{N} -\beta_i y_i w^T x_i\right] \\
&= \max_{\substack{0 \leq \beta \leq 1 \\ \sum_{i=1}^{N} \beta_i y_i = 0}} \quad \sum_{i=1}^{N} \beta_i + \left[\min_{w \neq 0, a \geq 0} \quad aC \sum_{i=1}^{N} \beta_i + \frac{a\sum_{i=1}^{N} -\beta_i y_i w^T x_i}{\|w\|}\right] \\
&= \max_{\substack{0 \leq \beta \leq 1 \\ \sum_{i=1}^{N} \beta_i y_i = 0}} \quad \sum_{i=1}^{N} \beta_i + \left[\min_{a \geq 0} \quad aC \sum_{i=1}^{N} \beta_i + a \min_{w \neq 0} \frac{\sum_{i=1}^{N} -\beta_i y_i w^T x_i}{\|w\|}\right] \\
&= \max_{\substack{0 \leq \beta \leq 1 \\ \sum_{i=1}^{N} \beta_i y_i = 0}} \quad \sum_{i=1}^{N} \beta_i + \left[\min_{a \geq 0} aC \sum_{i=1}^{N} \beta_i - a \min_{w \neq 0} \quad \frac{\sum_{i=1}^{N} \beta_i y_i w^T x_i}{\| - w\|}\right] \\
&= \max_{\substack{0 \leq \beta \leq 1 \\ \sum_{i=1}^{N} \beta_i y_i = 0}} \quad \sum_{i=1}^{N} \beta_i + \left[\min_{a \geq 0} \quad aC \sum_{i=1}^{N} \beta_i - a \left\|\sum_{i=1}^{N} -\beta_i y_i x_i\right\|^*\right] \\
&\quad = \max_{\beta} \quad \sum_{i=1}^{N} \beta_i \\
&\qquad s.t. \quad \left\|\sum_{i=1}^{N} \beta_i y_i x_i\right\|^* \leq C \sum_{i=1}^{N} \beta_i, \\
&\qquad\qquad \sum_{i=1}^{N} \beta_i y_i = 0, \\
&\qquad\qquad 0 \leq \beta_i \leq 1, \quad \forall i \in \{1, \ldots, N\}.
\end{aligned}$$

Appendix D. Derivation of C-SVM Dual Formulation

The dual of the C-SVM is formulated as follows, via the Lagrangian:

$$
\begin{aligned}
&\max_{\substack{\alpha\geq 0\\ \beta\geq 0}} \quad \min_{w,b} \quad \sum_{i=1}^{N}\xi_i + C\|w\| + \sum_{i=1}^{N}\beta_i\left[-y_i(w^Tx_i+b)+1-\xi_i\right] - \sum_{i=1}^{N}\alpha_i\xi_i \\
&= \max_{\substack{0\leq\beta\leq 1\\ \sum_{i=1}^N\beta_iy_i=0}} \quad \sum_{i=1}^{N}\beta_i + \left[\min_{w} \quad C\|w\| + \sum_{i=1}^{N}-\beta_iy_iw^Tx_i\right] \\
&= \max_{\substack{0\leq\beta\leq 1\\ \sum_{i=1}^N\beta_iy_i=0}} \quad \sum_{i=1}^{N}\beta_i + \left[\min_{w\neq 0,a\geq 0} \quad aC + \frac{a\sum_{i=1}^{N}-\beta_iy_iw^Tx_i}{\|w\|}\right] \\
&= \max_{\substack{0\leq\beta\leq 1\\ \sum_{i=1}^N\beta_iy_i=0}} \quad \sum_{i=1}^{N}\beta_i + \left[\min_{a\geq 0} \quad aC + a\min_{w\neq 0}\frac{\sum_{i=1}^{N}-\beta_iy_iw^Tx_i}{\|w\|}\right] \\
&= \max_{\substack{0\leq\beta\leq 1\\ \sum_{i=1}^N\beta_iy_i=0}} \quad \sum_{i=1}^{N}\beta_i + \left[\min_{a\geq 0} \quad aC - a\min_{w\neq 0}\frac{\sum_{i=1}^{N}\beta_iy_iw^Tx_i}{\|-w\|}\right] \\
&= \max_{\substack{0\leq\beta\leq 1\\ \sum_{i=1}^N\beta_iy_i=0}} \quad \sum_{i=1}^{N}\beta_i + \left[\min_{a\geq 0} \quad aC - a\left\|\sum_{i=1}^{N}-\beta_iy_ix_i\right\|^*\right] \\
&= \max_{\beta} \quad \sum_{i=1}^{N}\beta_i \\
&\qquad s.t. \quad \left\|\sum_{i=1}^{N}\beta_iy_ix_i\right\|^* \leq C, \\
&\qquad\qquad \sum_{i=1}^{N}\beta_iy_i = 0, \\
&\qquad\qquad 0\leq\beta_i\leq 1, \quad \forall i\in\{1,\ldots,N\}.
\end{aligned}
$$

Appendix E. Derivation of Formula (21)

To see how we derive (21), we begin with formula (14) with $f(w,X) = -y(w^TX+b)$ and $z=-C$:

$$
\begin{aligned}
\min_{\gamma<-C,w,b} \quad & \sum_{i=1}^{n}\left[\frac{-y_i(w^TX_i+b)}{-C-\gamma} - \frac{\gamma}{-C-\gamma}\right]^+ \\
s.t. \quad & \|w\| = 1.
\end{aligned} \tag{39}
$$

We then make the change of variable $w_{new} = \frac{w}{C}, b_{new} = \frac{b}{C}$. Note then that

$$
\begin{aligned}
\frac{-y_i(w^TX_i+b)}{-C-\gamma} &= C\frac{-y_i(w_{new}^TX_i+b_{new})}{-C-\gamma} \\
&= \frac{-y_i(w_{new}^TX_i+b_{new})}{\|w_{new}\|(-C-\gamma)}.
\end{aligned}
$$

Plugging this and the new variables into (41), we get:

$$
\begin{aligned}
\min_{\gamma<-C,w_{new},b_{new}} \quad & \sum_{i=1}^{n}\left[\frac{-y_i(w_{new}^TX_i+b_{new})}{\|w_{new}\|(-C-\gamma)} - \frac{\gamma}{-C-\gamma}\right]^+ \\
s.t. \quad & \|w_{new}\| = \frac{1}{C}.
\end{aligned} \tag{40}
$$

Finally, note that

$$
\begin{aligned}
\frac{1}{\|w_{new}\|(-C-\gamma)} + 1 &= \frac{1+\|w_{new}\|(-C-\gamma)}{\|w_{new}\|(-C-\gamma)} \\
&= \frac{1+(-1-\frac{\gamma}{C})}{(-1-\frac{\gamma}{C})} \\
&= \frac{-\gamma}{C}\left(\frac{C}{-C-\gamma}\right) \\
&= \frac{-\gamma}{-C-\gamma}.
\end{aligned}
$$

Plugging this into (40), we finally arrive at

$$
\begin{aligned}
\min_{\gamma<-C,w_{new},b_{new}} \quad & \sum_{i=1}^{n}\left[\left(\frac{1}{-C-\gamma}\right)\left(\frac{-y_i(w_{new}^TX_i+b_{new})+1}{\|w_{new}\|}\right)+1\right]^+ \\
s.t. \quad & \frac{1}{\|w_{new}\|} = C.
\end{aligned} \tag{41}
$$

Appendix F. Bregman Divergence

For a convex function $f:\mathbb{R}^n\to\mathbb{R}$, let $\partial f(w)$ denote the set of subderivatives $\nabla f(w)\in\partial f(w)$ at a point $w\in\mathbb{R}^n$. For a differentiable, convex f the Bregman Divergence at $w_1,w_2\in\mathbb{R}^n$ is defined as

$$
d_f(w_1,w_2) = f(w_1) - f(w_2) - \langle\nabla f(w_2), w_1-w_2\rangle\ . \tag{42}
$$

Since f is differentiable, $\nabla f(w)$ is the unique subderivative, and thus the divergence is well defined. If f is convex, but not differentiable, $\nabla f(w)$ may not be unique. In this case, however, following Appendix C of Bousquet and Elisseeff (2002), we can define a generalized Bregman Divergence. Letting the function $f^*(a) = \sup_w\langle w,a\rangle - f(w)$ denote the conjugate of f, we define the generalized Bregman Divergence at $w_1, a\in\mathbb{R}^n$ as,

$$
\bar{d}_f(w_1,a) = f(w_1) + f^*(a) - \langle w_1,a\rangle\ . \tag{43}
$$

As it relates to the normal Bregman Divergence, it is easy to check that for convex differentiable f,

$$d_f(w_1, w_2) = \bar{d}_f(w_1, \nabla f(w_2)) \; . \tag{44}$$

which follows from the property that

$$a \in \partial f(w) \iff f^*(a) = \langle a, w \rangle - f(w) \; .$$

As for its other properties, first note that the generalized divergence is non-negative. Second, notice that we have linearity, such that if $f = g + h$, for convex functions g, h, and we choose subderivatives of f, g, h satisfying $\nabla f(w_2) = \nabla g(w_2) + \nabla h(w_2)$, then

$$\bar{d}_f(w_1, \nabla f(w_2)) = \bar{d}_g(w_1, \nabla g(w_2)) + \bar{d}_h(w_1, \nabla h(w_2) \; . \tag{45}$$

To see this, we simply need to expand the right hand side in the following manner, where we use the fact that $a \in \partial f(w) \iff f^*(a) = \langle a, w \rangle - f(w)$ in the third and fifth equality.

$$\begin{aligned}
\bar{d}_f(w_1, \nabla f(w_2)) &= \bar{d}_{(g+h)}(w_1, \nabla(g+h)(w_2)) \\
&= (g+h)(w_1) + (g+h)^*(\nabla(g+h)(w_2)) - \langle w_1, \nabla(g+h)(w_2) \rangle \\
&= [(g+h)(w_1) - \langle w_1, \nabla(g+h)(w_2) \rangle] + \langle w_2, \nabla(g+h)(w_2) \rangle - (g+h)(w_2) \\
&= [(g+h)(w_1) - \langle w_1, \nabla(g+h)(w_2) \rangle] + (\langle w_2, \nabla g(w_2) \rangle - g(w_2)) \\
&\qquad + (\langle w_2, \nabla h(w_2) \rangle - h(w_2)) \\
&= [(g+h)(w_1) - \langle w_1, \nabla(g+h)(w_2) \rangle] + g^*(\nabla g(w_2)) + h^*(\nabla h(w_2)) \\
&= (g(w_1) + g^*(\nabla g(w_2)) - \langle w_1, \nabla g(w_2) \rangle) \\
&\qquad + (h(w_1) + h^*(\nabla h(w_2)) - \langle w_1, \nabla h(w_2) \rangle) \\
&= \bar{d}_g(w_1, \nabla g(w_2)) + \bar{d}_h(w_1, \nabla h(w_2) \; .
\end{aligned}$$

Finally, assume that w_f is a minimizer of f, meaning that $0 \in \partial f(w_f)$. Then, we have for any w_1,

$$\bar{d}_f(w_1, 0) = f(w_1) - f(w_f) \; . \tag{46}$$

References

Jinbo Bi and Tong Zhang. Support vector classification with input data uncertainty. *Advances in neural information processing systems*, 17:161, 2005.

Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526, 2002.

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and trends® in machine learning*, 3(1):1–122, 2011.

Chih-Chung Chang and Chih-Jen Lin. Training v-support vector classifiers: theory and algorithms. *Neural computation*, 13(9):2119–2147, 2001.

Olivier Chapelle. Training a support vector machine in the primal. *Neural computation*, 19 (5):1155–1178, 2007.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3): 273–297, 1995.

Justin Davis and Stan Uryasev. Analysis of hurricane damage using buffered probability of exceedance. *Research report 2014-4, ise dept., university of florida*, 2014.

Tao Pham Dinh and Hoai An Le Thi. Recent advances in dc programming and dca. In *Transactions on computational intelligence XIII*, pages 1–37. Springer, 2014.

Alexander Mafusalov and Stan Uryasev. Buffered probability of exceedance: Mathematical properties and optimization algorithms. *Research report 2014-1, ise dept., university of florida*, 2015.

Matthew Norton and Stan Uryasev. Maximization of auc and buffered auc in binary classification. *Research report 2014-2, ise dept., university of florida*, 2014.

Matthew Norton, Alexander Mafusalov, and Stan Uryasev. Cardinality of upper average and its application to network optimization. *Research report 2015-1, ise dept., university of florida*, 2015.

Fernando Pérez-Cruz, Jason Weston, DJL Herrmann, and Bernhard Schölkopf. Extension of the nu-svm range for classification. *Nato science series sub series III computer and system sciences*, 190:179–196, 2003.

Ralph Tyrell Rockafellar. Safeguarding strategies in risky optimization. *Presentation at the international workshop on engineering risk control and optimization, Gainesville, FL,*, 2009.

Ralph Tyrell Rockafellar. *Convex analysis*. Princeton university press, 2015.

Ralph Tyrell Rockafellar and Johannes Royset. On buffered failure probability in design and optimization of structures. *Reliability engineering & system safety, Vol. 95, 499-510*, 2010.

Ralph Tyrell Rockafellar and Stan Uryasev. Conditional value-at-risk for general loss distributions. *Journal of banking & finance*, 26(7):1443–1471, 2002.

Bernhard Schölkopf, Alex J Smola, Robert C Williamson, and Peter L Bartlett. New support vector algorithms. *Neural computation*, 12(5):1207–1245, 2000.

Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *International conference on computational learning theory*, pages 416–426. Springer, 2001.

Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.

Akiko Takeda and Masashi Sugiyama. ν-support vector machine as conditional value-at-risk minimization. In *Proceedings of the 25th international conference on machine learning*, pages 1056–1063. ACM, 2008.

Pham Dinh Tao and Le Thi Hoai An. Convex analysis approach to dc programming: theory, algorithms and applications. *Acta mathematica vietnamica*, 22(1):289–355, 1997.

Stan Uryasev. Buffered probability of exceedance and buffered service level: Definitions and properties. *Research report 2014-3, ise dept., university of florida*, 2014.

Ximing Wang, Neng Fan, and Panos M Pardalos. Stochastic subgradient descent method for large-scale robust chance-constrained support vector machines. *Optimization letters*, pages 1–12, 2016.

Huan Xu, Constantine Caramanis, and Shie Mannor. Robustness and regularization of support vector machines. *Journal of machine learning research*, 10(Jul):1485–1510, 2009.

Journal of Machine Learning Research 18 (2017) 1-29 Submitted 11/15; Revised 7/17; Published 8/17

Variational Particle Approximations

Ardavan Saeedi* ardavans@mit.edu
Computer Science & Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Tejas D. Kulkarni* tejasdkulkarni@gmail.com
DeepMind, London

Vikash K. Mansinghka vkm@mit.edu
Department of Brain & Cognitive Sciences
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

Samuel J. Gershman gershman@fas.harvard.edu
Department of Psychology and Center for Brain Science
Harvard University
Cambridge, MA 02138, USA

Editor: Lawrence Carin

Abstract

Approximate inference in high-dimensional, discrete probabilistic models is a central problem in computational statistics and machine learning. This paper describes discrete particle variational inference (DPVI), a new approach that combines key strengths of Monte Carlo, variational and search-based techniques. DPVI is based on a novel family of particle-based variational approximations that can be fit using simple, fast, deterministic search techniques. Like Monte Carlo, DPVI can handle multiple modes, and yields exact results in a well-defined limit. Like unstructured mean-field, DPVI is based on optimizing a lower bound on the partition function; when this quantity is not of intrinsic interest, it facilitates convergence assessment and debugging. Like both Monte Carlo and combinatorial search, DPVI can take advantage of factorization, sequential structure, and custom search operators. This paper defines DPVI particle-based approximation family and partition function lower bounds, along with the sequential DPVI and local DPVI algorithm templates for optimizing them. DPVI is illustrated and evaluated via experiments on lattice Markov Random Fields, nonparametric Bayesian mixtures and block-models, and parametric as well as non-parametric hidden Markov models. Results include applications to real-world spike-sorting and relational modeling problems, and show that DPVI can offer appealing time/accuracy trade-offs as compared to multiple alternatives.

Keywords: Bayesian inference, variational methods, Dirichlet process mixture model, Ising model, hidden Markov model, infinite relational model

1. Introduction

Probabilistic models defined over large collections of discrete random variables have arisen in multiple fields. Examples include hidden Markov models for sequential data; Bayesian networks; mixture models for tabular and relational data; and discrete Markov random field models, which have become popular in fields ranging from computer vision to information extraction. These models are typically specified in terms of a probability distribution $P(x)$ defined over a collection of variables $x = \{x_n\}$ occupying points in an underlying discrete space $\mathcal{X}$. One key approximate inference problem that arises in many applications is identifying high-probability configurations of the discrete variables.

Most approximate inference algorithms fall into two classes: Monte Carlo methods and variational methods. Monte Carlo methods generate samples from approximations to the posterior distribution that grow more accurate as the technique is given more compute time. The flexibility and simplicity of Monte Carlo methods have made them the workhorse of statistical computation (Robert and Casella, 2004). There are two basic approaches to Monte Carlo inference for discrete probabilistic models. The first, Markov chain Monte Carlo methods, work by running a Markov chain with transition operator T whose equilibrium distribution asymptotically approaches $P(x)$—that is, $T^k P_0(x) \approx P(x)$ for any initial distribution on states $P_0(x)$. The second, sequential Monte Carlo methods, build up a sample from a distribution that approximates $P(x)$ by sampling from a sequence of more tractable distributions, typically defined over subspaces of $\mathcal{X}$. However, their accuracy is difficult to measure, and the amount of computation required for satisfactory accuracy can be prohibitive in practice.

Variational methods (Wainwright and Jordan, 2008) treat probabilistic inference as an optimization problem over a set of distributions. This set is typically constrained (e.g., to factorized conjugate exponential distributions), thereby attaining efficiency at the expense of bias. In particular, variational methods tend to converge quickly and supply an easily monitored objective function (unlike Monte Carlo methods). However, for complex discrete models, the bias induced by variational approximations can sometimes lead to poor predictive performance. For example, consider a discrete probabilistic model where two binary variables x_1 and x_2 are constrained to take the same value, i.e. $P(x_1, x_2) = 0$ if $x_1 \neq x_2$. All the probability mass is on the states $x_1 = x_2 = 0$ and $x_1 = x_2 = 1$. In "mean-field" variational inference, this distribution might be approximated as $Q_{\theta_1}(x_1)Q_{\theta_2}(x_2)$, with θ_1 and θ_2 representing coin weights that used to model x_1 and x_2 as independent Bernoulli distributions. This family of variational approximations to $P(x_1, x_2)$ cannot capture the true distribution, and in fact cannot qualitatively capture the simple constraint that $x_1 = x_2$. These limitations prompted the development of more sophisticated approximations (e.g., Bouchard-Côté and Jordan, 2009; Jaakkola and Jordan, 1998), but these incur additional computational cost and can be difficult or impossible to apply to a given problem.

This paper introduces a new approximate inference method, called *discrete particle variational inference* (DPVI), that aims to combine key strengths of both Monte Carlo and variational inference. The key insight in DPVI is to use a weighted collection of samples—the kind of "particle approximation" output by Monte Carlo methods—as the approximating family for variational inference. Suppose we got to pick where to place the particles in the hypothesis space; where would we put them? Intuitively, we would want to distribute them in such a way that they cover high probability regions of the target distribution, but without the particles all devolving onto the mode of the distribution. This problem can be formulated precisely within the framework of variational inference. We derive a coordinate

©2017 Ardavan Saeedi, Tejas D. Kulkarni, Vikash K. Mansingkha, and Samuel J. Gershman.
* Denotes equal contribution.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/15-615.html

ascent update for particle approximations that iteratively minimizes the Kullback-Leibler (KL) divergence between the particle approximation and the target distribution.

In DPVI, the location of the particles become the parameters of the approximating family. This simple choice has appealing consequences. Like Monte Carlo, DPVI can handle problems where the posterior has multiple modes, and yields exact results in a well-defined limit (as the number of particles goes to infinity). Like standard mean-field variational methods, DPVI is based on optimizing a lower bound on the partition function; when this quantity is not of intrinsic interest, it facilitates convergence assessment and debugging. Like both Monte Carlo and combinatorial search, DPVI can take advantage of factorization, sequential structure, and custom search operators.

The rest of the paper is organized as follows. After introducing our general framework, we describe how it can be applied to filtering and smoothing problems. We then show experimentally that variational particle approximations can overcome a number of problems that are challenging for conventional Monte Carlo methods. In particular, our approach is able to produce a diverse, high probability set of particles in situations where Monte Carlo and mean-field variational methods sometimes degenerate.

2. Background

Consider the problem of approximating a probability distribution $P(x)$ over discrete latent variables $x = \{x_1, \ldots, x_N\}, x_n \in \{1, \ldots, M_n\}$, where the target distribution is known only up to a normalizing constant Z: $P(x) = f(x)/Z$. We will refer to $f(x) \geq 0$ as the *score* of x and Z as the *partition function*. We further assume that $P(x)$ is a Markov network defined on a graph G, so that $f(x)$ factorizes according to:

$$f(x) = \prod_c f_c(x_c), \tag{1}$$

where $c \subseteq \{1, \ldots, N\}$ indexes the maximal cliques of G.

2.1 Importance sampling and sequential Monte Carlo

A general way to approximate $P(x)$ is with a weighted collection of K particles, $\{x^1, \ldots, x^K\}$:

$$P(x) \approx Q(x) = \sum_{k=1}^{K} w^k \delta[x, x^k], \tag{2}$$

where $x^k = \{x_1^k, \ldots, x_N^k\}, x_n^k \in \{1, \ldots, M_n\}$ and $\delta[\cdot, \cdot] = 1$ if its arguments are equal and 0 otherwise. Importance sampling is a Monte Carlo method that stochastically generates particles from a proposal distribution, $x^k \sim \phi(\cdot)$, and computes the weight according to $w^k \propto f(x^k)/\phi(x^k)$. Importance sampling has the property that the particle approximation converges to the target distribution as $K \to \infty$ (Robert and Casella, 2004).

Sequential Monte Carlo methods such as particle filtering (Doucet et al., 2001) apply importance sampling to stochastic dynamical systems (where n indexes time) by sequentially sampling the latent variables at each time point using a proposal distribution $\phi(x_n|x_{n-1})$. This procedure can produce conditionally low probability particles; therefore, most algorithms include a resampling step which replicates high probability particles and kills off

low probability particles. The downside of resampling is that it can produce degeneracy: the particles become concentrated on a small number of hypotheses, and consequently the effective number of particles is low.

2.2 Variational Inference

Variational methods (Wainwright and Jordan, 2008) define a parametrized family of probability distributions $\mathcal{Q}$ and then choose $Q \in \mathcal{Q}$ that maximizes the *negative variational free energy*:

$$\mathcal{L}[Q] = \sum_x Q(x) \log \frac{f(x)}{Q(x)}. \tag{3}$$

The negative variational free energy is related to the partition function Z and the KL divergence through the following identity:

$$\log Z = \mathrm{KL}[Q||P] + \mathcal{L}[Q], \tag{4}$$

where

$$\mathrm{KL}[Q||P] = \sum_x Q(x) \log \frac{Q(x)}{P(x)}. \tag{5}$$

Since $\mathrm{KL}[Q||P] \geq 0$, the negative variational free energy is a lower bound on the log partition function, achieving equality when the KL divergence is minimized to 0. Maximizing $\mathcal{L}[Q]$ with respect to Q is thus equivalent to minimizing the KL divergence between Q and P.

Unlike the Monte Carlo methods described in the previous section, variational methods do not in general converge to the target distribution, since typically P is not in $\mathcal{Q}$. The advantage of variational methods is that they guarantee an improved bound after each iteration, and convergence is easy to monitor (unlike most Monte Carlo methods). In practice, variational methods are also often more computationally efficient.

We next consider particle approximations from the perspective of variational inference. We then turn to the application of particle approximations to inference in stochastic dynamical systems.

3. Variational Particle Approximations

Variational inference can be connected to Monte Carlo methods by viewing the particles as a set of variational parameters parameterizing Q. For the particle approximation defined in Eq. 2, the negative variational free energy takes the following form:

$$\mathcal{L}[Q] = \sum_{k=1}^{K} w^k \log \frac{f(x^k)}{w^k V^k}, \tag{6}$$

where $V^k = \sum_{j=1}^{K} \delta[x^j, x^k]$ is the number of times an identical replica of x^k appears in the particle set. We wish to find the set of K particles and their associated weights that maximize $\mathcal{L}[Q]$, subject to the constraint that $\sum_{k=1}^{K} w^k = 1$. This constraint can be implemented

by defining a new functional with Lagrange multiplier λ:

$$\tilde{\mathcal{L}}[Q] = \mathcal{L}[Q] + \lambda\left(\sum_{k=1}^{K} w^k - 1\right). \tag{7}$$

Taking the functional derivative of the Lagrangian with respect to w^k and equating to zero, we obtain:

$$\begin{aligned}\frac{\partial\tilde{\mathcal{L}}[Q]}{\partial w^k} &= \log f(x^k) - \log w^k - \log V^k + \lambda - 1 = 0\\ &\implies w^k = Z_Q^{-1} f(x^k)/V^k,\end{aligned} \tag{8}$$

where

$$Z_Q = \exp(\lambda - 1)^{-1} = \sum_{k=1}^{K} \frac{f(x^k)}{V^k}. \tag{9}$$

We can plug the above result back into the definition of $\mathcal{L}[Q]$:

$$\begin{aligned}\mathcal{L}[Q] &= Z_Q^{-1}\sum_{k=1}^{K}\frac{f(x^k)}{V^k}\log\frac{f(x^k)V^k}{Z_Q^{-1}f(x^k)V^k}\\ &= Z_Q^{-1}\sum_{k=1}^{K}\frac{f(x^k)}{V^k}\log Z_Q\\ &= \log Z_Q.\end{aligned} \tag{10}$$

Thus, $\mathcal{L}[Q]$ is maximized by choosing the K values of x with the highest score. The following theorem shows that allowing $V^k > 1$ (i.e., having replica particles) can never improve the bound.

Theorem 1 *Let Q and Q' denote two particle approximations, where Q consists of unique particles ($V_Q^k = 1$ for all k) and Q' is identical to Q except that particle $x_{Q'}^j$ is replicated $V_{Q'}^j$ times (displacing $V_{Q'}^j$ other particles with cumulative score F). For any choice of particles, $\mathcal{L}[Q] \geq \mathcal{L}[Q']$.*

Proof We first apply Jensen's inequality to obtain an upper bound on $\mathcal{L}[Q']$:

$$\mathcal{L}[Q'] \leq \log\sum_{k=1}^{K} w_{Q'}^k Z_{Q'} = \log\sum_{k=1}^{K}\frac{f(x_{Q'}^k)}{V_{Q'}^k}. \tag{11}$$

Since $\mathcal{L}[Q] = \log Z_Q$, we wish to show that $Z_Q \geq \sum_{k=1}^{K}\frac{f(x_{Q'}^k)}{V_{Q'}^k}$. All the particles in Q and Q' are identical except for the $V_{Q'}^j$ particles in Q that were displaced by replicas of $x_{Q'}^j$ in Q'; thus we only need to establish that $f(x_{Q'}^j) + F \geq \frac{V_{Q'}^j f(x_{Q'}^j)}{V_{Q'}^j} = f(x_{Q'}^j)$, where the left hand side of the inequality is the change in negative variational free energy after the replication

of particles. Since the score $f(x)$ can never be negative, the cumulative score F can also never be negative ($F \geq 0$) and the inequality holds for any choice of particles. ■

The variational bound can be optimized by coordinate ascent, as specified in Algorithm 1, which we refer to as *discrete particle variational inference* (DPVI). This algorithm takes advantage of the fact that when optimizing the bound with respect to a single variable, only potentials local to that variable need to be computed. In particular, let $\tilde{x}^k$ be a replica of x^k with a single-variable modification, $\tilde{x}_n^k = m$. We can compute the unnormalized probability of this particle efficiently using the following equation:

$$f(\tilde{x}^k) = f(x^k)\frac{\mathcal{F}_n(\tilde{x}^k)}{\mathcal{F}_n(x^k)}, \tag{12}$$

where $\mathcal{F}_n(x) = \prod_{c:n\in c} f_c(x_c)$. The variational bound for the modified particle can then be computed using Eq. 10. Particles can be initialized arbitrarily. When repeatedly iterated, DPVI will converge to a local maximum of the negative variational free energy.[1] Note that in principle more sophisticated methods can be used to find the top K modes (e.g., Flerova et al., 2012; Yanover and Weiss, 2003); however, we have found that this coordinate ascent algorithm is fast, easy to implement, and very effective in practice (as our experiments below demonstrate).

An important aspect of this framework is that it maintains one of the same asymptotic guarantees as importance sampling: Q converges to P as $K \to \infty$, since in this limit DPVI is equivalent to exact inference. Thus, DPVI combines advantages of variational methods (monotonically decreasing KL divergence between Q and P) with the asymptotic correctness of Monte Carlo methods. It is important to note that asymptotic correctness might be useless in practice unless something is known about the convergence rate. This issue is not unique to DPVI; it also applies to Monte Carlo and variational methods. For certain Markov chain Monte Carlo (MCMC) samplers, it can be shown that the chain converges to the posterior at a geometric rate (Mengersen et al., 1996; Meyn and Tweedie, 1993). A small amount of work has investigated convergence properties of variational methods for specific models (Hall et al., 2011; Wang et al., 2006), but in general the issue of convergence rate for variational methods is an open question.

The asymptotic complexity of DPVI in the sequential setting is $O(SNK)$ where S is the maximum support size of the latent variables. For the iterative update of the particles the complexity is $O(TCSK)$, where T is the maximum number of iterations until convergence and C is the maximum clique size. In our experiments, we empirically observed that we only need a small number of iterations and particles in order to outperform our baselines.

1. Naturally, initialization affects performance, since the objective function has local optima. For example, if the posterior is multimodal and none of the particles are initialized near the dominant mode, then the particle approximation will likely miss a significant portion of the probability mass. Studying the effects of initialization is an important practical challenge for the application of DPVI. In our experiments, we report averages across multiple random initializations.

Algorithm 1 Discrete particle variational inference

/*N is the number of latent variables */
/*x^k is the set of all latent variables for the kth particle: $x^k = \{x_1^k, \ldots, x_N^k\}$ */
/*M_n is the support of latent variable x_n */
Input: initial particle approximation Q with K particles, tolerance ϵ
while $|\mathcal{L}[Q] - \mathcal{L}[Q']| > \epsilon$ **do**
  for $n = 1$ **to** N **do**
    $\mathcal{X} = \emptyset$
    for $k = 1$ **to** K **do**
      Copy particle k: $\tilde{x}^k \leftarrow x^k$
      for $m = 1$ **to** M_n **do**
        Modify particle: $\tilde{x}_n^k \leftarrow m$
        Score $\tilde{x}^k$ using Eq. 12
        $\mathcal{X} \leftarrow \mathcal{X} \cup (\tilde{x}^k, f(\tilde{x}^k))$
      end for
    end for
    Select K unique particles from $\mathcal{X}$ with the largest scores
    Construct new particle approximation $Q'(x) = \sum_{k=1}^K w^k \delta[x, x^k]$
    Compute variational bound $\mathcal{L}[Q']$ using Eq. 10
  end for
end while
return particle approximation Q'

4. Filtering and Smoothing in Hidden Markov Models

We now describe how variational particle approximations can be applied to filtering and smoothing in hidden Markov models (HMMs). Consider a hidden Markov model with observations $y = \{y_1, \ldots, y_N\}$ generated by the following stochastic process:

$$P(y, x, \theta) = P(\theta) \prod_n P(y_n|x_n, \theta) P(x_n|x_{n-1}, \theta), \tag{13}$$

where θ is a set of transition and emission parameters. We are particularly interested in *marginalized* HMMs where the parameters are integrated out: $P(y, x) = \int_\theta P(y, x, \theta) d\theta$. This induces dependencies between observation n and all previous observations, making inference challenging.

Filtering is the problem of computing the posterior over the latent variables at time n given the history $y_{1:n}$. To construct the variational particle approximation of the filtering distribution, we need to compute the product of potentials for variable n:

$$\mathcal{F}_n(x) = P(y_n|x_{1:n}, y_{1:n-1}) P(x_n|x_{1:n-1}). \tag{14}$$

Recall from the previous section that $\mathcal{F}_n(x)$ is the joint probability of the maximal cliques to which x_n belongs. We can then apply the coordinate ascent update described in the previous section. This update is simplified in the filtering context due to the underlying Markov structure. Specifically, Eq. 12 is given by:

$$f(\tilde{x}^k) = f(x^k) P(y_n|x_n^k = m, x_{1:n-1}^k, y_{1:n-1}) P(x_n^k = m|x_{1:n-1}^k). \tag{15}$$

At each time step, the algorithm selects the K continuations (new variable assignments of the current particle set) that maximize the negative variational free energy.

Smoothing is the problem of computing the posterior over the latent variables at time n given data from both the past and the future, $y_{1:N}$. The product of potentials is given by:

$$\mathcal{F}_n(x) = P(y_n|x_{1:n}, y_{-n}) P(x_n|x_{-n}), \tag{16}$$

where x_{-n} refers to all the latent variables except x_n (and likewise for y_{-n}). This potential can be plugged into the updates described in the previous section.

To understand DPVI applied to filtering problems, it is helpful to contemplate three possible fates for a particle at time n (illustrated in Figure 1):

- **Selection**: A single continuation of particle k has non-zero weight. This can be seen as a deterministic version of particle filtering, where the sampling operation is replaced with a max operation.
- **Splitting**: Multiple continuations of particle k have non-zero weight. In this case, the particle is split into multiple particles at the next iteration.
- **Deletion**: No continuations of particle k have non-zero weight. In this case, the particle is deleted from the particle set.

Similar to particle filtering with resampling, DPVI deletes and propagates particles based on their probability. However, as we show later, DPVI is able to escape some of the problems associated with resampling.

5. Related Work

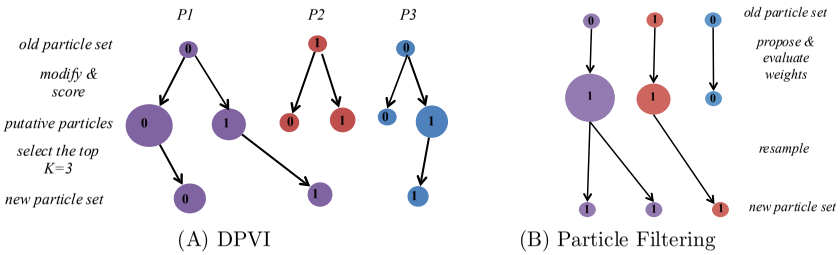
DPVI is related to several other ideas in the statistics literature:

- DPVI is a special case of a *mixture mean-field variational approximation* (Jaakkola and Jordan, 1998; Lawrence, 2000):

$$Q(x) = \sum_{k=1}^K Q(k) \prod_{n=1}^N Q(x_n|k). \tag{17}$$

  In DPVI, $Q(k) = w^k$ and $Q(x_n|k) = \delta[x_n, x_n^k]$. From the simple restriction that the component distributions must be delta functions, we derived a new algorithm that is simpler and more efficient than mixture mean-field (which requires separate updates for the mixture weights), while sacrificing some of its expressivity. Another distinct advantage of DPVI is that the variational updates do not require the additional lower bound used in general mixture mean-field, due to the intractability of the mean-field updates.

- When $K = 1$, DPVI is equivalent to *iterated conditional modes* (ICM; Besag, 1986), which iteratively maximizes each latent variable conditional on the rest of the variables. This algorithm is simple to implement, efficient (relative to variational and Monte Carlo algorithms), and has been successfully applied to computer vision tasks



(A) DPVI (B) Particle Filtering

Figure 1: Schematic of DPVI versus particle filtering for filtering problems. Illustration of different filtering scenarios over 2 time steps in a binary state space with $K = 3$ particles. The number in each circle indicates the binary value of the corresponding variable. Arrows indicate the evolution of the particles. (A) DPVI: The size of the putative particles represents the score of the particle. The K continuations with highest score are selected for propagation to the next time step. The size of the new particle set corresponds to the normalized score. Particle $P1$ is split, $P2$ is deleted and one putative particle from $P3$ is selected. (B) Particle filtering: The size of the node represents the weight of the particle for the resampling step.

such as reconstruction and segmentation of Markov random fields. However, the algorithm is susceptible to local optima without the aid of relaxation techniques like simulated annealing (Greig et al., 1989).

- DPVI is conceptually similar to nonparametric variational inference (Gershman et al., 2012), which approximates the posterior over a continuous state space using a set of particles convolved with a Gaussian kernel (see Miller et al., 2016, for more sophisticated extensions of this idea). This approach was shown to be effective for probabilistic models that lack the conjugate-exponential structure required for exact mean-field inference. Because nonparametric variational inference approximates continuous densities, it is inapplicable to the discrete problems considered here.

- Frank et al. (2009) used particle approximations within a variational message passing algorithm. The resulting approximation is "local" in the sense that the particles are used to approximate messages passed between nodes in a factor graph, in contrast to the "global" approximation produced by DPVI, which attempts to capture the distribution over the entire set of variables. It is an interesting question for future research to understand what classes of probabilistic models are better approximated using local vs. global approaches.

- Ionides (2008) described a truncated version of importance sampling in which weights falling below some threshold are set to the threshold value. Ionides (2008) showed that truncation reduced sensitivity to the proposal distribution and derived optimal

truncations as a function of the number of samples. This is similar (though not equivalent) to the DPVI setting where latent variables are sampled exhaustively and without replacement.

- Schniter et al. (2008) described an approximate inference algorithm, *fast Bayesian matching pursuit*, which can be viewed as a special case of DPVI applied to Gaussian mixture models.

- In Jones et al. (2005), a *shotgun stochastic search* algorithm is suggested, which proposes local changes to the latent variables with probability proportional to the unnormalized posterior. While this method of evaluating local changes is similar to our coordinate algorithm for the DPVI objective, it is important to note that DPVI is not a stochastic search algorithm (unless $K = 1$): It maintains a collection of particles in order to approximate the posterior distribution, which is important for applications that require a representation of uncertainty.

- Finally, DPVI is closely related to the problem of finding the K most probable latent variable assignments (Flerova et al., 2012; Yanover and Weiss, 2003). We view this problem through the lens of particle approximations, connecting it to both Monte Carlo and variational methods. The techniques developed for finding the K-best assignments could be fruitfully applied to optimizing the DPVI objective function.

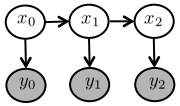
Our experimental strategy is to compare DPVI with popular algorithms that have similar computational complexity, which is why we focus on particle filtering, Gibbs sampling and mean-field approximations. Although mixture mean-field will by definition lead to a better approximation, its computational complexity is considerably greater, which may explain why it is has never achieved the popularity of standard mean-field approaches. Some of the approaches listed above are not applicable to the discrete setting (nonparametric variational inference and fast Bayesian matching pursuit), and some are point estimators instead of true probabilistic approximations (iterated conditional modes and shotgun stochastic search).

6. Experiments

In this section, we compare the performance of DPVI to several widely used approximate inference algorithms, including particle filtering, Gibbs sampling and variational methods. We first present a didactic example to illustrate how DPVI can sometimes succeed where particle filtering fails. We then apply DPVI to four popular but intractable probabilistic models: the Dirichlet process mixture model (DPMM; Antoniak, 1974; Escobar and West, 1995), the infinite HMM (iHMM; Beal et al., 2002; Teh et al., 2006), the infinite relational model (IRM; Kemp et al., 2006) and the Ising model.

(A)

DPVI	Monte Carlo	Mean-Field
$Q(x) = \sum_{k=1}^{K} w^k \prod_{n=1}^{N} \delta[x_n, x_n^k]$	$Q(x) = \frac{1}{K} \sum_{k=1}^{K} \prod_{n=1}^{N} \delta[\hat{x}_n, \hat{x}_n^k]$	$Q(x) = \prod_{n=1}^{N} Q(x_n)$



(B)

DPVI		Monte Carlo	Mean-Field
State Seq. $[x_0, x_1, x_2]$	Weight	State Seq. $[x_0, x_1, x_2]$	Mean-Field Parameters
[0,0,0]	0.15	[1,0,1]	$Q(x_0 = 0) = 0.4$
[0,1,1]	0.03	[1,1,1]	$Q(x_1 = 0) = 0.63$
[1,1,1]	0.32	[1,1,1]	$Q(x_2 = 0) = 0.15$
[1,0,0]	0.15	[1,1,1]	
[1,0,1]	0.35	[1,0,1]	
$Pr(x_0 = 1, x_1 = 0, x_2 = 1\|y) \approx 0.35$		$Pr(x_0 = 1, x_1 = 0, x_2 = 1\|y) \approx 0.4$	$Pr(x_0 = 1, x_1 = 0, x_2 = 1\|y) \approx 0.32$

Figure 2: Comparison of approximate inference schemes. (A) Approximating families for DPVI, Monte Carlo and mean-field. (B) Approximating the posterior probability of a sequence $(x_0 = 1, x_1 = 0, x_2 = 1)$ for the above 3 schemes on a binary HMM. Given the weights for different sequences in DPVI the posterior probability is the weight corresponding to that sequence. For Monte Carlo approximation, the posterior can be approximated from the normalized counts of sampled $(x_0 = 1, x_1 = 0, x_2 = 1)$ sequences. Finally, for the mean-field approximation, we have $Pr(x_0 = 1, x_1 = 0, x_2 = 1|y) = Q(x_0 = 1)Q(x_1 = 0)Q(x_2 = 1)$.

6.1 Didactic Example: Binary HMM

As a didactic example, we use a simple HMM with binary hidden states (x) and observations (y):

$$\begin{aligned} P(x_{n+1} = 0|x_n = 0) &= \alpha_0 \\ P(x_{n+1} = 1|x_n = 1) &= \alpha_1 \\ P(y_n = 0|x_n = 0) &= \beta_0 \\ P(y_n = 1|x_n = 1) &= \beta_1, \end{aligned} \tag{18}$$

with α_0, α_1, β_0, and β_1 all less than 0.5. Constraining the parameters to be less than 0.5 makes some sequences more likely; approximating the posterior using a particle filter (with resampling) may result in capturing only these sequences. We will use this model to illustrate how DPVI differs from particle filtering. Figure 2 compares several inference schemes for this model.

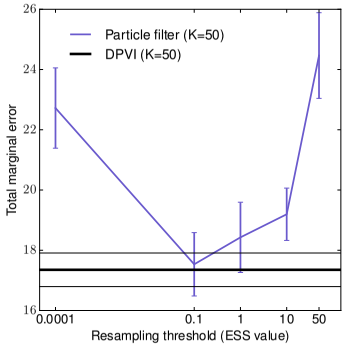


Figure 3: HMM with binary hidden states and observations. Total marginal error computed for a sequence of length 200. For particle filtering the total error for every ESS value is averaged over 5 sequences generated from the HMM; in addition, for each sequence we reran the particle filter 5 times (thus 25 runs total). Note the logarithmic scale of the x-axis. Error bars and the thin black lines correspond to standard error of the mean.

For illustration, we use the following parameters: $\alpha_0 = 0.2$, $\alpha_1 = 0.1$, $\beta_0 = 0.3$, and $\beta_1 = 0.2$. Suppose you observe a sequence generated from this model. For a sufficiently long sequence, a particle filter with resampling will eventually delete most conditionally unlikely particles, due to the fact that there is some probability on each step that any given unlikely particle will be deleted. The particle filter will thus suffer from degeneracy for long sequences. On the other hand, without resampling the approximation will degrade over time because conditionally unlikely particles are never replaced by better particles. For this reason, it is sometimes suggested that resampling only be performed when the effective sample size (ESS) falls below some threshold.

The ESS is calculated as $\text{ESS} = \frac{1}{\sum_{k=1}^{K}(w^k)^2}$. A low ESS means that most of the weight is being placed on a small number of particles, and hence the approximation may be degenerate (although in some cases this may mean that the target distribution is peaky). We evaluated particle filtering with multinomial resampling on synthetic data generated from the HMM described above. Approximation accuracy was measured by using the forward-backward algorithm to compute the hidden state posterior marginals exactly and then comparing these marginals to the particle approximation. Figure 3 shows performance as a function of ESS threshold, demonstrating that there is a fairly narrow range of thresholds for which performance is good. Thus in practice, successful applications of particle filtering may require computationally expensive tuning of this threshold.

In contrast, DPVI achieves performance comparable to the optimal particle filter, but without a tunable threshold. This occurs because DPVI uses an implicit threshold that

is automatically tuned to the problem. Instead of resampling particles, DPVI deletes or propagates particles deterministically based on their relative contribution to the variational bound. We can always incrementally add particles, unlike tuning the threshold. So although K can be viewed as a tuning parameter, we can adapt it with relatively little expense, monotonically increasing the approximation quality in a way that can be easily quantified.

6.2 Dirichlet Process Mixture Model

A DPMM generates data from the following process (Antoniak, 1974; Escobar and West, 1995):

$$G \sim \mathrm{DP}(\alpha, G_0), \qquad \theta_n | G \sim G, \qquad y_n | \theta_n \sim F(\theta_n),$$

where $\alpha \geq 0$ is a concentration parameter and G_0 is a base distribution over the parameter θ_n of the observation distribution $F(y_n|\theta_n)$. Since the Dirichlet process induces clustering of the parameters θ into K distinct values, we can equivalently express this model in terms of a distribution over cluster assignments, $x_n \in \{1, \ldots, C\}$. The distribution over x is given by the Chinese restaurant process (Aldous, 1985):

$$P(x_n = c | x_{1:n-1}) \propto \begin{cases} t_c & \text{if } k \leq C_+ \\ \alpha & \text{if } c = C_+ + 1, \end{cases} \tag{19}$$

where t_c is the number of data points prior to n assigned to cluster c and C_+ is the number of clusters for which $t_c > 0$.

6.2.1 Synthetic Data

We first demonstrate our approach on synthetic data sets drawn from various mixtures of bivariate Gaussians (see Table 1). The model parameters for each simulated data set were chosen to create a spectrum of increasingly overlapping clusters. In particular, we constructed models out of the following building blocks:

$$\mu_1 = \begin{pmatrix} 0.0, & 0.0 \end{pmatrix}, \qquad \mu_2 = \begin{pmatrix} 0.5, & 0.5 \end{pmatrix}$$
$$\Sigma_1 = \begin{pmatrix} 0.25, & 0.0 \\ 0.0, & 0.25 \end{pmatrix}, \qquad \Sigma_2 = \begin{pmatrix} 0.5, & 0.0 \\ 0.0, & 0.5 \end{pmatrix}.$$

For the DPMM, we used a Normal likelihood with a Normal-Inverse-Gamma prior on the component parameters:

$$y_{nd} | x_n = k \sim \mathcal{N}(m_{kd}, \sigma^2_{kd}), \qquad m_{kd} \sim \mathcal{N}(0, \sigma^2_{kd}/\tau), \qquad \sigma^2_{kd} \sim \mathrm{IG}(a, b), \tag{20}$$

where $d \in \{1, 2\}$ indexes observation dimensions and $\mathrm{IG}(a, b)$ denotes the Inverse Gamma distribution with shape a and scale b. We used the following hyperparameter values: $\tau = 25, a = 1, b = 1, \alpha = 0.5$.

Clustering accuracy was measured quantitatively using V-measure (Rosenberg and Hirschberg, 2007). V-measure is an entropy-based measure which explicitly measures how successfully the criteria of homogeneity and completeness have been satisfied. Figure 4 graphically demonstrates the discovery of latent clusters for both DPVI as well as particle filtering. As

Data set	PF ($K = 20$)	MF	DPVI ($K = 1$)	DPVI ($K = 20$)
D1: $[\mu_1, 4\mu_2, 8\mu_2], \Sigma_1$	0.97±0.03	0.80±0.10	0.93±0.05	0.99±0.02
D2: $[\mu_1, 4\mu_2, 8\mu_2], \Sigma_2$	0.89±0.05	0.63±0.02	0.86±0.07	0.90±0.03
D3: $[\mu_1, 2\mu_2, 4\mu_2], \Sigma_1$	0.58±0.12	0.53±0.05	0.51±0.03	0.74±0.16
D4: $[\mu_1, 2\mu_2, 4\mu_2], \Sigma_2$	0.50±0.06	0.29±0.05	0.46±0.05	0.55±0.07
D5: $[\mu_1, \mu_2, 2\mu_2], \Sigma_1$	0.05±0.05	0.28±0.12	0.014±0.02	0.14±0.10
D6: $[\mu_1, \mu_2, 2\mu_2], \Sigma_2$	0.15±0.08	0.12±0.03	0.11±0.06	0.19±0.07

Table 1: Clustering accuracy (V-Measure) for DPMM. Each data set consisted of 200 points drawn from a mixture of 3 Gaussians. For each data set, we repeated the experiment 150 times by iterating through random seeds, reporting mean and standard error. The left column shows the ground truth mean for each cluster and the covariance matrix (shared across clusters). PF denotes particle filtering and MF denotes mean-field.

Number of particles	**DPVI**	**Particle Filtering**
$K = 10$	15.20s	14.71s
$K = 50$	153.75s	184.17s
$K = 100$	567.84s	699.43s

Table 2: Run time comparison for DPMM with synthetic data using data set from Table 1. The run time of DPVI is slightly better than particle filtering for a single pass through the data set.

shown in Table 1, we observe only marginal improvements when the means are farthest from each other and variances are small, as these parameters leads to well-separated clusters in the training set. However, the relative accuracy of DPVI increases considerably when the clusters are overlapping, either due to the fact that the means are close to each other or the variances are high. One factor contributing to greater performance of DPVI might be the diversity term in the variational formulation.

An interesting special case is when $K = 1$. In this case, DPVI is equivalent to the greedy algorithm proposed by Daume (2007) and later extended by Wang and Dunson (2011). In fact, this algorithm was independently proposed in cognitive psychology by Anderson (1991). As shown in Table 1, DPVI with 20 particles outperforms the greedy algorithm, as well as particle filtering with 20 particles. We also demonstrate the run-time performance of DPVI compared to particle filtering in Table 2. It can be seen in our experiments that DPVI tends to be comparable to particle filtering or sometimes more efficient in terms of run-time for the same task, although the theoretical complexity is the same.

6.2.2 Spike Sorting

Spike sorting is an important problem in experimental neuroscience settings where researchers collect large amounts of electrophysiological data from multi-channel tetrodes.

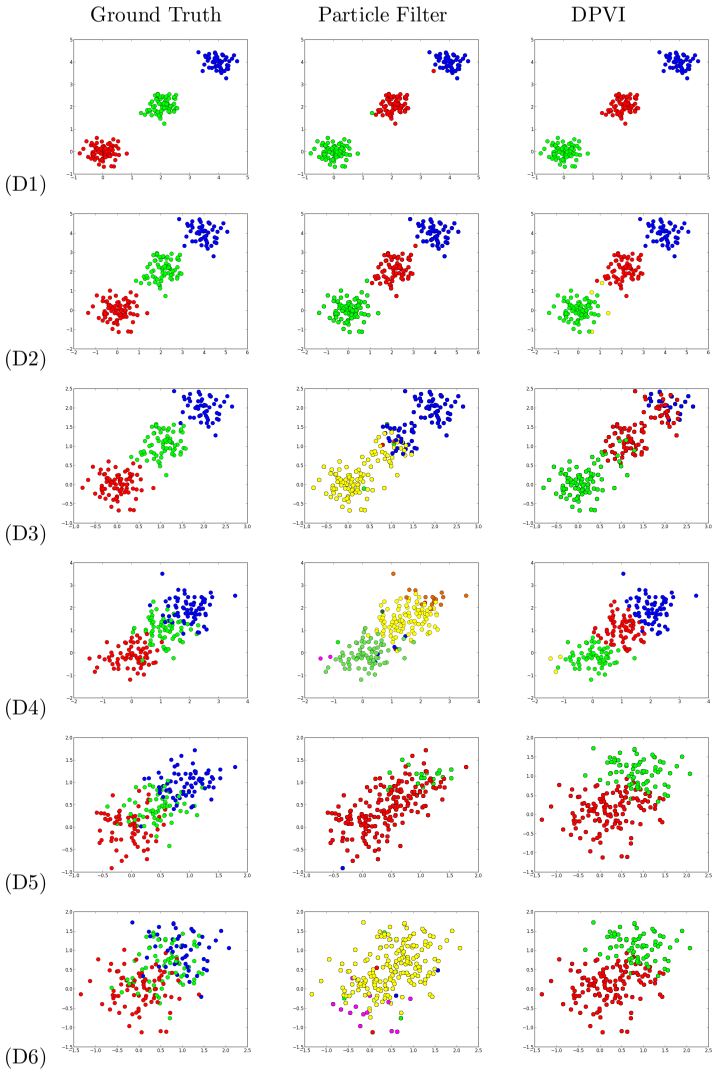


Figure 4: DPMM clustering of synthetic data sets. We treat DPMM as a filtering problem, analyzing one randomly chosen data point at a time. Colors indicate cluster assignments. Each row corresponds to one synthetic data set; refer to Table 1 for corresponding quantitative results. Column 1: Ground truth; Column 2: particle filtering; Column 3: DPVI. The DPVI filter scales similarly to the particle filter but does not underfit as severely.

The goal is to extract from noisy spike recordings attributes such as the number of neurons, and cluster spikes belonging to the same neuron. This problem naturally motivates the use of DPMM, since the number of neurons recorded by a single tetrode is unknown. Previously, Wood and Black (2008) applied the DPMM to spike sorting using particle filtering and Gibbs sampling. Here we show that DPVI can outperform particle filtering, achieving high accuracy even with a small number of particles.

We used data collected from a multiunit recording from a human epileptic patient (Quiroga et al., 2004). The raw spike recordings were preprocessed following the procedure proposed by Quiroga et al. (2004), though we note that our inference algorithm is agnostic to the choice of preprocessing. The original data consist of an input vector with $D = 10$ dimensions and 9196 data points. Following Wood and Black (2008), we used a Normal likelihood with a Normal-Inverse-Wishart prior on the component parameters:

$$\mathbf{y}_n|x_n = k \sim \mathcal{N}(\mathbf{m}_k, \Lambda_k), \qquad \mathbf{m}_k \sim \mathcal{N}(0, \Lambda_k/\tau), \qquad \Lambda_k \sim \mathrm{IW}(\Lambda_0, \nu), \tag{21}$$

where $\mathrm{IW}(\Lambda_0, \nu)$ denotes the Inverse Wishart distribution with degrees of freedom ν and scale matrix Λ_0. We used the following hyperparameter values: $\nu = D + 1, \Lambda_0 = \mathbf{I}, \tau = 0.01, \alpha = 0.1$.

We compared our algorithm to the current best particle filtering baseline, which uses stratified resampling (Wood and Black, 2008; Fearnhead, 2004). The same model parameters were used for all comparisons. Qualitative results, shown in Figure 5B, demonstrate that DPVI is better able to separate the spike waveforms into distinct clusters, despite running DPVI with 10 particles and particle filtering with 100 particles. We also provide quantitative results by calculating the held-out log-likelihood on an independent test set of spike waveforms. The quantitative results (summarized in Table 5C) demonstrate that even with only 10 particles DPVI can outperform particle filtering with 1000 particles.

6.3 Infinite HMM

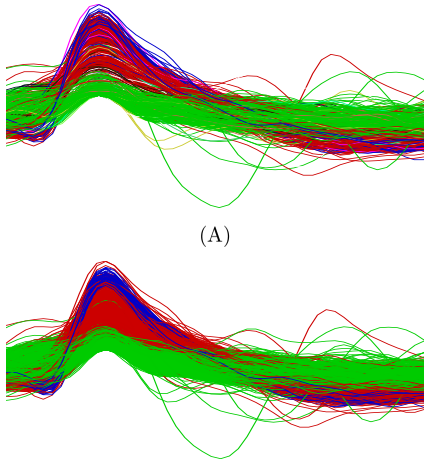
An iHMM generates data from the following process (Teh et al., 2006):

$$G_0 \sim \mathrm{DP}(\gamma, H), \qquad G_k|G_0 \sim \mathrm{DP}(\alpha, G_0),$$
$$x_n|x_{n-1} \sim G_{x_{n-1}}, \qquad \theta_k \sim H, \qquad y_n|x_n \sim F(\theta_{x_n}).$$

Like the DPMM, the iHMM induces a sequence of cluster assignments. The distribution over cluster assignments is given by the Chinese restaurant franchise (Teh et al., 2006). Letting t_{jc} denote the number of times cluster j transitioned to cluster c, x_n is assigned to cluster c with probability proportional to $t_{x_{n-1}c}$, or to a cluster never visited from x_{n-1} ($t_{x_{n-1}c} = 0$) with probability proportional to α. If an unvisited cluster is selected, x_n is assigned to cluster c with probability proportional to $\sum_j t_{jc}$, or to a new cluster (i.e., one never visited from any state, $\sum_j t_{jc} = 0$) with probability proportional to γ.

6.3.1 Synthetic Data

We generated 50 sequences with length 500 from 50 different HMMs, each with 10 hidden and 5 observed states. For the rows of the transition and initial probability matrices of the



(A)

(B)

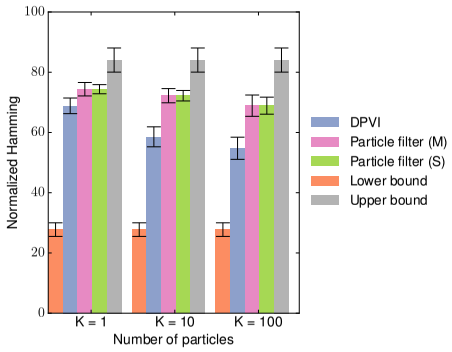
Method	Predictive LL
DPVI ($K = 10$)	-3.2474×10^5 ($\hat{C} = 3$)
DPVI ($K = 100$)	**-1.3888×10^5** ($\hat{\mathbf{C}} = \mathbf{3}$)
PF ($K = 10$)	-1.4771±0.21 × 10^6 ($\hat{C} = 37$)
PF ($K = 100$)	-5.6757±1.14 × 10^5 ($\hat{C} = 13$)
PF ($K = 1000$)	-3.2965×10^5 ($\hat{C} = 5$)

(C)

Method	Run time
DPVI ($K = 10$)	36.20s
DPVI ($K = 50$)	144.6s
DPVI ($K = 100$)	313.8s
PF ($K = 10$)	124s
PF ($K = 50$)	334.2s
PF ($K = 100$)	454.2s

(D)

Figure 5: Spike sorting using the DPMM. Each line is an individual spike waveform, colored according to the inferred cluster. (A) Result using particle filtering with 100 particles and stratified resampling as reported in Wood and Black (2008). (B) Result using DPVI. The same model parameters were used for both particle filtering and DPVI. (C) Spike sorting predictive log-likelihood scores for 200 test points. The best performance is achieved by DPVI with 100 particles. Shown in parentheses is the *maximum a posteriori* number of clusters, $\hat{C}$. (D) Run time comparison for DPMM obtained by using the spike sorting data set. The run time of DPVI is slightly better than particle filtering.



(A)

Method	Runtime (sec)
DPVI ($K = 1$)	1.28
DPVI ($K = 10$)	3.56
DPVI ($K = 100$)	204.42
PF ($K = 1$)	1.14
PF ($K = 10$)	1.92
PF ($K = 100$)	31.99

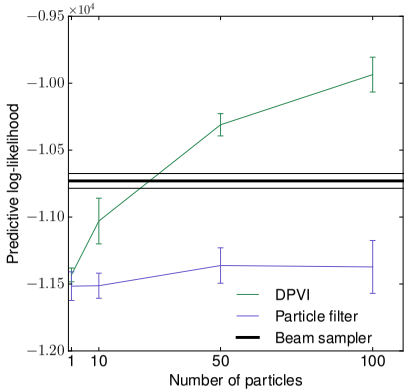
(B)

Figure 6: Infinite HMM on the synthetic data set. (A) Results on 500 synthetic data points generated from an HMM with 10 hidden states. Error is the Hamming distance between the true hidden sequence and the sampled sequence, averaged over 50 data sets. M: multinomial resampling; S: stratified resampling. Lower bound is the expected Hamming distance between data-generating distribution and ground truth. Upper bound is the expected Hamming distance between uniform distribution and ground truth. (B) Run time comparison for the synthetic iHMM data set. We denote particle filtering method by PF.

HMMs we used a symmetric Dirichlet prior with concentration parameter 0.1; for the emission probability matrix, we used a symmetric Dirichlet prior with concentration parameter 10.

Figure 6A illustrates the performance of DPVI and particle filtering (with multinomial and stratified resampling) for varying numbers of particles ($K = 1, 10, 100$). Performance error was quantified by computing the Hamming distance between the true hidden sequence and the sampled sequence. The Munkres algorithm was used to maximize the overlap between the two sequences. The results show that DPVI outperforms particle filtering in all three cases.

When the data consist of long sequences, resampling at every step will produce degeneracy in particle filtering; this tends to result in a smaller number of clusters relative to DPVI. The superior accuracy of DPVI suggests that a larger number of clusters is necessary to capture the latent structure of the data. Not surprisingly, this leads to longer run times (Figurets 6B), but it is important to note that particle filtering and DPVI have comparable per-cluster time complexity.



(A)

Method	Runtime (sec)
DPVI ($K = 1$)	4.73
DPVI ($K = 10$)	41.62
DPVI ($K = 100$)	1685
PF ($K = 1$)	1.64
PF ($K = 10$)	28.08
PF ($K = 100$)	211.66
Beam sampler	260

(B)

Figure 7: Infinite HMM for the text analysis task. (A) Predictive log-likelihood for the Alice in Wonderland data set. (B) Results using the "Alice in Wonderland" data set. The maximum number of inferred clusters is $\hat{C} = 147$ for DPVI ($K = 100$) and $\hat{C} = 21$ for particle filter ($K = 100$). As expected, the per cluster runtime of the two methods are comparable.

6.3.2 Text Analysis

We next analyzed a real-world data set, text taken from the beginning of "Alice in Wonderland", with 31 observation symbols (letters). We used the first 1000 characters for training,

and the subsequent 4000 characters for test. Performance was measured by calculating the predictive log-likelihood. We fixed the hyperparameters α and γ to 1 for both DPVI and the particle filtering.

We ran one pass of DPVI (filtering) and particle filtering over the training sequence. We then sampled 50 data sets from the distribution over the sequences. We truncated the number of states and used the learned transition and emission matrices to compute the predictive log-likelihood of the test sequence. To handle the unobserved emissions in the test sequence we used "add-δ" smoothing with $\delta = 1$. Finally, we averaged over all the 50 data sets.

We also compared DPVI to the beam sampler (Van Gael et al., 2008), a combination of dynamic programming and slice sampling, which was previously applied to this data set. For the beam sampler, we followed the setting of Van Gael et al. (2008). We run the sampler for 10000 iterations and collect a sample of hidden state sequence every 200 iterations. Figure 6B shows the predictive log-likelihood for varying numbers of particles. Even with a small number of particles, DPVI can outperform both particle filtering and the beam sampler.

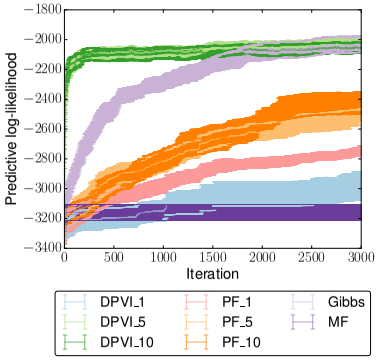
6.3.3 User Behavior Analysis

We analyzed a data set of user behavior in a photo editing software application. The data set contains sequences of edits applied by users to different photos. An iHMM can be utilized for better understanding the high-level tasks of the users. That is, a sequence of multiple edits may be required to perform a high-level task such as cropping or masking a photo. Our data set contains 30000 edits from which we use 1000 data points as a held-out set. There are 23 possible observations (edits) in the data set. We compare DPVI with 1, 5 and 10 particles with particle filtering, Gibbs sampling and mean field inference schemes. For all the inference schemes, we set the hyperparameters α and γ to 1. In every iteration of DPVI and particle filtering, we do a forward filtering-backward smoothing pass over all the sequences of the data set. Our results, shown in Figure 8(B), demonstrate that DPVI with 5 and 10 particles can converge in fewer iterations compared to other reasonable baselines. We illustrate the lower bound (Eq. 10) convergence in Figure 9. The results are computed over 20 runs and for 1, 5 and 10 particles.

It is important to note here that we do not expect DPVI to outperform Gibbs sampling in all scenarios; when computation time is not strongly limited, we expect DPVI and Gibbs to perform similarly. This point applies here as well as to the experiments reported in the sections below. We see DPVI as a useful alternative to Gibbs when the computational budget is low and the required fidelity of the approximation can be satisfied by capturing a few of the posterior modes.

6.4 Infinite Relational Model (IRM)

The IRM (Kemp et al., 2006) is a nonparametric model of relational systems. The model simultaneously discovers the clusters of entities and the relationships between the clusters. A key assumption of the model is that each entity belongs to exactly one cluster.



(A)

Method	Predictive LL
DPVI ($K = 1$)	-2983.24 ± 93.77
DPVI ($K = 5$)	$\mathbf{-2018.02 \pm 9.23}$
DPVI ($K = 10$)	$\mathbf{-2058.75 \pm 33.62}$
PF ($K = 1$)	-2743.60 ± 28.49
PF ($K = 5$)	-2503.95 ± 40.46
PF ($K = 10$)	-2424.01 ± 72.72
Mean field	-3158.69 ± 52.67
Gibbs	$\mathbf{-2030.50 \pm 56.55}$

(B)

Figure 8: Infinite HMM results for the user behavior analysis task. (A) Predictive log-likelihood vs iteration for the user behavior data set. The error bars correspond to the standard error and are computed over 20 runs. In every iteration of DPVI and particle filtering, we do a forward filtering-backward smoothing pass over all the sequences of the data set. (B) Predictive log-likelihood after 3000 iterations of DPVI, Gibbs, particle filter and mean field for the user behavior data set (with 1000 data points as held-out). The best performance is achieved by DPVI with 5-10 particles and also the Gibbs sampler. Run time for each epoch is as follows: 0.726 sec for mean field, 0.593 sec for Gibbs, 4.52 sec for particle filter with 10 particles, and 4.74 sec for DPVI with 10 particles.

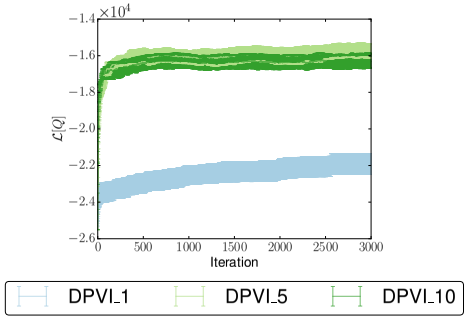


Figure 9: Infinite HMM results for the user behavior analysis task. Lower bound vs iteration for the user behavior data set. The error bars correspond to the standard error and are computed over 20 runs. In every iteration of DPVI, we do a forward filtering-backward smoothing pass over all the sequences of the data set.

Given a relation R involving J types of entities, the goal is to infer a vector of cluster assignments x^j for all the entities of each type $j = 1, \ldots, J$.[2] Assuming the cluster assignments for each type are independent, the joint density of the relation and the cluster assignment vectors can be written as:

$$P(R, x^1, \ldots, x^J) = P(R|x^1, \ldots, x^J) \prod_{j=1}^{J} P(x^j). \tag{22}$$

The cluster assignment vectors are drawn from a CRP(α) prior. Given the cluster assignment vectors, the relations are drawn from a Bernoulli distribution with a parameter η that depends on the clusters involved in that relation. For instance, in a single two-place relation, $\eta(a, b)$ is the probability of having a link between any given pair (i, j) where i is in cluster a and j is in cluster b.

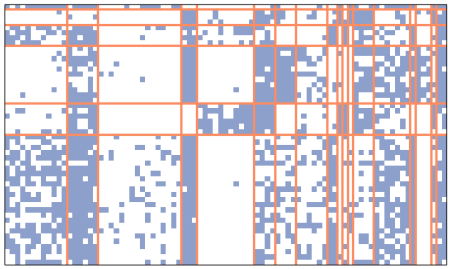
More formally, let us define an M dimensional relation $R : T^{d_1} \times \ldots T^{d_M} \mapsto \{0, 1\}$, over J different types. Each relational value is generated according to:

$$R(i_1, \ldots, i_M)|x^1, \ldots, x^J \sim \text{Bernoulli}(\eta(x^{d_1}_{i_1}, \ldots, x^{d_M}_{i_M})), \tag{23}$$

where d_m denotes the label of the type (i.e., $d_m \in \{1, \cdots, J\}$) and i_m is the entity occupying position m in the relation. Each entry of parameter matrix η is drawn from a Beta(β, β) distribution. By using a conjugate Beta-Bernoulli model, we can analytically marginalize the parameters η (see Kemp et al., 2006), allowing us to directly compute the likelihood of the relational matrix given the cluster assignments, $P(R|x^1, \ldots, x^J)$.

We compared the performance of DPVI with Gibbs sampling, using predictive log-likelihood on held-out data as a performance metric. The "animals" data set analyzed in

2. The IRM model can be defined for multiple relations but for simplicity we only describe the single relation case.



Sample animal clusters:
A1: Hippopotamus, Elephant, Rhinoceros
A2: Seal, Walrus, Dolphins, Blue Whale, Killer Whale, Humpback Whale
A3: Beaver, Otter, Polar Bear

Sample feature clusters:
F1: Hooves, Long neck, Horns
F2: Inactive, Slow, Bulbous Body, Tough Skin
F3: Lives in Fields, Lives in Plains, Grazer
F4: Walks, Quadrupedal, Ground
F5: Fast, Agility, Active, Tail

Figure 10: Co-clustering of animals (rows) and features (columns) after 50 iterations of DPVI with 10 particles in the infinite relational model.

Kemp et al. (2006), was used for this task. This data set (Osherson et al., 1991) is a two type data set $R : T_1 \times T_2 \to \{0, 1\}$ with animals and features as it types; it contains 50 animals and 85 features.

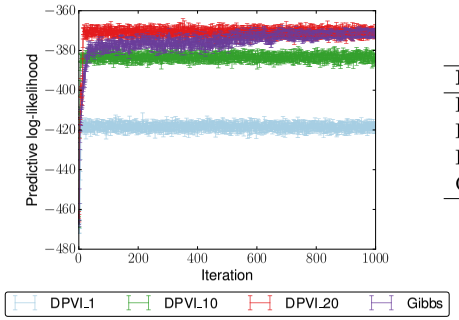
We removed 20% of the relations from the data set and computed the predictive log-likelihood for the held-out data. We ran DPVI with 1, 10 and 20 particles for 1000 iterations. Given the weights of the particles, we computed the weighted log-likelihood. We also ran 20 independent runs of the Gibbs sampler and DPVI for 1000 iterations and computed the average predictive log-likelihood. Every iteration scans all the data points in all the types sequentially. We set the hyperparameters α and β to 1. Figure 10 illustrates the co-clustering discovered by DPVI for the data set, demonstrating intuitively reasonable animal and feature clusters.

The results after 1000 iterations are presented in Table 11B. The best performance is achieved by DPVI with 20 particles. Figure 11A shows the predictive log-likelihood for every iteration of DPVI and Gibbs sampling. DPVI with 10 and 20 particles converge in 11 and 18 iterations, respectively. In terms of computation time per iteration of DPVI versus Gibbs, the only difference for DPVI with one particle and Gibbs is the sorting cost. Hence, for the multiple particle versus multiple runs of Gibbs sampling, the only additional cost is the sorting cost for multiple particles (e.g. 10 or 20). However, this insignificant additional cost is compensated for by a faster convergence rate in our experiments.

6.5 Ising Model

So far, we have been studying inference in directed graphical models, but DPVI can also be applied to undirected graphical models. We illustrate this using the Ising model for binary vectors $x \in \{-1, +1\}^N$:

$$f(x) = \exp\left\{\frac{1}{2} x W x^\top + \theta x^\top\right\}, \tag{24}$$



(A)

Method	Predictive LL
DPVI ($K = 1$)	-418.49 ± 4.24
DPVI ($K = 10$)	-382.46 ± 7.34
DPVI ($K = 20$)	$\mathbf{-370.67 \pm 6.23}$
Gibbs (avg. of 20 runs)	$\mathbf{-371.74 \pm 5.07}$

(B)

Figure 11: Infinite relational model results for the animals data set. (A) Predictive log-likelihood vs iteration for the animals data set. The error bars correspond to the standard error and for both methods are computed over 20 runs. (B) Predictive log-likelihood after 1000 iterations of DPVI and Gibbs (with 50 burnin iterations) for the animals data set (with 20 % held-out). DPVI with 20 particles performs in par with the Gibbs sampler.

where $W \in \mathbb{R}^{N \times N}$ and $\theta \in \mathbb{R}^N$ are fixed parameters. In particular, we study a square lattice ferromagnet, where $W_{ij} = \beta$ for neighboring nodes (0 otherwise) and $\theta_i = 0$ for all nodes. We refer to β as the *coupling strength*. This model has two global modes: when all the nodes are set to 1, and when all the nodes are set to 0. As the coupling strength increases, the probability mass becomes increasingly concentrated at the two modes.

We applied DPVI to this model, varying the number of particles and the coupling strength. At each iteration, we evaluated the change in log probability that would result from setting $\tilde{x}_n^k = 1$:

$$a_n^k = \sum_{n' \in c_n} W_{n,n'} x_{n'}^k + \theta_n, \tag{25}$$

and likewise the change for setting $\tilde{x}_n^k = 1$ can be computing by simply flipping the sign of a_n^k. Ordering these changes, we then took the top K to determine the new particle set.

To quantify performance, we computed the DPVI variational lower bound on the partition function and compared this to the lower bound furnished by the mean-field approximation (see Wainwright and Jordan, 2008). Figure 12A shows the results of this analysis for low coupling strength ($\beta = 0.01$) and high coupling strength ($\beta = 100$). DPVI consistently achieves a better lower bound than mean-field, even with a single particle, and this advantage is especially conspicuous for high coupling strength. Adding more particles improves the results, but more than 3 particles does not appear to confer any additional improvement for high coupling strength. These results illustrate how DPVI is able to capture multimodal target distributions, where mean-field approximations break down (since they cannot effectively handle multimodality).

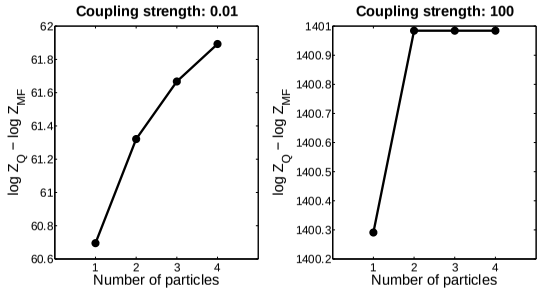


Figure 12: Ising model results. Difference between DPVI and mean-field lower bounds on the partition function. Positive values indicates superior DPVI performance. (A) Low coupling strength; (B) high coupling strength.

To illustrate the performance of DPVI further, we compared several posterior approximations for the Ising model in Figure 13. In addition to the mean-field approximation, we also compared DPVI with two other standard approximations: the Swendsen-Wang Monte Carlo sampler (Swendsen and Wang, 1987) and loopy belief propagation (Murphy et al., 1999). The sampler tended to produce noisy results, whereas mean-field and BP both failed to capture the multimodal structure of the posterior. In contrast, DPVI with two particles perfectly captured the two modes.

7. Conclusions

This paper introduced a particle-based variational method that applies to a broad class of inference problems in discrete models. We described a practical algorithm for optimizing the particle approximation, and showed empirically that it can outperform widely-used Monte Carlo and variational algorithms. The key to the success of this approach is an intentional selection of particles: rather than generating them randomly (as in Monte Carlo algorithms), we deterministically choose a set of unique particles that optimizes the KL divergence between the approximation and the target distribution.

This approach leads to an interesting view on the problem of resampling in sequential Monte Carlo. Resampling is necessary to remove conditionally unlikely particles, but the resulting loss of particle diversity can lead to degeneracy. As we showed in our experiments, tuning an ESS threshold for resampling can improve performance, but requires finding a relatively narrow sweet spot for the threshold. DPVI achieves comparable performance to the best particle filter by using a deterministic strategy for deleting and replacing particles and does not require tuning thresholds. Each particle is guaranteed to be unique and have high probability among all states discovered so far.

DPVI also suggests new hybrids of ideas from Monte Carlo and variational inference. Consider models where all particle extensions cannot be enumerated. In this setting, one could randomly choose particle extensions. To use these particles in a Monte Carlo scheme

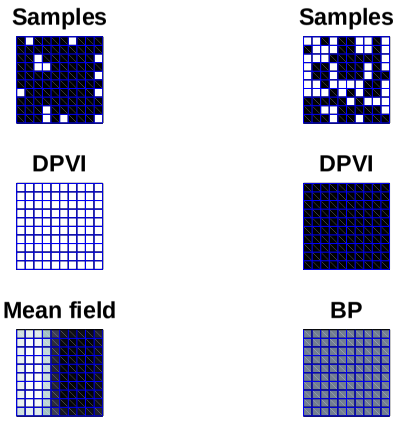


Figure 13: Ising model simulations. Examples of posteriors for the ferromagnetic lattice at low coupling strength. (Top) Two configurations from a Swendsen-Wang sampler. (Middle) Two DPVI particles. (Bottom left) Mean-field expected value. (Bottom right) Loopy belief propagation expected value.

we may need to know the output probability density of the particle extension mechanism. However, if we use the results in DPVI, we just need to be able to score the results under the joint probability distribution. No proposal distribution is needed. This could make it possible to use proposal mechanisms that can be seen to work well empirically but that are difficult to analyze *a priori*.

Although our empirical results are promising, much more empirical and theoretical work is needed to understand the fundamental tradeoffs between variational and Monte Carlo inference. However, the results for DPVI on several problems are promising, and the approach to defining variational approximations may be more broadly applicable. We hope this work encourages others to develop different hybrids of Monte Carlo and variational inference that overcome the limitations of each approach when used in isolation.

Acknowledgments

TDK was generously supported by the Leventhal Fellowship. VKM is supported by the Army Research Office Contract Number 0010363131, Office of Naval Research Award N000141310333, and the DARPA PPAML program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government. The U.S. Government is

authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

References

David Aldous. Exchangeability and related topics. *École d'Été de Probabilités de Saint-Flour XIII1983*, pages 1–198, 1985.

John R Anderson. The adaptive nature of human categorization. *Psychological Review*, 98: 409–429, 1991.

Charles E Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, pages 1152–1174, 1974.

Matthew J Beal, Zoubin Ghahramani, and Carl E Rasmussen. The infinite hidden Markov model. In *Advances in Neural Information Processing Systems*, pages 577–584, 2002.

Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, 48:259–302, 1986.

Alexandre Bouchard-Côté and Michael I Jordan. Optimization of structured mean field objectives. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 67–74. AUAI Press, 2009.

Hal Daume. Fast search for Dirichlet process mixture models. In *International Conference on Artificial Intelligence and Statistics*, pages 83–90, 2007.

Arnaud Doucet, Nando De Freitas, Neil Gordon, et al. *Sequential Monte Carlo Methods in Practice*. Springer New York, 2001.

Michael D Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90:577–588, 1995.

Paul Fearnhead. Particle filters for mixture models with an unknown number of components. *Statistics and Computing*, 14:11–21, 2004.

Natalia Flerova, Emma Rollon, and Rina Dechter. Bucket and mini-bucket schemes for m best solutions over graphical models. In *Graph Structures for Knowledge Representation and Reasoning*, pages 91–118. Springer, 2012.

Andrew Frank, Padhraic Smyth, and Alexander T Ihler. Particle-based variational inference for continuous systems. In *Advances in Neural Information Processing Systems*, 2009.

Samuel Gershman, Matt Hoffman, and David Blei. Nonparametric variational inference. *Proceedings of the 29th International Conference on Machine Learning*, 2012.

Dorothy M Greig, Bruce T Porteous, and Allan H Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 271–279, 1989.

Peter Hall, John T Ormerod, and MP Wand. Theory of Gaussian variational approximation for a Poisson mixed model. *Statistica Sinica*, 21:369–389, 2011.

Edward L Ionides. Truncated importance sampling. *Journal of Computational and Graphical Statistics*, 17:295–311, 2008.

Tommi S Jaakkola and Michael I Jordan. Improving the mean field approximation via the use of mixture distributions. In MI Jordan, editor, *Learning in Graphical Models*, pages 163–173. Springer, 1998.

Beatrix Jones, Carlos Carvalho, Adrian Dobra, Chris Hans, Chris Carter, and Mike West. Experiments in stochastic computation for high-dimensional graphical models. *Statistical Science*, 20:388–400, 2005.

Charles Kemp, Joshua B Tenenbaum, Thomas L Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *AAAI*, volume 3, page 5, 2006.

Neil D Lawrence. *Variational inference in probabilistic models*. PhD thesis, University of Cambridge, 2000.

Kerrie L Mengersen, Richard L Tweedie, et al. Rates of convergence of the Hastings and Metropolis algorithms. *The Annals of Statistics*, 24:101–121, 1996.

Sean P Meyn and Richard L Tweedie. *Markov Chains and Stochastic Stability*. Springer-Verlag, 1993.

Andrew C Miller, Nicholas Foti, and Ryan P Adams. Variational boosting: Iteratively refining posterior approximations. *arXiv preprint arXiv:1611.06585*, 2016.

Kevin P Murphy, Yair Weiss, and Michael I Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 467–475. Morgan Kaufmann Publishers Inc., 1999.

Daniel N Osherson, Joshua Stern, Ormond Wilkie, Michael Stob, and Edward E Smith. Default probability. *Cognitive Science*, 15:251–269, 1991.

Rodrigo Quian Quiroga, Zoltan Nadasdy, and Yoram Ben-Shaul. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. *Neural computation*, 16:1661–1687, 2004.

Christian P Robert and George Casella. *Monte Carlo Statistical Methods*. Springer, 2004.

Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *EMNLP-CoNLL*, volume 7, pages 410–420, 2007.

Philip Schniter, Lee C Potter, and Justin Ziniel. Fast Bayesian matching pursuit. In *Information Theory and Applications Workshop*, pages 326–333. IEEE, 2008.

Robert H Swendsen and Jian-Sheng Wang. Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, 58:86–88, 1987.

Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101:1566–1581, 2006.

Jurgen Van Gael, Yunus Saatci, Yee Whye Teh, and Zoubin Ghahramani. Beam sampling for the infinite hidden Markov model. In *Proceedings of the 25th international conference on Machine learning*, pages 1088–1095. ACM, 2008.

Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1:1–305, 2008.

Bo Wang, DM Titterington, et al. Convergence properties of a general algorithm for calculating variational bayesian estimates for a normal mixture model. *Bayesian Analysis*, 1: 625–650, 2006.

Lianming Wang and David B Dunson. Fast Bayesian inference in Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 20, 2011.

Frank Wood and Michael J Black. A nonparametric Bayesian alternative to spike sorting. *Journal of Neuroscience Methods*, 173:1–12, 2008.

Chen Yanover and Yair Weiss. Finding the M most probable configurations in arbitrary graphical models. In *Advances in Neural Information Processing Systems*, page None, 2003.

Journal of Machine Learning Research 18 (2017) 1-37 Submitted 1/16; Revised 2/17; Published 8/17

A Bayesian Framework for Learning Rule Sets for Interpretable Classification

Tong Wang TONG-WANG@UIOWA.EDU *University of Iowa*
Cynthia Rudin CYNTHIA@CS.DUKE.EDU *Duke University*
Finale Doshi-Velez FINALE@SEAS.HARVARD.EDU *Harvard University*
Yimin Liu LIUYIMIN2000@GMAIL.COM *Edward Jones*
Erica Klampfl EKLAMPFL@FORD.COM *Ford Motor Company*
Perry MacNeille PMACNEIL@FORD.COM *Ford Motor Company*

Editor: Maya Gupta

Abstract

We present a machine learning algorithm for building classifiers that are comprised of a *small* number of *short* rules. These are restricted disjunctive normal form models. An example of a classifier of this form is as follows: *If* X satisfies (condition A AND condition B) OR (condition C) OR $\cdots$, *then* $Y = 1$. Models of this form have the advantage of being interpretable to human experts since they produce a set of rules that concisely describe a specific class. We present two probabilistic models with prior parameters that the user can set to encourage the model to have a desired size and shape, to conform with a domain-specific definition of interpretability. We provide a scalable MAP inference approach and develop theoretical bounds to reduce computation by iteratively pruning the search space. We apply our method (Bayesian Rule Sets – *BRS*) to characterize and predict user behavior with respect to in-vehicle context-aware personalized recommender systems. Our method has a major advantage over classical associative classification methods and decision trees in that it does not greedily grow the model.

Keywords: disjunctive normal form, statistical learning, data mining, association rules, interpretable classifier, Bayesian modeling

1. Introduction

When applying machine learning models to domains such as medical diagnosis and customer behavior analysis, in addition to a reliable decision, one would also like to understand how this decision is generated, and more importantly, what the decision says about the data itself. For classification tasks specifically, a few summarizing and descriptive rules will provide intuition about the data that can directly help domain experts understand the decision process.

Our goal is to construct rule set models that serve this purpose: these models provide predictions and also descriptions of a class, which are reasons for a prediction. Here is an example of a rule set model for predicting whether a customer will accept a coupon for a nearby coffee house, where the coupon is presented by their car's mobile recommendation device:

if a customer (goes to coffee houses $\geq$ once per month AND destination = no urgent place AND passenger $\neq$ kids)
OR (goes to coffee houses $\geq$ once per month AND the time until coupon expires = one day)
then
predict the customer will accept the coupon for a coffee house.

This model makes predictions and provides characteristics of the customers and their contexts that lead to an acceptance of coupons.

Formally, a rule set model consists of a set of *rules*, where each rule is a conjunction of *conditions*. Rule set models predict that an observation is in the positive class when at least one of the rules is satisfied. Otherwise, the observation is classified to be in the negative class. Rule set models that have a small number of conditions are useful as non-black-box (interpretable) machine learning models. Observations come with predictions and also reasons (e.g., this observation is positive *because* it satisfies a particular rule).

This form of model appears in various fields with various names. First, they are sparse disjunctive normal form (DNF) models. In operations research, there is a field called "logical analysis of data" (Boros et al., 2000; Crama et al., 1988), which constructs rule sets based on combinatorics and optimization. Similar models are called "consideration sets" in marketing (see, e.g., Hauser et al., 2010), or "disjunctions of conjunctions." Consideration sets are a way for humans to handle information overload. When confronted with a complicated decision (such as which television to purchase), the theory of consideration sets says that humans tend to narrow down the possibilities of what they are willing to consider into a small disjunction of conjunctions ("only consider TV's that are small AND inexpensive, OR large and with a high display resolution"). In PAC learning, the goal is to learn rule set models when the data are non-noisy, meaning there exists a model within the class that perfectly classifies the data. The field of "rule learning" aims to construct rule set models. We prefer the terms "rule set" models or "sparse DNF" for this work as our goal is to create interpretable models for non-experts, which are not necessarily consideration sets, not other forms of logical data analysis, and pertain to data sets that are potentially noisy, unlike the PAC model.

For many decisions, the space of good predictive models is often large enough to include very simple models (Holte, 1993). This means that there may exist very sparse but accurate rule set models; the challenge is determining how to find them efficiently. Greedy methods used in previous works (Malioutov and Varshney, 2013; Pollack et al., 1988; Friedman and Fisher, 1999; Gaines and Compton, 1995), where rules are added to the model one by one, do not generally produce high-quality sparse models.

We create a Bayesian framework for constructing rule sets, which provides priors for controlling the size and shape of a model. These methods strike a nice balance between accuracy and interpretability through user-adjustable Bayesian prior parameters. To control interpretability, we introduce two types of priors on rule set models, one that uses beta-binomials to model the rule selecting process, called BRS-BetaBinomial, and one that uses Poisson distributions to model rule generation, called BRS-Poisson. Both can be adjusted to suit a domain-specific notion of interpretability; it is well-known that interpretability comes in different forms for different domains (Martens and Baesens, 2010; Martens et al., 2011; Huysmans et al., 2011; Allahyari and Lavesson, 2011; Rüping, 2006; Freitas, 2014).

We provide an approximate inference method that uses association rule mining and a randomized search algorithm (which is a form of stochastic coordinate descent or simulated annealing) to find optimal BRS maximum a posteriori (MAP) models. This approach is motivated by a theoretical bound that allows us to iteratively reduce the size of the computationally hard problem of finding a MAP solution. This bound states that we need only mine rules that are sufficiently fre-

©2017 Tong Wang, Cynthia Rudin, Finale Doshi-Velez, Yimin Liu, Erica Klampfl, and Perry MacNeille.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-003.html.

quent in the database and as the search continues, the bound becomes tighter and further reduces the search space. This greatly reduces computational complexity and allows the problem to be solved in practical settings. The theoretical result takes advantage of the strength of the Bayesian prior.

Our applied interest is to understand user responses to personalized advertisements that are chosen based on user characteristics, the advertisement, and the context. Such systems are called *context-aware recommender systems* (see surveys Adomavicius and Tuzhilin, 2005, 2008; Verbert et al., 2012; Baldauf et al., 2007, and references therein). One major challenge in the design of recommender systems, reviewed in Verbert et al. (2012), is the *interaction challenge*: users typically wish to know *why* certain recommendations were made. Our work addresses precisely this challenge: our models provide rules in data that describe conditions for a recommendation to be accepted.

The main contributions of our paper are as follows:

- We propose a Bayesian approach for learning rule set (DNF) classifiers. This approach incorporates two important aspects of performance, accuracy, and interpretability, and balance between them via user-defined parameters. Because of the foundation in Bayesian methodology, the prior parameters are meaningful; they represent the desired size and shape of the rule set.

- We derive bounds on the support of rules and number of rules in a MAP solution. These bounds are useful in practice because they safely (and drastically) reduce the solution space, improving computational efficiency. The bound on the size of a MAP model guarantees a sparse solution if prior parameters are chosen to favor smaller sets of rules. More importantly, the bounds become tighter as the search proceeds, reducing the search space until the search finishes.

- The simulation studies demonstrate the efficiency and reliability of the search procedure. Losses in accuracy usually result from a misspecified rule representation, not generally from problems with optimization. Separately, using publicly available data sets, we compare rule set models to interpretable and uninterpretable models from other popular classification methods. Our results suggest that BRS models can achieve competitive performance, and are particularly useful when data are generated from a set of underlying rules.

- We study in-vehicle mobile recommender systems. Specifically, we used Amazon Mechanical Turk to collect data about users interacting with a mobile recommendation system. We used rule set models to understand and analyze consumers' behavior and predict their response to different coupons recommended in different contexts. We were able to generate interpretable results that can help domain experts better understand their customers.

The remainder of our paper is structured as follows. In Section 2, we discuss related work. In Section 3, we present Bayesian Rule Set modeling. In Section 4, we introduce an approximate MAP inference method using associative rule mining and randomized search. We also present theoretical bounds on the support and the number of rules in an optimal MAP solution. In Section 5, we show the simulation studies to justify the inference methods. We then report experimental results using publicly available data, including the in-vehicle recommendation system data set we created, to show that BRS models are on-par with black-box models while under strict size restrictions for the purpose of being interpretable.

A shorter version of this work appeared at the International Conference on Data Mining (Wang et al., 2016).

2. Related Work

The models we are studying have different names in different fields: "disjunctions of conjunctions" in marketing, "classification rules" in data mining, "disjunctive normal forms" (DNF) in artificial intelligence, and "logical analysis of data" in operations research. Learning logical models of this form has an extensive history in various settings. The LAD techniques were first applied to binary data in (Crama et al., 1988) and were later extended to non-binary cases (Boros et al., 2000). In parallel, Valiant (1984) showed that DNFs could be learned in polynomial time in the PAC (probably approximately correct - non-noisy) setting, and recent work has improved those bounds via polynomial threshold functions (Klivans and Servedio, 2001) and Fourier analysis (Feldman, 2012). Other work studies the hardness of learning DNF (Feldman, 2006). None of these theoretical approaches considered the practical aspects of building a *sparse* model with realistic noisy data.

In the meantime, the data-mining literature has developed approaches to building logical conjunctive models. Associative classification and rule learning methods (e.g., Ma et al., 1998; Li et al., 2001; Yin and Han, 2003; Chen et al., 2006; Cheng et al., 2007; McCormick et al., 2012; Rudin et al., 2013; Dong et al., 1999; Michalski, 1969; Clark and Niblett, 1989; Frank and Witten, 1998) mine for frequent rules in the data and combine them in a heuristic way, where rules are ranked by an interestingness criteria. Some of these methods, like CBA, CPAR and CMAR (Li et al., 2001; Yin and Han, 2003; Chen et al., 2006; Cheng et al., 2007) still suffer from a huge number of rules. Inductive logic programming (Muggleton and De Raedt, 1994) is similar, in that it mines (potentially complicated) rules and takes the simple union of these rules as the rule set, rather than optimizing the rule set directly. This is a major disadvantage over the type of approach we take here. Another class of approaches aims to construct rule set models by greedily adding the conjunction that explains the most of the remaining data (Malioutov and Varshney, 2013; Pollack et al., 1988; Friedman and Fisher, 1999; Gaines and Compton, 1995; Cohen, 1995). Thus, again, these methods do not directly aim to produce globally optimal conjunctive models.

There are few recent techniques that do aim to fully learn rule set models. Hauser et al. (2010); Goh and Rudin (2014) applied integer programming approaches to solving the full problems but would face a computational challenge for even moderately sized problems. There are several branch-and-bound algorithms that optimize different objectives than ours (e.g., Webb, 1995). Rijnbeek and Kors (2010) proposed an efficient way to exhaustively search for short and accurate decision rules. That work is different than ours in that it does not take a generative approach, and their global objective does not consider model complexity, meaning they do not have the same advantage of reduction to a smaller problem that we have. Methods that aim to globally find decision lists or rule lists can be used to find optimal DNF models, as long as it is possible to restrict all of the labels in the rule list (except the default) to the same value. A rule list where all labels except the default rule are the same value is exactly a DNF formula. We are working on extending the CORELS algorithm (Angelino et al., 2017) to handle DNF.

Note that logical models are generally robust to outliers and naturally handle missing data, with no imputation needed for missing attribute values. These methods can perform comparably with traditional convex optimization-based methods such as support vector machines or lasso (though linear models are not always considered to be interpretable in many domains).

One could extend any method for DNF learning of binary classification problems to multi-class classification. A conventional way to directly extend a binary classifier is to decompose the multi-class problem into a set of two-class problems. Then for each binary classification problem, one could build a separate BRS model, e.g., by the one-against-all method. One would generate different rule sets for each class. For this approach, the issue of overlapping coverage of rule sets for different classes is handled, for instance, by an error correcting output codes (Schapire, 1997).

The main application we consider is in-vehicle context-aware recommender systems. The most similar works to ours include that of Baralis et al. (2011), who present a framework that discovers relationships between user context and services using association rules. Lee et al. (2006) create interpretable context-aware recommendations by using a decision tree model that considers location context, personal context, environmental context and user preferences. However, they did not study some of the most important factors we include, namely contextual information such as the user's destination, relative locations of services along the route, the location of the services with respect to the user's destination, passenger(s) in the vehicle, etc. Our work is related to recommendation systems for in-vehicle context-aware music recommendations (see Baltrunas et al., 2011; Wang et al., 2012), but whether a user will accept a music recommendation does not depend on anything analogous to the location of a business that the user would drive to. The setup of in-vehicle recommendation systems are also different than, for instance, mobile-tourism guides (Noguera et al., 2012; Schwinger et al., 2005; Van Setten et al., 2004; Tung and Soo, 2004) where the user is searching to accept a recommendation, and interacts heavily with the system in order to find an acceptable recommendation. The closest work to ours is probably that of Park et al. (2007) who also consider Bayesian predictive models for context aware recommender systems to restaurants. They also consider demographic and context-based attributes. They did not study advertising, however, which means they did not consider the locations to the coupon's venue, expiration times, etc.

3. Bayesian Rule Sets

We work with standard classification data. The data set S consists of $\{\mathbf{x}_n, y_n\}_{n=1,...N}$, where $y_n \in \{0,1\}$ with N_+ positive and N_- negative, and $\mathbf{x}_n \in \mathcal{X}$. $\mathbf{x}_n$ has J features. Let a represent a rule and $a(\cdot)$ with a corresponding Boolean function

$$a(\cdot) : \mathcal{X} \to \{0,1\}.$$

$a(\mathbf{x})$ indicates if $\mathbf{x}$ satisfies rule a. Let A denote a rule set and let $A(\cdot)$ represent a classifier,

$$A(\mathbf{x}) = \begin{cases} 1 & \exists a \in A, a(\mathbf{x}) = 1 \\ 0 & \text{otherwise.} \end{cases}$$

$\mathbf{x}$ is classified as positive if it satisfies at least one rule in A.

Figure 1 shows an example of a rule set. Each rule is a yellow patch that covers a particular area, and the rule applies to the area it covers. In Figure 1, the white oval in the middle indicates the positive class. Our goal is to find a set of rules A that covers mostly the positive class, but little of the negative class, while keeping A as a small set of short rules.

We present a probabilistic model for selecting rules. Taking a generative approach allows us to flexibly incorporate users' expectations on the "shape" of a rule set through the prior. The user can guide the model toward more domain-interpretable solutions by specifying the desired balance between the size and lengths of rules without committing to any particular value for these parameters.

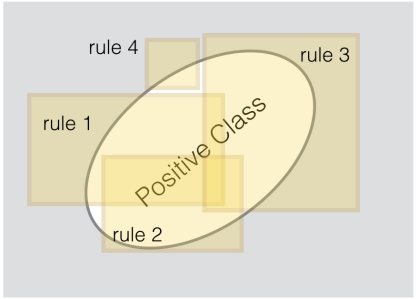


Figure 1: Illustration of Rule Sets. Area covered by any of rules is classified as positive. Area not covered by any rules is classified as negative.

We propose two models for the prior, a Beta-Binomial prior and a Poisson prior, which characterize the interpretability from different perspectives. The likelihood ensures that the model explains the data well and ensures good classification performance. We detail the priors and likelihood below.

3.1 Prior

We propose two priors. In the BRS-BetaBinomial model, the maximum length L of rules is predetermined by a user. Rules of the same length are placed into the same rule *pool*. The model uses L beta priors to control the probabilities of selecting rules from different pools. In a BRS-Poisson model, the "shape" of a rule set, which includes the number of rules and lengths of rules, is decided by drawing from Poisson distributions parameterized with user-defined values. Then the generative process fills it in with conditions by first randomly selecting attributes and then randomly selecting values corresponding to each attribute. We present the two prior models in detail.

3.1.1 BETA-BINOMIAL PRIOR

Rules are drawn randomly from a set $\mathcal{A}$. Assuming the interpretability of a rule is associated with the *length* of a rule (the number of conditions in a rule), the rule space $\mathcal{A}$ is divided into L pools indexed by the lengths, L being the maximum length the user allows.

$$\mathcal{A} = \cup_{l=1}^{L} \mathcal{A}_l. \tag{1}$$

Then, rules are drawn independently in each pool, and we assume in $\mathcal{A}_l$ each rule has probability p_l to be drawn, which we place a beta prior on. For $l \in \{1, ..., L\}$,

$$\text{Select a rule from } \mathcal{A}_l \sim \text{Bernoulli}(p_l) \tag{2}$$

$$p_l \sim \text{Beta}(\alpha_l, \beta_l). \tag{3}$$

In reality, it is not practical to enumerate all possible rules within $\mathcal{A}_l$ when ℓ is even moderately large. However, since we aim for intuitive models, we know that the optimal MAP model will consist of a small number of rules, each of which has sufficiently large support in the data (this

is formalized later within proofs). Our approximate inference technique thus finds and uses high support rules to use within $\mathcal{A}_l$. This effectively makes the beta-binomial model non-parametric.

Let M_l notate the number of rules drawn from $\mathcal{A}_l$, $l \in \{1, ..., L\}$. A BRS model A consists of rules selected from each pool. Then

$$p(A; \{\alpha_l, \beta_l\}_l) = \prod_{l=1}^{L} \int p_l^{M_l}(1-p_l)^{(|\mathcal{A}_l|-M_l)} dp_l$$
$$= \prod_{l=1}^{L} \frac{B(M_l + \alpha_l, |\mathcal{A}_l| - M_l + \beta_l)}{B(\alpha_l, \beta_l)}, \quad (4)$$

where $B(\cdot)$ represents a Beta function. Parameters $\{\alpha_l, \beta_l\}_l$ govern the prior probability of selecting a rule set. To favor smaller models, we would choose α_l, β_l such that $\frac{\alpha_l}{\alpha_l+\beta_l}$ is close to 0. In our experiments, we set $\alpha_l = 1$ for all l.

3.1.2 POISSON PRIOR

We introduce a data independent prior for the BRS model. Let M denote the total number of rules in A and L_m denote the lengths of rules for $m \in \{1, ...M\}$. For each rule, the length needs to be at least 1 and at most the number of all attributes J. According to the model, we first draw M from a Poisson distribution. We then draw L_m from truncated Poisson distributions (which only allow numbers from 1 to J), to decide the "shape" of a rule set. Then, since we know the rule sizes, we can now fill in the rules with conditions. To generate a condition, we first randomly select the attributes then randomly select values corresponding to each attribute. Let $v_{m,k}$ represent the attribute index for the k-th condition in the m-th rule, $v_{m,k} \in \{1, ...J\}$. $K_{v_{m,k}}$ is the total number of values for attribute $v_{m,k}$. Note that a "value" here refers to a category for a categorical attribute, or an interval for a discretized continuous variable. To summarize, a rule set is constructed as follows:

1: Draw the number of rules: $M \sim \text{Poisson}(\lambda)$
2: **for** $m \in \{1, ...M\}$ **do**
3: Draw the number of conditions in m-th rule: $L_m \sim$ Truncated-Poisson$(\eta), L_m \in \{1, ...J\}$
4: Randomly select L_m attributes from J attributes without replacement
5: **for** $k \in \{1, ...L_m\}$ **do**
6: Uniformly at random select a value from $K_{v_{m,k}}$ values corresponding to attribute $v_{m,k}$
7: **end for**
8: **end for**

We define the normalization constant $\omega(\lambda, \eta)$, so the probability of generating a rule set A is

$$p(A; \lambda, \eta) = \frac{1}{\omega(\lambda, \eta)} \text{Poisson}(M; \lambda) \prod_{m=1}^{M} \text{Poisson}(L_m; \eta) \frac{1}{\binom{J}{L_m}} \prod_{k=1}^{L_m} \frac{1}{K_{v_{m,k}}}. \quad (5)$$

3.2 Likelihood

Let $A(\mathbf{x}_n)$ denote the classification outcome for $\mathbf{x}_n$, and let y_n denote the observed outcome. Recall that $A(\mathbf{x}_n) = 1$ if $\mathbf{x}_n$ obeys any of the rules $a \in A$. We introduce likelihood parameter ρ_+ to represent the prior probability that $y_n = 1$ when $A(\mathbf{x}_n) = 1$, and ρ_- to represent the prior probability

that $y_n = 0$ when $A(\mathbf{x}_n) = 0$. Thus:

$$y_n | \mathbf{x}_n, A \sim \begin{cases} \text{Bernoulli}(\rho_+) & A(\mathbf{x}_n) = 1 \\ \text{Bernoulli}(1-\rho_-) & A(\mathbf{x}_n) = 0 \end{cases},$$

with ρ_+ and ρ_- drawn from beta priors:

$$\rho_+ \sim \text{Beta}(\alpha_+, \beta_+) \text{ and } \rho_- \sim \text{Beta}(\alpha_-, \beta_-). \quad (6)$$

Based on the classification outcomes, the training data are divided into four cases: true positives (TP $= \sum_n A(\mathbf{x}_n) y_n$), false positives (FP $= \sum_n A(\mathbf{x}_n)(1-y_n)$), true negatives (TN $= \sum_n (1-A(\mathbf{x}_n))\,(1-y_n)$) and false negatives (FN $= \sum_n (1-A(\mathbf{x}_n))\, y_n$). Then, it can be derived that

$$p(S|A; \alpha_+, \beta_+, \alpha_-, \beta_-) = \int \rho_+^{\text{TP}}(1-\rho_+)^{\text{FP}} \rho_-^{\text{TN}}(1-\rho_-)^{\text{FN}} d\rho_+ d\rho_-$$
$$= \frac{B(\text{TP}+\alpha_+, \text{FP}+\beta_+)}{B(\alpha_+, \beta_+)} \frac{B(\text{TN}+\alpha_-, \text{FN}+\beta_-)}{B(\alpha_-, \beta_-)}. \quad (7)$$

According to (6), $\alpha_+, \beta_+, \alpha_-, \beta_-$ govern the probability that a prediction is correct on training data, which determines the likelihood. To ensure the model achieves the maximum likelihood when all data points are classified correctly, we need $\rho_+ \geq 1-\rho_+$ and $\rho_- \geq 1-\rho_-$. So we choose $\alpha_+, \beta_+, \alpha_-, \beta_-$ such that $\frac{\alpha_+}{\alpha_+ + \beta_+} > 0.5$ and $\frac{\alpha_-}{\alpha_- + \beta_-} > 0.5$. Parameter tuning for the prior parameters will be shown in experiments section. Let H denote a set of parameters for a data S,

$$H = \left\{\alpha_+, \beta_+, \alpha_-, \beta_-, \theta_{\text{prior}}\right\},$$

where θ_{prior} depends on the prior model, our goal is to find an optimal rule set A^* that

$$A^* \in \arg\max_A p(A|S; H). \quad (8)$$

Solving for a MAP model is equivalent to maximizing

$$F(A, S; H) = \log p(A; H) + \log p(S|A; H), \quad (9)$$

where either of the two priors provided above can be used for the first term, and the likelihood is in (7). We write the objective as $F(A)$, the prior probability of selecting A as $p(A)$, and the likelihood as $p(S|A)$, omitting dependence on parameters when appropriate.

4. Approximate MAP Inference

In this section, we describe a procedure for approximately solving for the maximum *a posteriori* (MAP) solution to a BRS model.

Inference in Rule Set modeling is challenging because it involves a search over exponentially many possible sets of rules: since each rule is a conjunction of conditions, the number of rules increases exponentially with the number of conditions, and the number of sets of rules increases exponentially with the number of rules. This is the reason learning a rule set has always been a difficult problem in theory.

Inference becomes easier, however, for our BRS model, due to the computational benefits brought by the Bayesian prior, which can effectively reduce the original problem to a much more manageable size and significantly improve computational efficiency. Below, we detail how to exploit the Bayesian prior to derive useful bounds during any search process. Here we use a stochastic coordinate descent search technique, but the bounds hold regardless of what search technique is used.

4.1 Search Procedure

Given an objective function $F(A)$ over discrete search space of different sets of rules and a temperature schedule function over time steps, $T^{[t]} = T_0^{1-\frac{t}{N_{\text{iter}}}}$, a simulated annealing (Hwang, 1988) procedure is a discrete time, discrete state Markov Chain where at step t, given the current state $A^{[t]}$, the next state $A^{[t+1]}$ is chosen by first randomly selecting a proposal from the neighbors, and the proposal is accepted with probability $\min\left\{1, \exp\left(\frac{F(A^{[t+1]})-F(A^{[t]})}{T^{[t]}}\right)\right\}$, in order to find an optimal rule set A^*. Our version of simulated annealing is similar also to the ϵ-greedy algorithm used in multi-armed bandits (Sutton and Barto, 1998).

Similar to the Gibbs sampling steps used by Letham et al. (2015); Wang and Rudin (2015) for rule-list models, here a "neighboring" solution is a rule set whose edit distance is 1 to the current rule set (one of the rules is different). Our simulated annealing algorithm proposes a new solution via two steps: choosing an action from ADD, CUT and REPLACE, and then choosing a rule to perform the action on.

In the first step, the selection of which rules to ADD, CUT or REPLACE is not chosen uniformly. Instead, the simulated annealing proposes a neighboring solution that aims to do better than the current model with respect to a randomly chosen misclassified point. At iteration $t+1$, an example k is drawn from data points misclassified by the current model $A^{[t]}$. Let $\mathcal{R}_1(\mathbf{x}_k)$ represent a set of rules that $\mathbf{x}_k$ satisfies, and let $\mathcal{R}_0(\mathbf{x}_k)$ represent a set of rules that $\mathbf{x}_k$ does not satisfy. If example k is positive, it means the current rule set fails to cover it so we propose a new model that covers example k, by either adding a new rule from $\mathcal{R}_1(\mathbf{x}_k)$ to $A^{[t]}$, or replacing a rule from $A^{[t]}$ with a rule from $\mathcal{R}_1(\mathbf{x}_k)$. The two actions are chosen with equal probabilities. Similarly, if example k is negative, it means the current rule set covers wrong data, and we then find a neighboring rule set that covers less, by removing or replacing a rule from $A^{[t]} \cap \mathcal{R}_0(\mathbf{x}_k)$, each with probability 0.5.

In the second step, a rule is chosen to perform the action on. To choose the best rule, we evaluate the *precision* of tentative models obtained by performing the selected action on each candidate rule. This precision is:

$$Q(A) = \frac{\sum_i A(\mathbf{x}_i) y_i}{\sum_i A(\mathbf{x}_i)}.$$

Then a choice is made between exploration, which means choosing a random rule (from the ones that improve on the new point), and exploitation, which means choosing the best rule (the one with the highest precision). We denote the probability of exploration as p in our search algorithm. This randomness helps to avoid local minima and helps the Markov Chain to converge to a global minimum. We detail the three actions below.

- ADD
  1. Select a rule z according to $\mathbf{x}_k$
     - With probability p, draw z randomly from $\mathcal{R}_1(\mathbf{x}_k)$. (Explore)
     - With probability $1-p$, $z = \underset{a \in \mathcal{R}_1(\mathbf{x}_k)}{\arg\max}\ Q(A^{[t]} \cup a)$. (Exploit)
  2. Then $A^{[t+1]} \leftarrow A^{[t]} \cup z$.
- CUT
  1. Select a rule z according to $\mathbf{x}_k$
     - With probability p, draw z randomly from $A^{[t]} \cap \mathcal{R}_0(\mathbf{x}_k)$. (Explore)
     - With probability $1-p$, $z = \underset{a \in A^{[t]} \cap \mathcal{R}_0(\mathbf{x}_k)}{\arg\max}\ Q(A^{[t]} \backslash a)$. (Exploit)
  2. Then $A^{[t+1]} \leftarrow A^{[t]} \backslash z$.
- REPLACE: first CUT, then ADD.

To summarize: the proposal strategy uses an ϵ-greedy strategy to determine when to explore and when to exploit, and when it exploits, the algorithm uses a randomized coordinate descent approach by choosing a rule to maximize improvement.

4.2 Iteratively reducing rule space

Another approach to reducing computation is to directly reduce the set of rules. This dramatically reduces the search space, which is the power set of the set of rules. We take advantage of the Bayesian prior and derive a deterministic bound on MAP BRS models that exclude rules that fail to satisfy the bound.

Since the Bayesian prior is constructed to penalize large models, a BRS model tends to contain a small number of rules. As a small number of rules must cover the positive class, each rule in the MAP model must cover enough observations. We define the number of observations that satisfy a rule as the *support* of the rule:

$$\text{supp}(a) = \sum_{n=1}^{N} \mathbf{1}(a(\mathbf{x}_n) = 1). \tag{10}$$

Removing a rule will yield a simpler model (and a smaller prior) but may lower the likelihood. However, we can prove that the loss in likelihood is bounded as a function of the support. For a rule set A and a rule $z \in A$, we use $A_{\backslash z}$ to represent a set that contains all rules from A except z, i.e.,

$$A_{\backslash z} = \{a | a \in A, a \neq z\}.$$

Define

$$\Upsilon = \frac{\beta_-(N_+ + \alpha_+ + \beta_+ - 1)}{(N_- + \alpha_- + \beta_-)(N_+ + \alpha_+ - 1)}.$$

Then the following holds:

Lemma 1 $p(S|A) \geq (\Upsilon)^{\text{supp}(z)} p(S|A_{\backslash z})$.

All proofs in this paper are in Appendix A. This lemma shows that if $\Upsilon \leq 1$, removing a rule with high support lowers the likelihood.

We wish to derive lower bounds on the support of rules in a MAP model. This will allow us to remove (low-support) rules from the search space without any loss in posterior. This will simultaneously yield an upper bound on the maximum number of rules in a MAP model. The bounds will be updated as the search proceeds.

Recall that $A^{[\tau]}$ denotes the rule set at time τ, and let $v^{[t]}$ denote the best solution found until iteration t, i.e.,

$$v^{[t]} = \max_{\tau \leq t} F(A^{[\tau]}).$$

We first look at BRS-BetaBinomial models. In a BRS-BetaBinomial model, rules are selected based on their lengths, so there are separate bounds for rules of different lengths, and we notate the upper bounds of the number of rules at step t as $\{m_l^{[t]}\}_{l=1,\ldots,L}$, where $m_l^{[t]}$ represents the upper bound for the number of rules of length l.

We then introduce some notations that will be used in the theorems. Let $\mathcal{L}^*$ denote the maximum likelihood of data S, which is achieved when all data are classified correctly (this holds when $\alpha_+ > \beta_+$ and $\alpha_- > \beta_-$), i.e. TP $= N_+$, FP $= 0$, TN $= N_-$, and FN $= 0$, giving:

$$\mathcal{L}^* := \max_A p(S|A) = \frac{B(N_+ + \alpha_+, \beta_+)}{B(\alpha_+, \beta_+)} \frac{B(N_- + \alpha_-, \beta_-)}{B(\alpha_-, \beta_-)}. \tag{11}$$

For a BRS-BetaBinomial model, the following is true.

Theorem 1 *Take a BRS-BetaBinomial model with parameters*

$$H = \{L, \alpha_+, \beta_+, \alpha_-, \beta_-, \{\mathcal{A}_l, \alpha_l, \beta_l\}_{l=1,..L}\},$$

where $L, \alpha_+, \beta_+, \alpha_-, \beta_-, \{\alpha_l, \beta_l\}_{l=1,...L} \in \mathbb{N}^+$, $\alpha_+ > \beta_+, \alpha_- > \beta_-$ *and* $\alpha_l < \beta_l$. *Define* $A^* \in \arg\max_A F(A)$ *and* $M^* = |A^*|$. *If* $\Upsilon \leq 1$, *we have:* $\forall a \in A^*$, *supp*$(a) \geq C^{[t]}$ *and*

$$C^{[t]} = \left\lceil \frac{\log \max_l \left(\frac{|\mathcal{A}_l| - m_l^{[t-1]} + \beta_l}{m_l^{[t-1]} - 1 + \alpha_l} \right)}{\log \Upsilon} \right\rceil, \tag{12}$$

where

$$m_l^{[t]} = \left\lfloor \frac{\log \mathcal{L}^* + \log p(\emptyset) - v^{[t]}}{\log \frac{|\mathcal{A}_l| + \beta_l - 1}{\alpha_l + m_l^{[t-1]} - 1}} \right\rfloor,$$

and $m_l^{[0]} = \left\lfloor \frac{\log \mathcal{L}^* - \log p(S|\emptyset)}{\log \frac{|\mathcal{A}_l| + \beta_l - 1}{\alpha_l + |\mathcal{A}_l| - 1}} \right\rfloor$ *and* $v^{[0]} = F(\emptyset)$. *The size of* A^* *is upper bounded by*

$$M^* \leq \sum_{l=1}^{L} m_l^{[t]}.$$

In the equation for $m_l^{[t]}$, $p(\emptyset)$ is the prior of an empty set, which is also the maximum prior for a rule set model, which occurs when we set $\alpha_l = 1$ for all l. Since $\mathcal{L}^*$ and $p(\emptyset)$ upper bound the likelihood and prior, respectively, $\log \mathcal{L}^* + \log p(\emptyset)$ upper bounds the optimal objective value. The difference between this value and $v^{[t]}$, the numerator, represents the room for improvement from the current solution $v^{[t]}$. The smaller the difference, the smaller the $m_l^{[t]}$, and thus the larger the minimum support bound. In the extreme case, when an empty set is the optimal solution, i.e., the likelihood and the prior achieve the upper bound $\mathcal{L}^*$ and $p(\emptyset)$ at the same time, and the optimal solution is discovered at time t, then the numerator becomes zero and $m_l^{[t]} = 0$ which precisely refers to an empty set. Additionally, parameters α_l and β_l jointly govern the probability of selecting rules of length l, according to formula (3). When α_l is set to be small and β_l is set to be large, p_l is small since $\mathbb{E}(p_l) = \frac{\alpha_l}{\alpha_l + \beta_l}$. Therefore the resulting $m_l^{[t]}$ is small, guaranteeing that the algorithm will choose a smaller number of rules overall.

Specifically, at step 0 when there is no $m_l^{[t-1]}$, we set $m_l^{[t-1]} = |\mathcal{A}_l|$, yielding

$$m_l^{[0]} = \left\lfloor \frac{\log \mathcal{L}^* - \log p(S|\emptyset)}{\log \frac{|\mathcal{A}_l| + \beta_l - 1}{\alpha_l + |\mathcal{A}_l| - 1}} \right\rfloor, \tag{13}$$

which is the upper bound we obtain before we start learning for A^*. This bound uses an empty set as a comparison model, which is a reasonable benchmark since the Bayesian prior favors smaller models. (It is possible, for instance, that the empty model is actually close to optimal, depending on the prior.) As $\log p(S|\emptyset)$ increases, $m_l^{[0]}$ becomes smaller, which means the model's maximum possible size is smaller. Intuitively, if an empty set already achieves high likelihood, adding rules will often hurt the prior term and achieve little gain on the likelihood.

Using this upper bound, we get a minimum support which can be used to reduce the search space before simulated annealing begins. As $m^{[t]}$ decreases, the minimum support increases since the rules need to jointly cover positive examples. Also, if fewer rules are needed it means each rule needs to cover more examples.

Similarly, for BRS-Poisson models, we have:

Theorem 2 *Take a BRS-Poisson model with parameters*

$$H = \{L, \alpha_+, \beta_+, \alpha_-, \beta_-, \lambda, \eta\},$$

where $L, \alpha_+, \beta_+, \alpha_-, \beta_- \in \mathbb{N}^+$, $\alpha_+ > \beta_+, \alpha_- > \beta_-$. *Define* $A^* \in \arg\max_A F(A)$ *and* $M^* = |A^*|$. *If* $\Upsilon \leq 1$, *we have:* $\forall a \in A^*$, *supp*$(a) \geq C^{[t]}$ *and*

$$C^{[t]} = \left\lceil \frac{\log \frac{x}{M^{[t]}}}{\log \Upsilon} \right\rceil, \tag{14}$$

where

$$x = \frac{e^{-\eta} \lambda \max\left\{ \left(\frac{\eta}{2}\right) \Gamma(J+1), \left(\frac{\eta}{2}\right)^J \right\}}{\Gamma(J+1)},$$

and

$$M^{[t]} = \left\lfloor \frac{\log \frac{(\lambda+1)^\lambda}{\Gamma(\lambda+1)}}{\log \frac{\lambda+1}{x}} + \frac{\log \mathcal{L}^* + \log p(\emptyset) - v^{[t]}}{\log \frac{\lambda+1}{x}} \right\rfloor.$$

The size of A^* is upper bounded by

$$M^* \leq M^{[t]}.$$

Similar to Theorem 1, $\log \mathcal{L}^* + \log p(\emptyset)$ upper bounds the optimal objective value and $\log \mathcal{L}^* + \log p(\emptyset) - v^{[t]}$ is an upper bound on the room for improvement from the current solution $v^{[t]}$. The smaller the bounds becomes, the larger is the minimum support, thus reducing the rule space iteratively whenever a new maximum solution is discovered.

The algorithm, which applies the bounds above, is given as Algorithm 1.

Algorithm 1 Inference algorithm.

procedure SIMULATED ANNEALING(N_{iter})
  $\mathcal{A} \leftarrow$ FP-Growth(S)
  $A^{[0]} \leftarrow$ a randomly generated rule set
  $A_{\text{max}} \leftarrow A^{[0]}$
  for $t = 0 \rightarrow N_{\text{iter}}$ **do**
    $\mathcal{A} \leftarrow \{a \in \mathcal{A}, \text{supp}(a) \geq C^{[t]}\}$ ($C^{[t]}$ is from (12) or (14), depending on the model choice)
    $S_\epsilon \leftarrow$ misclassified examples by $A^{[t]}$ (Find misclassified examples)
    $(\mathbf{x}_k, y_k) \leftarrow$ a random example drawn from S_ϵ
    if $y_k = 1$ **then**

$$\text{action} \leftarrow \begin{cases} \text{ADD, with probability } 0.5 \\ \text{REPLACE, with probability } 0.5 \end{cases}$$

    else

$$\text{action} \leftarrow \begin{cases} \text{CUT, with probability } 0.5 \\ \text{REPLACE, with probability } 0.5 \end{cases}$$

    $A^{[t+1]} \leftarrow \text{action}(\mathcal{A}, A^{[t]}, \mathbf{x}_k)$
    $A_{\text{max}} \leftarrow \arg\max(F(A_{\text{max}}), F(A^{[t+1]}))$ (Check for improved optimal solution)
    $\alpha = \min\left\{1, \exp\left(\frac{F(A^{[t+1]}) - F(A^{[t]})}{T^{[t]}}\right)\right\}$ (Probability of an annealing move)

$$A^{[t+1]} \leftarrow \begin{cases} A^{[t+1]}, \text{ with probability } \alpha \\ A^{[t]}, \text{ with probability } 1-\alpha \end{cases}$$

  end for
  return A_{max}
end procedure

4.3 Rule Mining

We mine a set of candidate rules $\mathcal{A}$ before running the search algorithm and only search within $\mathcal{A}$ to create an approximate but more computationally efficient inference algorithm. For categorical attributes, we consider both positive associations (e.g., x_j='blue') and negative associations (x_j='not green') as conditions. (The importance of negative conditions is stressed, for instance, by Brin et al., 1997; Wu et al., 2002; Teng et al., 2002). For numerical attributes, we create a set of binary variables for each numerical attribute, by comparing it with a set of thresholds (usually quantiles), and add their negations as separate attributes. For example, we discretize age with three thresholds, creating six binary variables in total, i.e., age≥25, age≥50, age≥ 75 and the negations for each of

them, i.e., age<25, age<50 and age<75. We then mine for frequent rules within the set of positive observations S^+. To do this, we use the FP-growth algorithm (Borgelt, 2005), which can in practice be replaced with any desired frequent rule-mining method.

Sometimes for large data sets with many features, even when we restrict the length of rules and the minimum support, the number of generated rules could still be too large to handle, as it can be exponential in the number of attributes. For example, for one of the advertisement data sets, a million rules are generated when the minimum support is 5% and the maximum length is 3. In that case, we use a second criterion to screen for the most potentially useful M_0 rules, where M_0 is user-defined and depends on the user's computational capacity. We first filter out rules on the lower right plane of ROC space, i.e., their false positive rate is greater than true positive rate. Then we use *information gain* to screen rules, similarly to other works (Chen et al., 2006; Cheng et al., 2007). For a rule a, the information gain is

$$\text{InfoGain}(S|a) = H(S) - H(S|a), \tag{15}$$

where $H(S)$ is the entropy of the data and $H(S|a)$ is the conditional entropy of data that split on rule a. Given a data set S, entropy $H(S)$ is constant; therefore our screening technique chooses the M_0 rules that have the smallest $H(S|a)$.

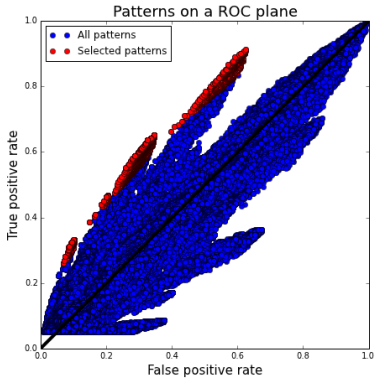


Figure 2: All rules and selected rules on a ROC plane

We illustrate the effect of screening on one of our mobile advertisement data sets. We mined all rules with minimum support 5% and maximum length 3. For each rule, we computed its true positive rate and false positive rate on the training data and plotted it as a dot in Figure 2. The top 5000 rules with highest information gains are colored in red, and the rest are in blue. As shown in the figure, information gain indeed selected good rules as they are closer to the upper left corner in ROC space. For many applications, this screening technique is not needed, and we can simply use the entire set of pre-mined rules that have support above the required threshold for the optimal solution provided in the theorems above.

5. Simulation Studies

We present simulation studies to show the interpretability and accuracy of our model and the efficiency of the simulated annealing procedure. We first demonstrate the efficiency of the simulated annealing algorithm by searching within candidate rules for a "true rule set" that generates the data. Simulations show that our algorithm can recover the true rule set with high probability within a small number of iterations despite the large search space. We then designed the second set of simulations to study the trade-off between accuracy and simplicity. BRS models may lose accuracy to gain model simplicity when the number of attributes is large. Combining the two simulations studies, we were able to hypothesize that possible losses in accuracy are often due to the choice of model representation as a rule set, rather than simulated annealing. In this section, we choose the BRS-BetaBinomial model for the experiments. BRS-Poisson results would be similar; the only difference is the choice of prior.

5.1 Simulation Study 1: Efficiency of Simulated Annealing

In the first simulation study, we would like to test if simulated annealing is able to recover a true rule set given that these rules are in a candidate rule set, and we would like to know how efficiently simulated annealing finds them. Thus we omit the step of rule mining for this experiment and directly work with generated candidate rules, which we know in this case contains a true rule set.

Let there be N observations, $\{\mathbf{x}_n\}_{n=1,...,N}$ and a collection of M candidate rules, $\{a_m\}_{m=1,...M}$. We can construct an $N \times M$ binary matrix where the entry in the n-th row and m-th column is the Boolean function $a_m(x_n)$ that represents whether the n-th observation satisfies the m-th rule. Let the true rule set be a subset of the candidate rules, $A^* \subset \{a_m\}_{m=1,...M}$. The labels $\{y_n\}_{n=1,...N}$ satisfy

$$y_n = \begin{cases} 1 & \exists a \in A^*, a(\mathbf{x}_n) = 1 \\ 0 & \text{otherwise.} \end{cases} \tag{16}$$

We ran experiments with varying N and M. Each time, we first generated an $N \times M$ binary matrix by setting the entries to 1 with probability 0.035, and then selected 20 columns to form A^*. (Values 0.035 and 20 are chosen so that the positive class contains roughly 50% of the observations. We also ensured that the 20 rules do not include each other.) Then $\{y_n\}_{n=1,...N}$ were generated as above. We randomly assigned lengths from 1 to 5 to $\{a_m\}_{m=1,...M}$. Finally, we ran Algorithm 1 on the data set. The prior parameters were set as below for all simulations:

$$\alpha_l = 1, \beta_l = |\mathcal{A}_l| \text{ for } l = 1, ..., 3,$$
$$\alpha_+ = \alpha_- = 1000, \beta_+ = \beta_- = 1.$$

For each pair of N and M, we repeated the experiment 100 times and recorded intermediate output models at different iterations. Since multiple different rule sets can cover the same points, we compared labels generated from the true rule set and the recovered rule set: a label is 1 if the instance satisfies the rule set and 0 otherwise. We recorded the training error for each intermediate output model. The mean and standard deviation of training error rate are plot Figure 3, along with run time to convergence in seconds on the right figure.

Comparing the three sets of experiments, we notice that different sizes of the binary matrix led to differing convergence times due to different computation cost of handling the matrices, yet the three error rate curves in Figure 3 almost completely overlap. Neither the size of the data set N nor

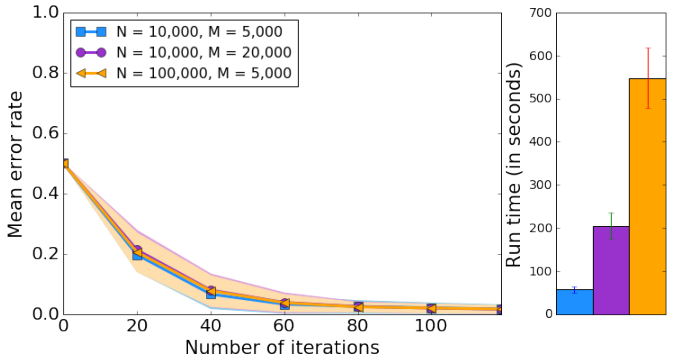


Figure 3: Convergence of mean error rate with number of iterations for data with varying N and M and running time (in seconds), obtained by the BRS-BetaBinomial.

the size of the rule space M affected the search. This is because the strategy based on curiosity and bounds chose the best solution efficiently, regardless of the size of data or the rule space.

More importantly, this study shows that simulated annealing is able to recover a true rule set A^*, or an equivalent rule set with optimal performance, with high probability, with few iterations.

5.2 Simulation Study 2: Accuracy-Interpretability Trade-off

We observe from the first simulation study that simulated annealing does not tend to cause losses in accuracy. In the second simulation study, we would like to identify whether the rule representation causes losses in accuracy. In this study, we used the rule mining step and directly worked with data $\{\mathbf{x}_n\}_{n=1,...,N}$. Without loss of generality, we assume $\mathbf{x}_n \in \{0,1\}^J$.

In each experiment, we first constructed an $N \times J$ binary matrix $\{\mathbf{x}_n\}_{n=1,...N}$ by setting the entries to 1 with probability 0.5, then randomly generated 20 rules from $\{\mathbf{x}_n\}_{n=1,...,N}$ to form A^* and finally generated labels $\{y_n\}_{n=1,...,N}$ following formula (16). We checked the balance of the data before pursuing the experiment. The data set was used only if the positive class was within $[30\%, 70\%]$ of the total data, and otherwise regenerated.

To constrain computational complexity, we set the maximum length of rules to be 7 and then selected the top 5000 rules with highest information gain. Then we performed Algorithm 1 on these candidate rules to generate a BRS model $\tilde{A}$. Ideally, $\tilde{A}$ should be simpler than A^*, with a possible loss in its accuracy as a trade-off.

To study the influence of the dimensions of data on accuracy and simplicity, we varied N and J and repeated each experiment 100 times. Figure 4 shows the accuracy and number of rules in $\tilde{A}$ in each experiment. The number of rules in $\tilde{A}$ was almost always less than the number of rules in A^* (which was 20). On average, $\tilde{A}$ contained 12 rules, which was 60% of the size of A^*, slightly compromising accuracy. BRS needed to compromise more when J was large since it became harder to maintain the same level of simplicity.

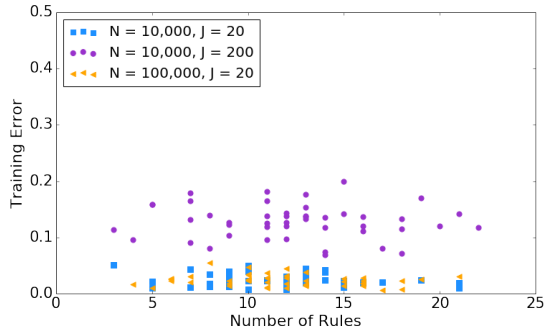


Figure 4: Accuracy and complexity of output BRS models for data with varying N and J, obtained from the BRS-BetaBinomial.

We show in Figure 5 the average training error at different iterations to illustrate the convergence rate. Simulated annealing took less than 50 iterations to converge. Note that it took fewer iterations than simulation study 1 since the error rate curve started from less than 0.5, due to rule pre-screening using information gain.

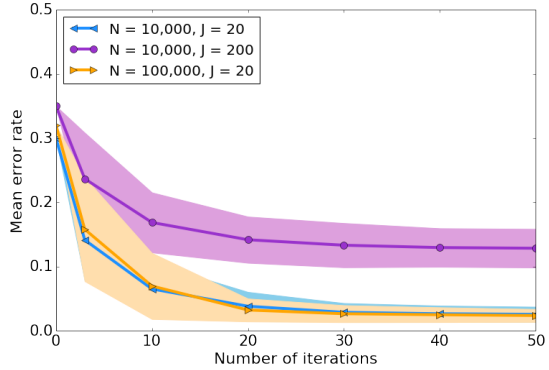


Figure 5: Convergence of mean error rate with varying N and J, obtained by BRS-BetaBinomial.

Figure 4 and Figure 5 both indicate that increasing the number of observations did not affect the performance of the model. BRS is more sensitive to the number of attributes. The larger J is, the less informative a short rule becomes to describe and distinguish a class in the data set, and the more BRS has to trade-off, in order to keep simple rules within the model.

To summarize: when we model using fewer rules, it benefits interpretability but can lose information. Rules with a constraint on the maximum length tend to be less accurate when dealing with

high dimensional feature spaces. The loss increases with the number of features. This is the price paid for using an interpretable model. There is little price paid for using our optimization method.

6. Experiments

Our main application of interest is mobile advertising. We collected a large advertisement data set using Amazon Mechanical Turk that we made publicly available (Wang et al., 2015), and we tested BRS on other publicly available data sets. We show that our model can generate interpretable results that are easy for humans to understand, without losing too much (if any) predictive accuracy. In situations where the ground truth consists of deterministic rules (similarly to the simulation study), our method tends to perform better than other popular machine learning techniques.

6.1 UCI Data Sets

We test BRS on ten data sets from the UCI machine learning repository (Bache and Lichman, 2013) with different types of attributes; four of them contain categorical attributes, three of them contain numerical attributes and three of them contain both. We performed 5-fold cross validation on each data set. Since our goal was to produce models that are accurate and interpretable, we set the maximum length of rules to be 3 for all experiments. We compared with interpretable algorithms: Lasso (without interaction terms to preserve interpretability), decision trees (C4.5 and CART), and inductive rule learner RIPPER (Cohen, 1995), which produces a rule list greedily and anneals locally afterward. In addition to interpretable models, we also compare BRS with uninterpretable models from random forests and SVM to set a benchmark for expected accuracy on each data set. We used implementations of baseline methods from R packages, and tuned model parameters via 5-fold nested cross validation.

Table 1: Accuracy of each comparing algorithm (mean ± std) on ten UCI data sets.

	Data Type	**BRS1**	**BRS2**	Lasso	C4.5	CART	RIPPER	Random Forest	SVM
connect-4	Categorical	.76±.01	.75±.01	.70±.00	.83±.00	.69±.01	———	.86±.00	.82±.00
mushroom		1.00±.00	1.00±.00	.96±.00	1.00±.00	.99±.00	.99±.00	1.00±.00	.93±.00
tic-tac-toe		1.00±.00	1.00±.00	.71±.02	.92±.03	.93±.02	.98±.01	.99±.00	.99±.00
chess		.89±.01	.89±.01	.83±.01	.97±.00	.92±.04	.91±.01	.91±.00	.99±.00
magic	Numerical	.80±.02	.79±.01	.76±.01	.79±.00	.77±.00	.78±.00	.78±.01	.78±.01
banknote		.97±.01	.97±.01	1.00±.00	.90±.01	.90±.02	.91±.01	.91±.01	1.00±.00
indian-diabetes		.72±.03	.73±.03	.67±.01	.66±.03	.67±.01	.67±.02	.75±.03	.69±.01
adult	Mixed	.83±.00	.84±.01	.82±.02	.83±.00	.82±.01	.83±.00	.84±.00	.84±.00
bank-marketing		.91±.00	.91±.00	.90±.00	.90±.00	.90±.00	.90±.00	.90±.00	.90±.00
heart		.85±.03	.84±.03	.85±.04	.76±.06	.77±.06	.78±.04	.81±.06	.86±.06
Rank of accuracy		2.6±1.7	2.6±1.8	5.4±2.8	4.3±2. 9	5.6±1.8	4.8±1.5	2.7±1.7	2.8±2.2

Table 1 displays the means and standard deviations of the test accuracy for all algorithms (the lead author's laptop ran out of memory when applying RIPPER to the connect-4 data set),

Table 2: Runtime (in seconds) of each comparing algorithm (mean ± std) on ten UCI data sets.

	BRS1	**BRS2**	Lasso	C4.5	CART	RIPPER	Random Forest	SVM
connect-4	288.1±22.2	279.1±20.9	0.9±0.0	30.4±0.6	20.5±0.4	———	746.6±14.9	1427.8±35.1
mushroom	8.8±1.3	8.18±1.3	0.1±0.0	0.5±0.0	0.6±0.0	1.2±0.1	15.6±0.8	23.5±1.2
tic-tac-toe	1.0±0.0	1.0±0.0	0.0±0.0	0.1±0.0	0.1±0.0	0.3±0.0	1.3±.0.0	0.2±0.0
chess	44.8±8.2	41.6±8.2	0.1±0.0	1.3±0.4	0.3±0.1	50.6±18.0	47.5±1.9	1193.3±79.6
magic	31.3±4.4	29.3±4.4	0.1±0.0	1.4±0.1	2.3±0.2	8.8±1.7	43.5±1.3	139.7±3.2
banknote	1.0±0.1	0.9±0.1	0.0±0.0	0.1±0.0	0.1±0.0	0.2±0.0	1.1±0.0	0.1±0.0
indian-diabetes	0.6±0.1	0.6±0.1	0.0±0.0	0.1±0.0	0.1±0.0	0.2±0.0	0.7±0.0	0.2±0.0
adult	50.2±6.2	47.1±6.2	0.4±0.0	9.8±0.3	10.7±0.4	66.0±8.8	344.4±7.7	174.6±4.7
bank-marketing	55.2±2.7	50.8±2.7	0.4±0.0	11.6±0.6	7.6±0.3	23.9±4.6	320.0±9.8	395.1±28.1
heart	0.5±0.1	0.5±0.1	0.0±0.0	0.1±0.0	0.0±0.0	0.2±0.1	0.3±0.0	0.0±0.0

and the rank of their average performance. Table 2 displays the runtime. BRS1 represents BRS-BetaBinomial and BRS2 represents BRS-Poisson. While BRS models were under strict restriction for interpretability purposes ($L = 3$), BRS's performance surpassed that of the other interpretable methods and was on par with uninterpretable models. This is because, firstly, BRS has Bayesian priors that favor rules with a large support (Theorem 1, 2) which naturally avoids overfitting of the data; and secondly, BRS optimizes a global objective while decision trees and RIPPER rely on local greedy splitting and pruning methods, and interpretable versions of Lasso are linear in the base features. Decision trees and RIPPER are local optimization methods and tend not to have the same level of performance as globally optimal algorithms, such as SVM, Random Forests, BRS, etc. The class of rule set models and decision tree models are the same: both create regions within input space consisting of a conjunction of conditions. The difference is the choice of optimization method: global vs. local.

For the tic-tac-toe data set, the positive class can be classified using exactly eight rules. BRS has the capability to exactly learn these conditions, whereas the greedy splitting and pruning methods that are pervasive throughout the data mining literature (e.g., CART, C4.5) and convexified approximate methods (e.g., SVM) have difficulty with this. Both linear models and tree models exist that achieve perfect accuracy, but the heuristic splitting/pruning and convexification of the methods we compared with prevented these perfect solutions from being found.

BRS achieves accuracies competitive to uninterpretable models while requiring a much shorter runtime, which grows slowly with the size of the data, unlike Random forest and SVM. This indicates that BRS can reliably produce a good model within a reasonable time for large data sets.

6.1.1 PARAMETER TUNING

A MAP solution maximizes the sum of logs of prior and likelihood. The scale of the prior and likelihood directly determines how the model trades off between fitting the data and achieving the desired sparsity level. Again, we choose the BRS-BetaBinomial for this demonstration. The Bayesian model uses parameters $\{\alpha_l, \beta_l\}_{l=1}^{L}$ to govern the prior for selecting rules and parameters $\alpha_+, \beta_+, \alpha_-, \beta_-$ to govern the likelihood of data. We study how sensitive the results are to different

parameters and how to tune the parameters to get the best performing model. We choose one data set from each category from Table 1.

We fixed the prior parameters $\alpha_l = 1, \beta_l = |\mathcal{A}_l|$ for $l = 1, .., L$, and only varied likelihood parameters $\alpha_+, \beta_+, \alpha_-, \beta_-$ to analyze how the performance changes as the magnitude and ratio of the parameters change. To simplify the study, we took $\alpha_+ = \alpha_- = \alpha$, and $\beta_+ = \beta_- = \beta$. We let $s = \alpha + \beta$ and s was chosen from $\{100, 1000, 10000\}$. We let $\rho = \frac{\alpha}{\alpha+\beta}$ and ρ varied within $[0, 1]$. Here, s and ρ uniquely define α and β. We constructed models for different s and ρ and plotted the average out-of-sample accuracy as s and ρ changed in Figure 6 for three data sets representative of categorical, numerical and mixed data types. The X axis represents ρ and Y axis represents s. Accuracies increase as ρ increases, which is consistent with the intuition of ρ_+ and ρ_-. The highest accuracy is always achieved at the right half of the curve. The performance is less sensitive to the magnitude s and ratio ρ, especially when $\rho > 0.5$. For tic-tac-toe and diabetes, the accuracy became flat once ρ becomes greater than a certain threshold, and the performance was not sensitive to ρ either after that point; it is important as it makes tuning the algorithm easy in practice. Generally, taking ρ close to 1 leads to a satisfactory output. Table 1 and 2 were generated by models with $\rho = 0.9$ and $s = 1000$.

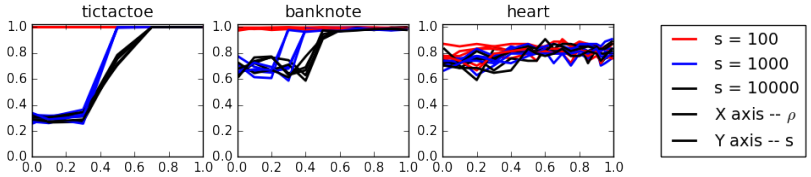


Figure 6: Parameter tuning experiments. Accuracy vs. ρ for all data sets. X axis represents ρ and Y axis represents accuracy.

6.1.2 DEMONSTRATION WITH TIC-TAC-TOE DATA

Let us illustrate the reduction in computation that comes from the bounds in the theorems in Section 4. For this demonstration, we chose the tic-tac-toe data set since the data can be classified exactly by eight rules which are naturally an underlying true rule set, and our method recovered the rules successfully, as shown in Table 1. We use BRS-BetaBinomial for this demonstration.

First, we used FP-Growth (Borgelt, 2005) to generate rules. At this step, we find all possible rules with length between 1 to 9 (the data has 9 attributes) and has minimum support of 1 (the rule's itemset must appear at least once int the data set). FP-growth generates a total of 84429 rules. We then set the hyper-parameters as:

$$\alpha_l = 1, \beta_l = |\mathcal{A}_l| \text{ for } l = 1, ..., 9,$$
$$\alpha_+ = N_+ + 1, \beta_+ = N_+, \alpha_- = N_- + 1, \beta_- = N_-.$$

We ran Algorithm 1 on this data set. We kept a list of the best solutions $v^{[t]}$ found until time t and updated $m_l^{[t-1]}$ and the minimum support with the new $v^{[t]}$ according to Theorem 1. We show in Figure 7 the minimum support at different iterations whenever a better $v^{[t]}$ is obtained. Within

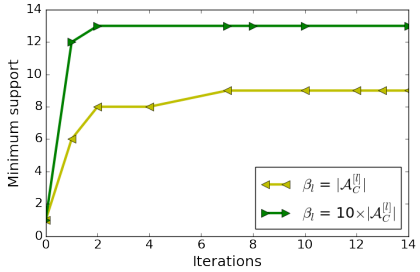
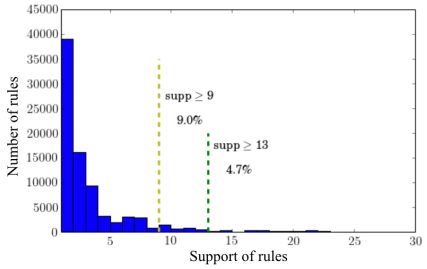


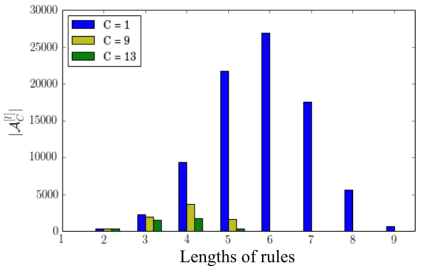
Figure 7: Updated minimum support at different iterations

ten iterations, the minimum support increased from 1 to 9. If, however, we choose a stronger prior for small models by increasing β_l to 10 times $|\mathcal{A}_l|$ for all l and keeping all the other parameters unchanged, then the minimum support increases even faster and reached 13 very quickly. This is consistent with the intuition that if the prior penalizes large models more heavily, the output tends to have a smaller size. Therefore each rule will need to cover more data.

Placing a bound on the minimum support is critical for computation since the number of possible rules decays exponentially with the support, as shown in Figure 8a. As the minimum support is increased, more rules are removed from the original rule space. Figure 8a shows the percentage of rules left in the solution space as the minimum threshold is increased from 1 to 9 and to 13. At a minimum support of 9, the rule space is reduced to 9.0% of its original size; at a minimum support of 13, the rule space is reduced to 4.7%. This provides intuition of the benefit of Theorem 1.



(a) Reduced rule space as the minimum support increases



(b) Number of rules of different lengths mined at different minimum support.

Figure 8: Demonstration with Tic-Tac-Toe data set

Although rules are filtered out based on their support, we observe that longer rules are more heavily reduced than shorter rules. This is because the support of a rule is negatively correlated with its length, as the more conditions a rule contains, the fewer observations can satisfy it. Figure 8b shows distributions of rules across different lengths when mined at different minimum support levels. Before any filtering (i.e., C = 1), more than half of the rules have lengths greater than 5. As C is increased to 9 and 13, these long rules are almost completely removed from the rule space.

To summarize, the strength of the prior is important for computation, and changes the nature of the problem: a stronger prior can dramatically reduce the size of the search space. Any increase in minimum support, that the prior provides, leads to a smaller search by the rule mining method and a smaller optimization problem for the BRS algorithm. These are multiple order-of-magnitude decreases in the size of the problem.

6.2 Application to In-vehicle Recommendation System

For this experiment, our goal was to understand customers' response to recommendations made in an in-vehicle recommender system that provides coupons for local businesses. The coupons would be targeted to the user in his/her particular context. Our data were collected on Amazon Mechanical Turk via a survey that we will describe shortly. We used Turkers with high ratings (95% or above). Out of 752 surveys, 652 were accepted, which generated 12684 data cases (after removing rows containing missing attributes).

The prediction problem is to predict if a customer is going to accept a coupon for a particular venue, considering demographic and contextual attributes. Answers that the user will drive there 'right away' or 'later before the coupon expires' are labeled as '$Y = 1$' and answers 'no, I do not want the coupon' are labeled as '$Y = 0$'. We are interested in investigating five types of coupons: bars, takeaway food restaurants, coffee houses, cheap restaurants (average expense below $20 per person), expensive restaurants (average expense between $20 to $50 per person). In the first part of the survey, we asked users to provide their demographic information and preferences. In the second part, we described 20 different driving scenarios (see examples in Appendix B) to each user along with additional context information and coupon information (see Appendix B for a full description of attributes) and asked the user if s/he will use the coupon.

For this problem, we wanted to generate simple BRS models that are easy to understand. So we restricted the lengths of rules and the number of rules to yield very sparse models. Before mining rules, we converted each row of data into an item which is an attribute and value pair. For categorical attributes, each attribute-value pair was directly coded into a condition. Using marital status as an example, 'marital status is single' was converted into (MaritalStatus: Single), (MaritalStatus: Not Married partner), and (MaritalStatus: Not Unmarried partner), (MaritalStatus: Not Widowed). For discretized numerical attributes, the levels are ordered, such as: age is '20 to 25', or '26 to 30', etc.; additionally, each attribute-value pair was converted into two conditions, each using one side of the range. For example, age is '20 to 25' was converted into (Age:>=20) and (Age:<=25). Then each condition is a half-space defined by threshold values. For the rule mining step, we set the minimum support to be 5% and set the maximum length of rules to be 4. We used information gain in Equation (15) to select the best 5000 rules to use for BRS. We ran simulated annealing for 50000 iterations to obtain a rule set.

We compared with interpretable classification algorithms that span the space of widely used methods that are known for interpretability and accuracy, C4.5, CART, Lasso, RIPPER, and a naïve baseline using top K rules, referred to as Top$K$. For C4.5, CART, Lasso, and RIPPER, we used the RWeka package in R and tuned the hyper-parameters to generate different models on a ROC plane. For the Top$K$ method, we varied K from 1 to 10 to produce ten models using best K pre-mined rules ranked by the information gain in Equation (15). For BRS, We varied the hyperparameters

$\alpha_+, \beta_+, \alpha_-, \beta_-$ to obtain different sets of rules. For all methods, we picked models on their ROC frontiers and reported their performance below.

Performance in accuracy To compare their predictive performance, we measured out-of-sample AUC (the Area Under The ROC Curve) from 5-fold testing for all methods, reported in Table 3. The BRS classifiers, while restricted to produce sparse disjunctions of conjunctions, had better performance than decision trees and RIPPER, which use greedy splitting and pruning methods, and do not aim to globally optimize. Top$K$'s performance was substantially below that of other methods. BRS models are also comparable to Lasso, but the form of the model is different. BRS models do not require users to cognitively process coefficients.

	Bar	Takeaway Food	Coffee House	Cheap Restaurant	Expensive Restaurant
BRS1	0.773 (0.013)	0.682 (0.005)	0.760 (0.010)	0.736 (0.022)	0.705 (0.025)
BRS2	0.776 (0.011)	0.667 (0.023)	0.762 (0.007)	0.736 (0.019)	0.707 (0.030)
C4.5	0.757 (0.015)	0.602 (0.051)	0.751 (0.018)	0.692 (0.033)	0.639 (0.027)
CART	0.772 (0.019)	0.615 (0.035)	0.758 (0.013)	0.732 (0.018)	0.657 (0.010)
Lasso	0.795 (0.014)	0.673 (0.042)	0.786 (0.011)	0.769 (0.024)	0.706 (0.017)
RIPPER	0.762 (0.015)	0.623 (0.048)	0.762(0.012)	0.705(0.023)	0.689 (0.034)
Top$K$	0.562 (0.015)	0.523 (0.024)	0.502(0.012)	0.582(0.023)	0.508 (0.011)

Table 3: AUC comparison for mobile advertisement data set, means and standard deviations over folds are reported.

Performance in complexity For the same experiment, we would also like to know the complexity of all methods at different accuracy levels. Since the methods we compare have different structures, there is not a straightforward way to directly compare complexity. However, decision trees can be converted into equivalent rule set models. For a decision tree, an example is classified as positive if it falls into any positive leaf. Therefore, we can generate equivalent models in rule set form for each decision tree, by simply collecting branches with positive leaves. Therefore, the four algorithms we compare, C4.5, CART, RIPPER, and BRS have the same form. To measure the complexity, we count the total number of conditions in the model, which is the sum of lengths for all rules. This loosely represents the cognitive load needed to understand a model. For each method, we take the models that are used to compute the AUC in Table 3 and plot their accuracy and complexity in Figure 9. BRS models achieved the highest accuracy at the same level of complexity. This is not surprising given that BRS performs substantially more optimization than other methods.

To show that the benefits of BRS did not come from rule mining or screening using heuristics, we compared the BRS models with Top$K$ models that rely solely on rule mining and ranking with heuristics. Figure 10 shows there is a substantial gain in the accuracy of BRS models compared to Top$K$ models.

From Table 3 we observe that Lasso achieved consistently good AUC. For the five coupons, the average number of nonzero coefficients for lasso models in different folds are 93, 94.6, 90.2, 93.2 and 93.4, which is on the order of $\sim$20 times larger than the number of conditions used in BRS and other rule based models; in this case, the lasso models are not interpretable.

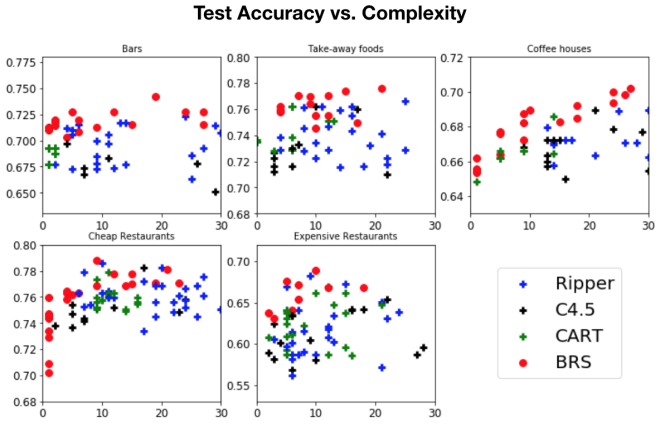


Figure 9: Test accuracy vs complexity for BRS and other models on mobile advertisement data sets for different coupons

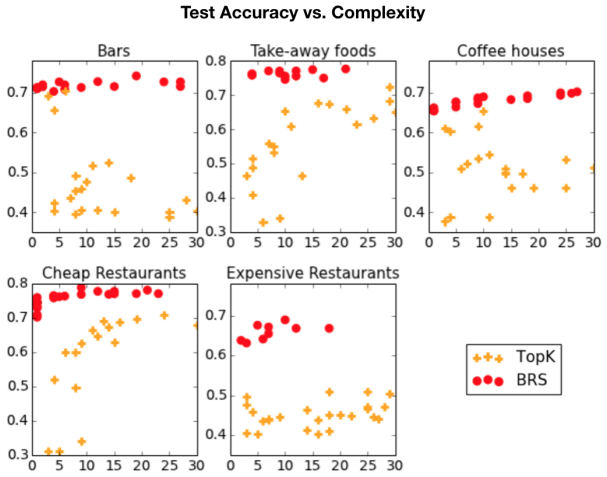
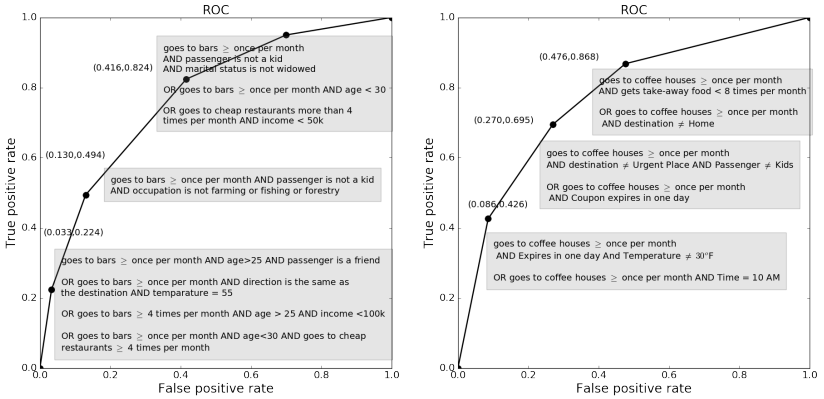


Figure 10: Test accuracy vs complexity for BRS and Top$K$ on mobile advertisement data sets for different coupons

Examples of BRS models In practice, for this particular application, the benefits of interpretability far outweigh small improvements in accuracy. An interpretable model can be useful to a vender

choosing whether to provide a coupon and what type of coupon to provide, it can be useful to users of the recommender system, and it can be useful to the designers of the recommender system to understand the population of users and correlations with successful use of the system. As discussed in the introduction, rule set classifiers are particularly natural for representing consumer behavior, particularly consideration sets, as modeled here.

We show several classifiers produced by BRS in Figure 11, where the curves were produced by models from the experiments discussed above. Example rule sets are listed in each box along the curve. For instance, the classifier near the middle of the curve in Figure 11 (a) has one rule, and reads "If a person visits a bar at least once per month, is not traveling with kids, and their occupation is not farming/fishing/forestry, then predict the person will use the coupon for a bar before it expires." In these examples (and generally), we see that a user's general interest in a coupon's venue (bar, coffee shop, etc.) is the most relevant attribute to the classification outcome; it appears in every rule in the two figures.



(a) Coupons for bars

(b) Coupons for coffee houses

Figure 11: ROC for data set of coupons for bars and coffee houses.

7. Conclusion

We presented a method that produces rule set (disjunctive normal form) models, where the shape of the model can be controlled by the user through Bayesian priors. In some applications, such as those arising in customer behavior modeling, the form of these models may be more useful than traditional linear models. Since finding sparse models is computationally hard, most approaches take severe heuristic approximations (such as greedy splitting and pruning in the case of decision trees, or convexification in the case of linear models). These approximations can severely hurt performance, as is easily shown experimentally, using data sets whose ground truth formulas are not difficult to find. We chose a different type of approximation, where we make an up-front statistical assumption in building our models out of pre-mined rules, and aim to find the globally optimal

solution in the reduced space of rules. We then find theoretical conditions under which using pre-mined rules provably does not change the set of MAP optimal solutions. These conditions relate the size of the data set to the strength of the prior. If the prior is sufficiently strong and the data set is not too large, the set of pre-mined rules is provably sufficient for finding an optimal rule set. We showed the benefits of this approach on a consumer behavior modeling application of current interest to "connected vehicle" projects. Our results, using data from an extensive survey taken by several hundred individuals, show that simple rules based on a user's context can be directly useful in predicting the user's response.

Appendix A.

Proof 1 *(of Lemma 1) Let TP, FP, TN, and FN be the number of true positives, false positives, true negatives and false negatives in S classified by A. We now compute the likelihood for model $A_{\setminus z}$. The most extreme case is when rule z is an 100% accurate rule that applies only to real positive data points and those data points satisfy only z. Therefore once removing it, the number of true positives decreases by supp(z) and the number of false negatives increases by supp(z). That is,*

$$\begin{aligned} p(S|A_{\setminus z}) \geq & \frac{B(TP - supp(z) + \alpha_+, FP + \beta_+)}{B(\alpha_+, \beta_+)} \frac{B(TN + \alpha_-, FN + supp(z) + \beta_-)}{B(\alpha_-, \beta_-)} \\ = & p(S|A) \cdot g_3(supp(z)), \end{aligned} \tag{17}$$

where

$$g_3(supp(z)) = \frac{\Gamma(TP + \alpha_+ - supp(z))}{\Gamma(TP + \alpha_+)} \frac{\Gamma(TP + FP + \alpha_+ + \beta_+)}{\Gamma(TP + FP + \alpha_+ + \beta_+ - supp(z))} \tag{18}$$

$$\frac{\Gamma(FN + \beta_- + supp(z))}{\Gamma(FN + \beta_-)} \frac{\Gamma(TN + FN + \alpha_- + \beta_-)}{\Gamma(TN + FN + \alpha_- + \beta_- + supp(z))}. \tag{19}$$

Now we break down $g_3(supp(z))$ to find a lower bound for it. The first two terms in (18) become

$$\begin{aligned} & \frac{\Gamma(TP + \alpha_+ - supp(z))}{\Gamma(TP + \alpha_+)} \frac{\Gamma(TP + FP + \alpha_+ + \beta_+)}{\Gamma(TP + FP + \alpha_+ + \beta_+ - supp(z))} \\ = & \frac{(TP + FP + \alpha_+ + \beta_+ - supp(z)) \ldots (TP + FP + \alpha_+ + \beta_+ - 1)}{(TP + \alpha_+ - supp(z)) \ldots (TP + \alpha_+ - 1)} \\ \geq & \left(\frac{TP + FP + \alpha_+ + \beta_+ - 1}{TP + \alpha_+ - 1} \right)^{supp(z)} \\ \geq & \left(\frac{N_+ + \alpha_+ + \beta_+ - 1}{N_+ + \alpha_+ - 1} \right)^{supp(z)}. \end{aligned} \tag{20}$$

Equality holds in (20) when TP $= N_+$, FP $= 0$. Similarly, the last two terms in (18) become

$$\begin{aligned}
&\frac{\Gamma(FN+\beta_- + supp(z))}{\Gamma(FN+\beta_-)}\frac{\Gamma(TN+FN+\alpha_-+\beta_-)}{\Gamma(TN+FN+\alpha_-+\beta_-+supp(z))}\\
=&\frac{(FN+\beta_-)\dots(FN+\beta_-+supp(z)-1)}{(TN+FN+\alpha_-+\beta_-)\dots(TN+FN+\alpha_-+\beta_-+supp(z)-1)}\\
\geq&\left(\frac{FN+\beta_-}{FN+TN+\alpha_-+\beta_-}\right)^{supp(z)}\\
\geq&\left(\frac{\beta_-}{N_-+\alpha_-+\beta_-}\right)^{supp(z)}. \qquad (21)
\end{aligned}$$

Equality in (21) holds when TN $= N_-$, FN $= 0$. Combining (17), (18), (20) and (21), we obtain

$$\begin{aligned}
P(S|A_{\setminus z}) &\geq \left(\frac{N_+ + \alpha_+ + \beta_+ - 1}{N_+ + \alpha_+ - 1}\frac{\beta_-}{N_- + \alpha_- + \beta_-}\right)^{supp(z)} \cdot p(S|A)\\
&= \Upsilon^{supp(z)} p(S|A). \qquad (22)
\end{aligned}$$

Proof 2 *(of Theorem 1)*

Step 1 *We first prove the upper bound $m_l^{[t]}$. Since $A^* \in \arg\max_A F(A)$, $F(A^*) \geq v^{[t]}$, i.e.,*

$$\log p(S|A^*) + \log p(A^*) \geq v^{[t]}. \qquad (23)$$

We then upper bound the two terms on the left-hand-side.

Let M_l^* denote the number of rules of length l in A^*. The prior probability of selecting A^* from $\mathcal{A}$ is

$$p(A^*) = p(\emptyset)\prod_{l=1}^{L}\frac{B(M_l+\alpha_l, |\mathcal{A}_l| - M_l + \beta_l)}{B(\alpha_l, |\mathcal{A}_l|+\beta_l)}.$$

We observe that when $0 \leq M_l^* \leq |\mathcal{A}_l|$,

$$B(M_l+\alpha_l, |\mathcal{A}_l| - M_l + \beta_l) \leq B(\alpha_l, |\mathcal{A}_l|+\beta_l),$$

giving

$$\begin{aligned}
p(A^*) &\leq p(\emptyset)\frac{B(M_{l'}+\alpha_{l'}, |\mathcal{A}_{l'}| - M_{l'} + \beta_{l'})}{B(\alpha_{l'}, |\mathcal{A}_{l'}|+\beta_{l'})}\\
&= p(\emptyset)\frac{\alpha_{l'}\cdot(\alpha_{l'}+1)\cdots(M_{l'}^*+\alpha_{l'}-1)}{(|\mathcal{A}_{l'}| - M_{l'}^* + \beta_{l'})\cdots(|\mathcal{A}_{l'}|+\beta_{l'}-1)}\\
&\leq p(\emptyset)\left(\frac{M_{l'}^*+\alpha_{l'}-1}{|\mathcal{A}_{l'}|+\beta_{l'}-1}\right)^{M_{l'}^*} \qquad (24)
\end{aligned}$$

for all $l' \in \{1, ..., L\}$. The likelihood is upper bounded by

$$p(S|A^*) \leq \mathcal{L}^*. \qquad (25)$$

Substituting (24) and (25) into (23) we get

$$\log\mathcal{L}^* + \log p(\emptyset) + M_{l'}^*\log\left(\frac{M_{l'}^*+\alpha_{l'}-1}{|\mathcal{A}_{l'}|+\beta_{l'}-1}\right) \geq v^{[t]}, \qquad (26)$$

which gives

$$\begin{aligned}
M_{l'}^* &\leq \frac{\log\mathcal{L}^* + \log p(\emptyset) - v^{[t]}}{\log\left(\frac{|\mathcal{A}_{l'}|+\beta_{l'}-1}{M_{l'}^*+\alpha_{l'}-1}\right)}\\
&\leq \frac{\log\mathcal{L}^* + \log p(\emptyset) - v^{[t]}}{\log\left(\frac{|\mathcal{A}_{l'}|+\beta_{l'}-1}{m_{l'}^{[t-1]}+\alpha_{l'}-1}\right)}, \qquad (27)
\end{aligned}$$

where (27) follows because $m_{l'}^{[t-1]}$ is an upper bound on $M_{l'}^*$. Since $M_{l'}^*$ has to be an integer, we get

$$\begin{aligned}
M_{l'}^* &\leq \left\lfloor\frac{\log\mathcal{L}^* + \log p(\emptyset) - v^{[t]}}{\log\left(\frac{|\mathcal{A}_{l'}|+\beta_{l'}-1}{m_{l'}^{[t-1]}+\alpha_{l'}-1}\right)}\right\rfloor\\
&= m_{l'}^{[t]} \text{ for all } l' \in \{1, ..., L\}. \qquad (28)
\end{aligned}$$

Thus

$$M^* = \sum_{l=1}^{L} M_l^* \leq \sum_{l=1}^{L} m_l^{[t]}. \qquad (29)$$

Step 2: *Now we prove the lower bound on the support. We would like to prove that a MAP model does not contain rules of support less than a threshold. To show this, we prove that if any rule z has support smaller than some constant C, then removing it yields a better objective, i.e.,*

$$F(A) \leq F(A_{\setminus z}). \qquad (30)$$

Our goal is to find conditions on C such that this inequality holds. Assume in rule set A, M_l comes from pool $\mathcal{A}_l$ of rules with length l, $l \in \{1, ...L\}$, and rule z has length l' so it is drawn from $\mathcal{A}_{l'}$.

$A_{\setminus z}$ consists of the same rules as A except missing one rule from $\mathcal{A}_{l'}$. We must have:

$$\begin{aligned}
p(A_{\setminus z}) =& \frac{B(M_{l'}+\alpha_{l'}, |\mathcal{A}_{l'}| - M_{l'} + \beta_{l'})}{B(\alpha_{l'}, \beta_{l'})}\cdot\prod_{l\neq l'}\frac{B(M_l+\alpha_l, |\mathcal{A}_l| - M_l + \beta_l)}{B(\alpha_l, \beta_l)}\\
=& p(A)\cdot\frac{B(M_{l'}+\alpha_{l'}, |\mathcal{A}_{l'}| - M_{l'} + \beta_{l'})}{B(M_{l'}+1+\alpha_{l'}, |\mathcal{A}_{l'}| - M_{l'} - 1 + \beta_{l'})}\\
=& p(A)\cdot\frac{|\mathcal{A}_{l'}| - M_{l'} + \beta_{l'}}{M_{l'} - 1 + \alpha_{l'}}.
\end{aligned}$$

$\frac{|\mathcal{A}_{l'}| - M_{l'} + \beta_{l'}}{M_{l'} - 1 + \alpha_{l'}}$ decreases monotonically as $M_{l'}$ increases, so it is lower bounded at the upper bound on $M_{l'}$, $m_{l'}^{[t]}$, obtained in step 1. Therefore

$$\frac{|\mathcal{A}_{l'}| - M_{l'} + \beta_{l'}}{M_{l'} - 1 + \alpha_{l'}} \geq \frac{|\mathcal{A}_{l'}| - m_{l'}^{[t]} + \beta_{l'}}{m_{l'}^{[t]} - 1 + \alpha_{l'}} \text{ for } l' \in \{1, ...L\}.$$

Thus

$$p(A_{\setminus z}) \geq \max_l \left(\frac{|\mathcal{A}_l| - m_l^{[t]} + \beta_l}{m_l^{[t]} - 1 + \alpha_l} \right) p(A). \tag{31}$$

Combining (31) and Lemma 1, the joint probability of S and $A_{\setminus z}$ is bounded by

$$\begin{aligned} P(S, A_{\setminus z}) =& p(A_{\setminus z}) P(S|A_{\setminus z}) \\ \geq& \max_l \left(\frac{|\mathcal{A}_l| - m_l^{[t]} + \beta_l}{m_l^{[t]} - 1 + \alpha_l} \right) (\Upsilon)^{supp(z)} \cdot P(S, A). \end{aligned}$$

In order to get $P(S, A_{\setminus z}) \geq P(S, A)$, and with $\Upsilon \leq 1$ from the assumption in the theorem's statement, we must have

$$supp(z) \leq \frac{\log \max_l \left(\frac{|\mathcal{A}_l| - m_l^{[t]} + \beta_l}{m_l^{[t]} - 1 + \alpha_l} \right)}{\log \Upsilon}.$$

This means if $\exists z \in A$ such that $supp(z) \leq \frac{\log \max_l \left(\frac{|\mathcal{A}_l| - m_l^{[t]} + \beta_l}{m_l^{[t]} - 1 + \alpha_l} \right)}{\log \Upsilon}$, then $A \notin \arg\max_{A'} F(A')$, which is equivalent to saying, for any $z \in A^*$,

$$supp(z) \geq \frac{\log \max_l \left(\frac{|\mathcal{A}_l| - m_l^{[t]} + \beta_l}{m_l^{[t]} - 1 + \alpha_l} \right)}{\log \Upsilon}.$$

Since the support of a rule has to be integer,

$$supp(z) \geq \left\lceil \frac{\log \max_l \left(\frac{|\mathcal{A}_l| - m_l^{[t]} + \beta_l}{m_l^{[t]} - 1 + \alpha_l} \right)}{\log \Upsilon} \right\rceil.$$

Before proving Theorem 2, we first prove the following lemma that we will need later.

Lemma 2 *Define a function $g(l) = \left(\frac{\lambda}{2}\right)^l \Gamma(J - l + 1), \lambda, J \in \mathbb{N}^+$. If $1 \leq l \leq J$,*

$$g(l) \leq \max \left\{ \left(\frac{\lambda}{2}\right) \Gamma(J+1), \left(\frac{\lambda}{2}\right)^J \right\}.$$

Proof 3 *(Of Lemma 2) In order to bound $g(l)$, we will show that $g(l)$ is convex, which means its maximum value occurs at the endpoints of the interval we are considering. The second derivative of $g(l)$ respect to l is*

$$g''(l) = g(l) \left[\left(\ln \frac{\lambda}{2} + \gamma - \sum_{k=1}^{\infty} \frac{1}{k} - \frac{1}{k + J - L_m} \right)^2 + \sum_{k=1}^{\infty} \frac{1}{(k + J - l)^2} \right] > 0, \tag{32}$$

since at least one of the terms $\frac{1}{(k+J-l)^2} > 0$. Thus $g(l)$ is strictly convex. Therefore the maximum of $g(l)$ is achieved at the boundary of L_m, namely 1 or J. So we have

$$\begin{aligned} g(l) &\leq \max \{g(1), g(J)\} \\ &= \max \left\{ \left(\frac{\lambda}{2}\right) \Gamma(J+1), \left(\frac{\lambda}{2}\right)^J \right\}. \end{aligned} \tag{33}$$

Proof 4 *(Of Theorem 2) We follow similar steps in the proof for Theorem 1.*
Step 1 *We first derive an upper bound M^* at time t, denoted as $M^{[t]}$.*

The probability of selecting a rule set A^* depends on the number of rules M^* and the length of each rule, which we denote as $L_m, m \in \{1, ..., M^*\}$, so the prior probability of selecting A^* is

$$\begin{aligned} P(A^*; \theta) &= \omega(\lambda, \eta) Poisson(M^*; \lambda) \prod_m^{M^*} Poisson(L_m; \eta) \frac{1}{\binom{J}{L_m}} \prod_k^{L_m} \frac{1}{K_{v_{m,k}}} \\ &= p(\emptyset) \frac{\lambda^{M^*}}{\Gamma(M^*+1)} \prod_m^{M^*} \frac{e^{-\eta} \eta^{L_m} \Gamma(J - L_m + 1)}{\Gamma(J+1)} \prod_k^{L_m} \frac{1}{K_{v_{m,k}}} \\ &\leq p(\emptyset) \frac{1}{\Gamma(M^*+1)} \left(\frac{e^{-\eta} \lambda}{\Gamma(J+1)} \right)^{M^*} \prod_m^{M^*} \left(\frac{\eta}{2}\right)^{L_m} \Gamma(J - L_m + 1). \end{aligned} \tag{34}$$

(34) follows from $K_{v_{m,k}} \geq 2$ since all attributes have at least two values. Using Lemma 2 we have that

$$\left(\frac{\eta}{2}\right)^{L_m} \Gamma(J - L_m + 1) \leq \max \left\{ \left(\frac{\eta}{2}\right) \Gamma(J+1), \left(\frac{\eta}{2}\right)^J \right\}. \tag{35}$$

Combining (34) and (35), and the definition of x, we have

$$P(A^*; \theta) \leq p(\emptyset) \frac{x^{M^*}}{\Gamma(M^*+1)}. \tag{36}$$

Now we apply (23) combining with (36) and (25), we get

$$\log \mathcal{L}^* + \log p(\emptyset) + \log \frac{x^{M^*}}{\Gamma(M^*+1)} \geq v^{[t]} \tag{37}$$

Now we want to upper bound $\frac{x^{M^*}}{\Gamma(M^*+1)}$.

If $M^* \leq \lambda$, the statement of the theorem holds trivially. For the remainder of the proof we consider, if $M^* > \lambda$, $\frac{x^{M^*}}{\Gamma(M^*+1)}$ is upper bounded by

$$\frac{x^{M^*}}{M^*!} \leq \frac{x^\lambda}{\lambda!} \frac{x^{(M^*-\lambda)}}{(\lambda+1)^{(M^*-\lambda)}},$$

where in the denominator we used $\Gamma(M^*+1) = \Gamma(\lambda+1)(\lambda+1) \cdots M^* \geq \Gamma(\lambda+1)(\lambda+1)^{(M^*-\lambda)}$. So we have

$$\log \mathcal{L}^* + \log p(\emptyset) + \log \frac{x^\lambda}{\Gamma(\lambda+1)} + (M - \lambda) \log \frac{x}{\lambda+1} \geq v^{[t]}. \tag{38}$$

We have $\frac{x}{\lambda+1} \leq 1$. *To see this, note* $\frac{\Gamma(J)}{\Gamma(J+1)} < 1, e^{-\eta}\left(\frac{\eta}{2}\right) < 1$ *and* $\frac{e^{-\eta}\left(\frac{\eta}{2}\right)^J}{\Gamma(J+1)} < 1$ *for every* η *and* J. *Then solving for* M^* *in (38), using* $\frac{x}{\lambda+1} < 1$ *to determine the direction of the inequality yields:*

$$M^* \leq M^{[t]} = \left\lfloor \lambda + \frac{\log \mathcal{L}^* + \log p(\emptyset) + \log \frac{x^\lambda}{\Gamma(\lambda+1)} - v^{[t]}}{\log \frac{\lambda+1}{x}} \right\rfloor \tag{39}$$

$$= \left\lfloor \frac{\log \frac{(\lambda+1)^\lambda}{\Gamma(\lambda+1)}}{\log \frac{\lambda+1}{x}} + \frac{\log \mathcal{L}^* + \log p(\emptyset) - v^{[t]}}{\log \frac{\lambda+1}{x}} \right\rfloor. \tag{40}$$

Step 2 *Now we prove the lower bound on the support of rules in an optimal set* A^*. *Similar to the proof for Theorem 1, we will show that for a rule set* A, *if any rule* a_z *has support* $supp(z) < C$ *on data* S, *then* $A \notin \arg\min_{A' \in \Lambda_S} F(A')$. *Assume rule* a_z *has support less than C and* $A_{\setminus z}$ *has the* k*-th rule removed from* A. *Assume* A *consists of* M *rules, and the* z*-th rule has length* L_z.

We relate $P(A_{\setminus z}; \theta)$ *with* $P(A; \theta)$. *We multiply* $P(A_{\setminus z}; \theta)$ *with 1 in disguise to relate it to* $P(A; \theta)$:

$$P(A_{\setminus z}) = \frac{M^*\Gamma(J+1)}{\lambda e^{-\eta}\eta^{L_z}\Gamma(J-L_z+1)\prod_k^{L_z}\frac{1}{K_{v_{z,k}}}} P(A)$$

$$\geq \frac{M^*\Gamma(J+1)}{\lambda e^{-\eta}\left(\frac{\eta}{2}\right)^{L_z}\Gamma(J-L_z+1)} P(A) \tag{41}$$

$$= \frac{M^*\Gamma(J+1)}{\lambda e^{-\eta} g(L_z; \eta, J)} P(A) \tag{42}$$

$$\geq \frac{M^* P(A)}{x} \tag{43}$$

where (41) follows that $K_{v_{m,k}} \geq 2$ *since all attributes have at least two values, (42) follows the definition of* $g(l)$ *in Lemma 2 and (43) uses the upper bound in Lemma 2 and. Then combining (22) with (42), the joint probability of* S *and* $A_{\setminus z}$ *is lower bounded by*

$$P(S, A_{\setminus z}) = P(A_{\setminus z})P(S|A_{\setminus z})$$

$$\geq \frac{M^*\Upsilon^{supp(z)}}{x} P(S, A).$$

In order to get $P(S, A_{\setminus z}) \geq P(S, A)$, *we need*

$$\frac{M^*\Upsilon^{supp(z)}}{x} \geq 1,$$

i.e.,

$$\Upsilon^{supp(z)} \geq \frac{x}{M^*} \geq \frac{x}{M^{[t]}},$$

We have $\Upsilon \leq 1$, *thus*

$$supp(z) \leq \frac{\log \frac{x}{M^{[t]}}}{\log \Upsilon}. \tag{44}$$

Therefore, for any rule z *in a MAP model* A^*,

$$supp(z) \geq \left\lceil \frac{\log \frac{x}{M^{[t]}}}{\log \Upsilon} \right\rceil. \tag{45}$$

Appendix B. Mobile Advertisement data sets

The attributes of this data set include:

1. User attributes
   - Gender: male, female
   - Age: below 21, 21 to 25, 26 to 30, etc.
   - Marital Status: single, married partner, unmarried partner, or widowed
   - Number of children: 0, 1, or more than 1
   - Education: high school, bachelors degree, associates degree, or graduate degree
   - Occupation: architecture & engineering, business & financial, etc.
   - Annual income: less than $12500, $12500 - $24999, $25000 - $37499, etc.
   - Number of times that he/she goes to a bar: 0, less than 1, 1 to 3, 4 to 8 or greater than 8
   - Number of times that he/she buys takeaway food: 0, less than 1, 1 to 3, 4 to 8 or greater than 8
   - Number of times that he/she goes to a coffee house: 0, less than 1, 1 to 3, 4 to 8 or greater than 8
   - Number of times that he/she eats at a restaurant with average expense less than $20 per person: 0, less than 1, 1 to 3, 4 to 8 or greater than 8
   - Number of times that he/she goes to a bar: 0, less than 1, 1 to 3, 4 to 8 or greater than 8
2. Contextual attributes
   - Driving destination: home, work, or no urgent destination
   - Location of user, coupon and destination: we provide a map to show the geographical location of the user, destination, and the venue, and we mark the distance between each two places with time of driving. The user can see whether the venue is in the same direction as the destination.
   - Weather: sunny, rainy, or snowy
   - Temperature: 30F°, 55F°, or 80F°
   - Time: 10AM, 2PM, or 6PM
   - Passenger: alone, partner, kid(s), or friend(s)
3. Coupon attributes
   - time before it expires: 2 hours or one day

All coupons provide a 20% discount. The survey was divided into different parts, so that Turkers without children would never see a scenario where their “kids" were in the vehicle. Figure 12 and 13 show two examples of scenarios in the survey.

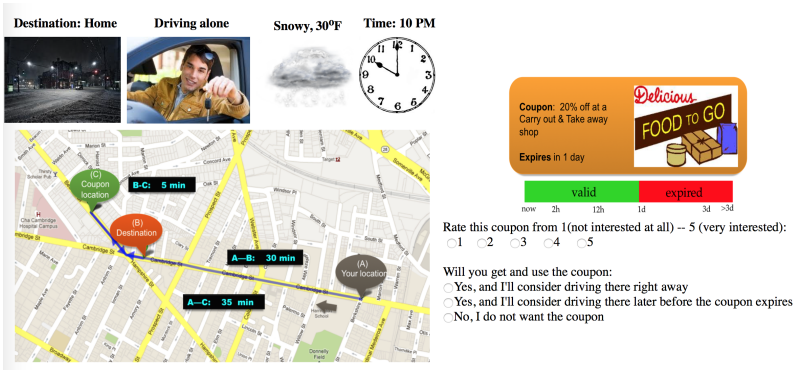


Figure 12: Example 1 of scenario in the survey

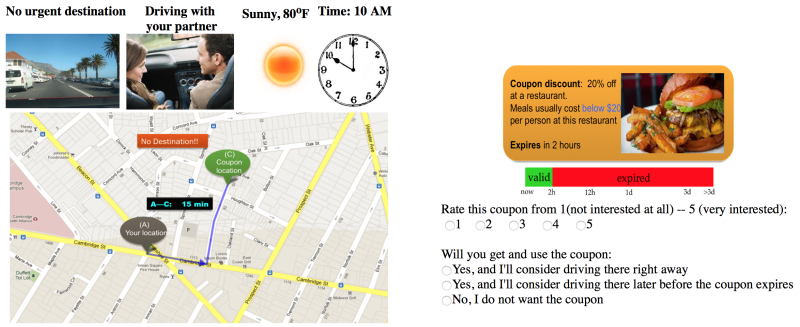


Figure 13: Example 2 of scenario in the survey

References

Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, 2005. ISSN 1041-4347.

Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Proc of the 2008 ACM Conf on Rec Systems*, RecSys '08, pages 335–336, New York, NY, USA, 2008. ACM.

Hiva Allahyari and Niklas Lavesson. User-oriented assessment of classification model understandability. In *SCAI*, pages 11–19, 2011.

E. Angelino, N. Larus-Stone, D. Alabi, M. Seltzer, and C. Rudin. Learning certifiably optimal rule lists for categorical data. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'17)*, 2017.

K. Bache and M. Lichman. UCI machine learning repository, 2013. URL `http://archive.ics.uci.edu/ml`.

Matthias Baldauf, Schahram Dustdar, and Florian Rosenberg. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4):263–277, 2007.

Linas Baltrunas, Marius Kaminskas, Bernd Ludwig, Omar Moling, Francesco Ricci, Aykan Aydin, Karl-Heinz Lüke, and Roland Schwaiger. Incarmusic: Context-aware music recommendations in a car. In *E-Commerce and Web Technologies*, pages 89–100. Springer, 2011.

Elena Baralis, Luca Cagliero, Tania Cerquitelli, Paolo Garza, and Marco Marchetti. Cas-mine: providing personalized services in context-aware applications by means of generalized rules. *Knowledge and information systems*, 28(2):283–310, 2011.

Christian Borgelt. An implementation of the fp-growth algorithm. In *Proc of the 1st interl workshop on open source data mining: frequent pattern mining implementations*, pages 1–5. ACM, 2005.

Endre Boros, Peter L Hammer, Toshihide Ibaraki, Alexander Kogan, Eddy Mayoraz, and Ilya Muchnik. An implementation of logical analysis of data. *IEEE Transactions on Knowledge and Data Engineering*, 12(2):292–306, 2000.

Sergey Brin, Rajeev Motwani, and Craig Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *ACM SIGMOD Record*, volume 26 (2), pages 265–276. ACM, 1997.

Guoqing Chen, Hongyan Liu, Lan Yu, Qiang Wei, and Xing Zhang. A new approach to classification based on association rule mining. *Decision Support Systems*, 42(2):674–689, 2006.

Hong Cheng, Xifeng Yan, Jiawei Han, and Chih-Wei Hsu. Discriminative frequent pattern analysis for effective classification. In *ICDE*, pages 716–725. IEEE, 2007.

P. Clark and T. Niblett. The cn2 induction algorithm. *Machine Learning*, 3:261–283, 1989.

William Cohen. Fast effective rule induction. In *Proceedings of the twelfth international conference on machine learning*, pages 115–123, 1995.

Yves Crama, Peter L Hammer, and Toshihide Ibaraki. Cause-effect relationships and partially defined boolean functions. *Annals of Operations Research*, 16(1):299–325, 1988.

Guozhu Dong, Xiuzhen Zhang, Limsoon Wong, and Jinyan Li. Caep: Classification by aggregating emerging patterns. In *International Conference on Discovery Science*, pages 30–42. Springer, 1999.

Vitaly Feldman. Hardness of approximate two-level logic minimization and pac learning with membership queries. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 363–372. ACM, 2006.

Vitaly Feldman. Learning DNF expressions from fourier spectrum. *CoRR*, abs/1203.0594, 2012.

Eibe Frank and Ian H. Witten. Generating accurate rule sets without global optimization. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 144–151, 1998.

Alex A Freitas. Comprehensible classification models: a position paper. *ACM SIGKDD Explorations Newsletter*, 15(1):1–10, March 2014.

Jerome H. Friedman and Nicholas I. Fisher. Bump hunting in high-dimensional data. *Statistics and Computing*, 9(2):123–143, 1999.

Brian R. Gaines and Paul Compton. Induction of ripple-down rules applied to modeling large databases. *Journal of Intelligent Information Systems*, 5(3):211–228, 1995.

Siong Thye Goh and Cynthia Rudin. Box drawings for learning with imbalanced data. In *Proceedings of the 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, 2014.

John R Hauser, Olivier Toubia, Theodoros Evgeniou, Rene Befurt, and Daria Dzyabura. Disjunctions of conjunctions, cognitive simplicity, and consideration sets. *Jour of Marketing Research*, 47(3):485–496, 2010.

Robert C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11(1):63–91, 1993.

Johan Huysmans, Karel Dejaeger, Christophe Mues, Jan Vanthienen, and Bart Baesens. An empirical evaluation of the comprehensibility of decision table, tree and rule based predictive models. *Decision Support Systems*, 51(1):141–154, 2011.

Chii-Ruey Hwang. Simulated annealing: theory and applications. *Acta Applicandae Mathematicae*, 12(1):108–111, 1988.

Adam R. Klivans and Rocco Servedio. Learning dnf in time $2^{O(n^{1/3})}$. In *Proc of the 33rd Annual ACM Symp on Theory of Computing*, STOC '01, pages 258–265, New York, NY, USA, 2001. ACM.

Bae-Hee Lee, Heung-Nam Kim, Jin-Guk Jung, and Geun-Sik Jo. Location-based service with context data for a restaurant recommendation. In *Database and Expert Systems Applications*, pages 430–438. Springer, 2006.

Benjamin Letham, Cynthia Rudin, Tyler H. McCormick, and David Madigan. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *Annals of Applied Statistics*, 2015. accepted with minor revision.

Wenmin Li, Jiawei Han, and Jian Pei. Cmar: Accurate and efficient classification based on multiple class-association rules. In *ICDM 2001*, pages 369–376. IEEE, 2001.

Bing Ma, Wynne Liu, and Yiming Hsu. Integrating classification and association rule mining. In *Proceedings of the 4th International Conf on Knowledge Discovery and Data Mining (KDD)*, 1998.

Dmitry Malioutov and Kush Varshney. Exact rule learning via boolean compressed sensing. In *ICML*, 2013.

David Martens and Bart Baesens. Building acceptable classification models. In *Data Mining*, pages 53–74. Springer, 2010.

David Martens, Jan Vanthienen, Wouter Verbeke, and Bart Baesens. Performance of classification models from a user perspective. *Decision Support Systems*, 51(4):782–793, 2011.

Tyler McCormick, Cynthia Rudin, and David Madigan. A hierarchical model for association rule mining of sequential events: An approach to automated medical symptom prediction. *Annals of Applied Statistics*, 2012.

R.S. Michalski. On the quasi-minimal solution of the general covering problem. In *Proceedings of the Fifth International Symposium on Information Processing*, pages 125–128, 1969.

Stephen Muggleton and Luc De Raedt. Inductive logic programming: Theory and methods. *The Journal of Logic Programming*, 19:629–679, 1994.

José M. Noguera, Manuel J. Barranco, Rafael J. Segura, and Luis Martinez. A mobile 3d-gis hybrid recommender system for tourism. *Information Sciences*, 215(0):37 – 52, 2012.

Moon-Hee Park, Jin-Hyuk Hong, and Sung-Bae Cho. Location-based recommendation system using bayesian user's preference model in mobile devices. In *Ubiquitous Intelligence and Computing*, pages 1130–1139. Springer, 2007.

Murray M Pollack, Urs E Ruttimann, and Pamela R Getson. Pediatric risk of mortality (prism) score. *Critical care medicine*, 16(11):1110–1116, 1988.

Peter R Rijnbeek and Jan A Kors. Finding a short and accurate decision rule in disjunctive normal form by exhaustive search. *Machine learning*, 80(1):33–62, 2010.

Cynthia Rudin, Benjamin Letham, and David Madigan. Learning theory analysis for association rules and sequential event prediction. *Journal of Machine Learning Research*, 14:3384–3436, 2013.

Stefan Rüping. *Learning interpretable models*. PhD thesis, Universität Dortmund, 2006.

Robert E. Schapire. Using output codes to boost multiclass learning problems. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML)*, pages 313–321, 1997.

W Schwinger, Chr Grün, B Pröll, W Retschitzegger, and A Schauerhuber. Context-awareness in mobile tourism guides–a comprehensive survey. Technical report, Johannes Kepler University Linz, 2005.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Wei-Guang Teng, Ming-Jyh Hsieh, and Ming-Syan Chen. On the mining of substitution rules for statistically dependent items. In *ICDM 2003*, pages 442–449. IEEE, 2002.

Hung-Wen Tung and Von-Wun Soo. A personalized restaurant recommender agent for mobile e-service. In *IEEE Int Conf on e-Technology, e-Commerce and e-Service, 2004. (EEE)*, pages 259–262, 2004.

L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, November 1984. ISSN 0001-0782.

Mark Van Setten, Stanislav Pokraev, and Johan Koolwaaij. Context-aware recommendations in the mobile tourist application compass. In *Adaptive hypermedia and adaptive web-based systems*, pages 235–244. Springer, 2004.

Katrien Verbert, Nikos Manouselis, Xavier Ochoa, Martin Wolpers, Hendrik Drachsler, Ivana Bosnic, and Erik Duval. Context-aware recommender systems for learning: a survey and future challenges. *IEEE Transactions on Learning Technologies*, 5(4):318–335, 2012.

Fulton Wang and Cynthia Rudin. Falling rule lists. In *Proceedings of Artificial Intelligence and Statistics (AISTATS)*, 2015.

Tong Wang, Cynthia Rudin, F Doshi, Yimin Liu, Erica Klampfl, and Perry MacNeille. In-vehicle coupon recommendation dataset, 2015. URL `www.researchgate.net/publication/\318645502_in-vehicle_coupon_recommendation`.

Tong Wang, Cynthia Rudin, Finale Doshi, Yimin Liu, Erica Klampfl, and Perry MacNeille. Bayesian or's of and's for interpretable classification with application to context aware recommender systems. In *Proceedings of the International Conference on Data Mining (ICDM)*, 2016.

Xinxi Wang, David Rosenblum, and Ye Wang. Context-aware mobile music recommendation for daily activities. In *Proceedings of the 20th ACM International Conference on Multimedia*, pages 99–108, 2012.

Geoffrey I. Webb. Opus: An efficient admissible algorithm for unordered search. *J. Artif. Intell. Res.*, 3:431–465, 1995.

Xindong Wu, Chengqi Zhang, and Shichao Zhang. Mining both positive and negative association rules. In *Proceedings of the Nineteenth ICML*, pages 658–665. Morgan Kaufmann Publishers Inc., 2002.

Xiaoxin Yin and Jiawei Han. Cpar: Classification based on predictive association rules. In *SDM*, volume 3, pages 369–376. SIAM, 2003.

Journal of Machine Learning Research 18 (2017) 1-46 Submitted 2/16; Revised 3/17; Published 7/17

A Robust-Equitable Measure for Feature Ranking and Selection

A. Adam Ding A.DING@NEU.EDU
Department of Mathematics
Northeastern University
Boston, MA 02115, USA

Jennifer G. Dy JDY@ECE.NEU.EDU
Department of Electrical and Computer Engineering
Northeastern University
Boston, MA 02115, USA

Yi Li LI.YI3@HUSKY.NEU.EDU
Department of Mathematics
Northeastern University
Boston, MA 02115, USA

Yale Chang YCHANG@COE.NEU.EDU
Department of Electrical and Computer Engineering
Northeastern University
Boston, MA 02115, USA

Editor: Gal Elidan

Abstract

In many applications, not all the features used to represent data samples are important. Often only a few features are relevant for the prediction task. The choice of dependence measures often affect the final result of many feature selection methods. To select features that have complex nonlinear relationships with the response variable, the dependence measure should be *equitable*, a concept proposed by Reshef et al. (2011); that is, the dependence measure treats linear and nonlinear relationships equally. Recently, Kinney and Atwal (2014) gave a mathematical definition of *self-equitability*. In this paper, we introduce a new concept of *robust-equitability* and identify a robust-equitable copula dependence measure, the *robust copula dependence* (RCD) measure. RCD is based on the L_1-distance of the copula density from uniform and we show that it is equitable under both equitability definitions. We also prove theoretically that RCD is much easier to estimate than mutual information. Because of these theoretical properties, the RCD measure has the following advantages compared to existing dependence measures: it is robust to different relationship forms and robust to unequal sample sizes of different features. Experiments on both synthetic and real-world data sets confirm the theoretical analysis, and illustrate the advantage of using the dependence measure RCD for feature selection.

Keywords: dependence measure, feature selection, copula, equitability, mutual information

1. Introduction

The performance of machine learning algorithms is dependent on the input features representing each data sample. Often not all of these features are useful: some may be irrelevant and some may be redundant. Feature selection is thus needed to help improve the performance of learning tasks. Moreover, feature selection can decrease the computational cost of algorithms, and provide domain experts with an increased understanding of which factors are important.

Feature selection algorithms can be categorized based on how the learning algorithms is incorporated into the selection algorithm: *filter*, *wrapper*, or *embedded* methods (Kohavi and John, 1997; Guyon and Elisseeff, 2003; Yu and Liu, 2004). Filter methods (Kira and Rendell, 1992; Yu and Liu, 2004; Peng et al., 2005; He et al., 2005; Song et al., 2007) pre-select features, without running the learning algorithm. Features are evaluated only through the intrinsic properties of the data. Wrapper methods (Kohavi and John, 1997; Guyon et al., 2002; Dy and Brodley, 2004) "wraps" the search around the learning algorithm and evaluate candidate feature subsets based on learning performance in each candidate feature subset. Embedded methods (Tibshirani, 1996; Vapnik and Vapnik, 1998) incorporate feature search and the learning algorithm into a single optimization problem formulation. Wrapper and embedded methods, contrary to filter methods, select features specific to the learning algorithm; thus, they are most likely to be more accurate than filter methods on a particular learning algorithm, but the features they choose may not be appropriate for other algorithms. Another limitation of wrapper methods is that they are computationally expensive because they need to train and test the learning algorithm for each feature subset candidate, which can be prohibitive when working with high-dimensional data.

Filter methods rely on measures based on intrinsic properties of the data. More specifically, they evaluate features based on some dependence measure criterion between features and the target variable and select the subset of features that optimizes this criterion. Let d be the number of original features. An exhaustive search, which involves 2^d possible feature subsets is computationally impractical. Thus, one commonly employs heuristic search strategies, such as greedy approaches (e.g., sequential forward/backward search (Pudil et al., 1994)). However, these strategies can lead to local optima. Random search methods, such as genetic algorithms, add some randomness to help escape from local optima. When the dimensionality is very high, one can only afford an individual search. Individual search methods (Guyon and Elisseeff, 2003; He et al., 2005) evaluate each feature individually according to a criterion and then select features, which either satisfies a condition or are top-ranked. The problem with individual search methods is that they ignore feature interaction and dependencies. To account for such interactions and dependencies, Yu and Liu (2004) selects relevant features individually and then add a separate redundancy removal step to account for linear correlation between features; Peng et al. (2005) suggests another way, by maximizing relevance and minimizing redundancy (mRMR) together.

In addition to search strategies, the performance of filter methods depend heavily on the choice of dependence measures. The ability to measure the dependence between random variables is a fundamental problem in statistics and machine learning. One of the simplest and most common dependence measure is the Pearson correlation coefficient (ρ_{lin}). However, this measure only captures linear relationships. Another popular measure is mutual information (MI). MI can capture nonlinear dependencies but is difficult to estimate (Fernandes and Gloor, 2010; Reshef et al., 2011) (see Theorem 3 in Section 4). Kernel-based dependence measures (Gretton et al., 2005a; Fukumizu et al., 2007) (e.g., the Hilbert-Schmidt Independence Criterion (HSIC)) have been introduced as an alternative to MI which does not require explicitly learning joint distributions. However, HSIC depends on the choice of kernels. Hilbert-Schmidt Normalized Information Criterion (HSNIC), also known as normalized conditional cross-covariance operator (NOCCO) (Fukumizu et al., 2007), is kernel-free, meaning it does not depend on the choice of kernels in the limit of infinite data. Even though HSNIC is kernel-free, both HSIC and HSNIC's values may vary when we use different

©2017 Ding, Dy, Li and Chang.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-079.html.

scales. Poczos et al. (2012) applied Maximum Mean Discrepancy (MMD) after empirical copula transformation to make the kernel-based dependence measure invariant to strictly monotone transformation of the marginal variables. The Copula-MMD (CMMD) can also be written in HSIC formulation after empirical copula transformation. Similarly, Reddi and Póczos (2013) applied HSNIC after empirical copula transformation (CHSNIC). Other dependence measures can also be applied after empirical copula transformation, resulting in measures that are also invariant to strictly monotone transformations. However, they (e.g., CMMD and CHSNIC) may fail to treat non-monotonic relations equally.

Reshef et al. (2011) proposed the concept of *equitability*, which states that a dependence measure should give equal importance to all relations: linear and nonlinear. For example, we expect a fair dependence measure to treat a perfectly linear relationship and a perfectly sinusoid relationship equally. Kinney and Atwal (2014) mathematically defined equitability by proposing *self-equitability*—under a nonlinear regression model with additive noise, a dependence measure should be invariant to any deterministic transformation of the marginal variables, under a nonlinear regression model with additive noise (a formal definition is provided in Definition 1, Section 2). A self-equitable dependence measure will treat all forms of relationships equally in the large data limit for the additive noise model. Kinney and Atwal (2014) proved that MI is self-equitable, and recommended its usage.

To choose among the many self-equitable dependence measures, we further propose a new *robust-equitability* concept such that the measure also treats all forms of relationships equally in the mixture noise model. That is, in a mixture distribution with p proportion of deterministic signal hidden in continuous independent background noise, the measure should reflect the signal strength p. The mixture noise model reflects real applications where measurements (features) are often corrupted with noise. For example, sensor data maybe corrupted by noise from hardware and environmental factors. Reshef et al. (2011, 2015b) considered equitability for a statistic. Our robust-equitability, as well as Kinney and Atwal (2014)'s self-equitability, is defined on the population quantity instead. Particularly, in the mixture distribution above, we define a dependence measure as *weakly-robust-equitable* if it is a monotone transformation of the proportion p, and is robust-equitable if it equals to p exactly.

In this paper, we show that among a class of self-equitable copula-based dependence measures, only *robust copula dependence* (RCD), defined as the total variational distance (the half of the L_1 distance) between copula density and uniform (independence) density, is also weakly-robust-equitable (and robust-equitable). Without referring to the copula density, RCD can be equivalently stated as the total variational distance between the probability distribution and the (independent) product of its marginal distributions, and is equivalent to the Silvey's Delta measure (Silvey, 1964). In the literature, the Silvey's Delta (RCD) was only cited as an abstract benchmark. Here, we propose a k-nearest-neighbor (KNN)-based estimator for RCD and prove its consistency. Besides the L_1 distance RCD, we also investigated properties of the L_2 distance between copula density and the uniform density (we call CD_2). CD_2 is the theoretical value of HSNIC in the large data limit (Fukumizu et al., 2007).

In addition, the robust-equitability study in this paper provides insights on the difficulty of estimating MI. Some authors studied the convergence of MI estimators by imposing the Hölder condition on the copula density. This Hölder condition, while being a standard condition for density estimations, does not hold for any commonly used copula (Omelka et al., 2009; Segers, 2012). Under a more realistic Hölder condition on the bounded region of copula density, we provide a theoretical proof that the mutual information (MI)'s minimax risk is infinite. This provides a theoretical explanation on the statistical difficulty of estimating MI observed by practitioners (Fernandes and Gloor, 2010; Reshef et al., 2011). Moreover, we prove that although both MI and CD_2 are self-equitable, they are not robust-equitable. Therefore, MI and CD_2 may not rank the features correctly by dependence strength in some cases, even in the large data limit. We confirm this phenomena on both synthetic and real-world data sets. In contrast, RCD is consistently estimable under the same condition. As for kernel-based dependence measures, HSIC and CMMD are neither self-equitable nor robust-equitable, HSNIC and CHSNIC are self-equitable but not robust-equitable and their estimators converge very slowly. Since RCD is the only measure that is both self-equitable and robust-equitable among these measures, it can be very useful for feature selection.

In summary, the contributions of this paper are: (1) the introduction of the concept of robust-equitability; (2) the identification of RCD as a dependence measure that is both self- and robust-equitable and the proposal of a practical consistent estimator for RCD; (3) theoretically proving that non-robust-equitable measures MI and CD_2 cannot be consistently estimated and showing that this can lead to incorrect selection of features when sample size is large or when sample sizes are unequal for different features; and finally, (4) demonstrating that the robust-equitable RCD is a better dependence measure for feature selection compared to existing dependence measures through experiments on synthetic and real-world data sets, in terms of robustness to function types, correctness in large sample size and correctness in unequal sample sizes. This paper is a substantially extended version of our conference version (Chang et al., 2016). In particular, this work includes the following additional materials: (1) a more complete treatment of the motivation and rationale of equitability definitions—we discuss the relationship of equitability to Renyi's theorems, to more copula-based dependence measures, and to independence tests; (2) a more complete theoretical treatment of the difficulty in estimation of mutual information (MI) versus RCD; in particular, we add a theorem showing that the difficulty of MI estimation is not due to the unboundedness of its definition, but is intrinsic due its being non-robust-equitable; and (3) more extensive empirical studies illustrating how equitability helps in feature selection.

The rest of this paper is organized as follows. In Section 2, we motivate the equitability concepts, discussing different equitability definitions and relationship to copula and Renyi's theorems. Particularly, we propose the concept of robust-equitability, and define a robust-equitable dependence measure called robust copula dependence. In Section 3, we prove MI and CD_2 are not consistently estimable. We also prove RCD can be consistently estimated and provide its estimators based on kernel density estimation (KDE) and k-nearest-neighbors (KNN). In Section 4, we provide feature selection experiments on synthetic and real data sets to demonstrate the advantage of RCD compared to existing dependence measures. We end with conclusions and discussions in Section 5.

2. A Robust-Equitable Dependence Measure

In this section, we investigate the theoretical properties of different dependence measures. In particular, we would like the dependence measures to have the following characteristics. We would like the dependence measures to rank a feature with less noise as having a stronger dependence with the response variable compared to features with more noise. We do not want measures that prefer a particular type of relationship (e.g., linear). Moreover, we do not want the measures to be too sensitive to sample size (i.e., when different features have unequal sample sizes, the dependence measure should not prefer a feature simply because it has more samples, but should still rank the features

based on the strength of the deterministic signal compared to noise). Note that it is becoming more common for databases to have features that have unequal sample sizes due to the prevalence of data collection from heterogeneous sources. For example, a clinical database may have more samples with clinical features compared to samples with genomic information. In such as setting, we would like to use all the data available to perform feature selection rather than to create equal sample sizes by throwing away samples from the larger set. In this paper, we formalize these ideas through the recently proposed equitability concept: We want to use dependence measures that reflect the noise level, regardless of relationship type.

2.1 Self-equitability, Rényi's Axioms and Copula-based Dependence Measures

Reshef et al. (2011) proposed that an equitable measure should "give similar scores to equally noisy relationships of different types." Kinney and Atwal (2014) mathematically defined self-equitability through invariance under all nonlinear relationships in the regression model

$$Y = f(X) + \epsilon, \tag{1}$$

where f is a deterministic function, ϵ is the random noise variable whose distribution may depend on $f(X)$ as long as ϵ has no additional dependence on X.

Definition 1 *A dependence measure $D[X;Y]$ is self-equitable if and only if $D[X;Y] = D[f(X);Y]$ whenever f is the function in model (1).*

Kinney and Atwal (2014) recommended usage of a self-equitable measure: mutual information (MI).

To understand self-equitability better, we notice that Definition 1 is very similar to Rényi's Sixth Axiom A6, both are defined through the invariance of the dependence measure under transformations. Rényi (1959) proposed seven axioms for dependence measures $D[X;Y]$. **(A1)** $D[X;Y]$ is defined for any random variables X and Y; **(A2)** symmetric $D[X;Y] = D[Y;X]$; **(A3)** $0 \leq D[X;Y] \leq 1$; **(A4)** $D(X;Y) = 0$ if and only if X and Y are statistically independent; **(A5)** $D(X;Y) = 1$ if either $X = f(Y)$ or $Y = g(X)$ for some Borel-measurable functions f and g; **(A6)** If f and g are Borel-measurable, one-one mappings of the real line into itself then $D[f(X);g(Y)] = D[X;Y]$; **(A7)** If the joint distribution of X and Y is bivariate Gaussian, with linear correlation coefficient ρ, then $D[X;Y] = |\rho|$.

For a symmetric dependence measure (satisfying Axiom A2), Axiom A6 can be rewritten as $D[f(X);Y] = D[X;Y]$ for all Borel-measurable f. Hence self-equitability is a weaker version requiring an extra assumption that f satisfies the model (1).

It is known that Rényi's maximum correlation coefficient is the only measure that satisfies all seven Rényi's Axioms. However, Rényi's maximum correlation coefficient has a number of major drawbacks, e.g., it equals 1 too often and is generally not effectively estimable (Schweizer and Wolff, 1981; Székely and Rizzo, 2009). Hence enforcing all seven axioms is often considered too strong a constraint on the dependence measure, while some axioms are often considered desirable. For example, HSIC is shown to satisfy the first four axioms by Gretton et al. (2005a) and Gretton et al. (2005b).

For comparison, another weakened version of Axiom A6 is to restrict the transformations to monotone functions (Schweizer and Wolff, 1981), but without imposing the regression model (1). We may call this version of Axiom (A6*) *weak-equitability*.

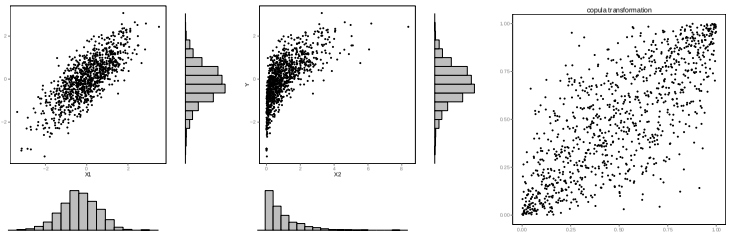


Figure 1: Left: Bivariate Gaussian with $\rho = 0.75$. Middle: Data with exponential marginal for X. Right: The Gaussian copula. The first two distributions have the same copula as in the right figure.

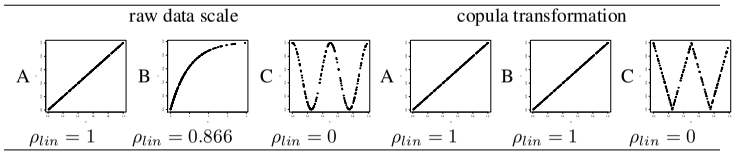


Table 1: Pearson correlation coefficient on three function relationships.

Definition 2 *A dependence measure $D[X;Y]$ is weak-equitable if and only if $D[X;Y] = D[f(X);Y]$ whenever f is a strictly monotone continuous deterministic function.*

The weak-equitable dependence measures treat all monotone (but not all nonlinear) relationships equally, and is a property shared by all copula-based dependence measures (e.g., CMMD and CHSNIC). Sklar's theorem ensures that, for any joint distribution function $F_{X,Y}(x,y) = Pr(X \leq x, Y \leq y)$, there exists a copula C—a probability distribution on the unit square $\mathcal{I}^2 = [0,1] \times [0,1]$—such that

$$F_{X,Y}(x,y) = C[F_X(x), F_Y(y)] \qquad \text{for all } x, y. \tag{2}$$

Here $F_X(x) = Pr(X \leq x)$ and $F_Y(y) = Pr(Y \leq y)$ are the marginal cumulative distribution functions (CDFs) of X and Y respectively. In other words, the copula C is the joint CDF of the two copula-transformed, uniformly distributed, variables $U = F_X(X)$ and $V = F_Y(Y)$. In this way, the copula decomposition separates the dependence from any marginal effects, and the copula C captures all the dependence between X and Y. Figure 1 shows the data from two distributions with different marginals but the same dependence structure.

Table 1 shows three simple examples and their respective copula transformations. We can see that the linear correlation ρ_{lin} prefers the linear relationship in (A). Applying on the copula-transformed variables on the right half of Table 1, ρ_{lin} (now equivalent to Spearman's ρ) becomes invariant to monotone transformation in (B), but still cannot capture the non-monotone nonlinear relationship in (C).

While copula-based dependence measures treats the monotone functions equally, equitability aims to also treat non-monotone functions equally. However, the original Rényi's Axiom A6 may be overly strong, and self-equitability aims to treat non-monotone functions equally only under the regression model (1).

We first consider some self-equitable copula-based dependence measures and further choose among these measures based on a new equitability definition in the next Subsection 2.2. Mutual information (MI), the recommended measure in Kinney and Atwal (2014), is self-equitable and is based on copula density $c(u, v)$,

$$\mathrm{MI} = \int_{I^2} \log[c(u, v)]c(u, v)dudv, \tag{3}$$

where I^2 is the unit square. We now consider a large class of self-equitable copula-based measures. Since the marginal variables X, Y are independent if and only if the corresponding copula distribution is uniform, we measure the dependence between X, Y through the distance between their copula distribution and the uniform distribution. Let the *Copula Distance* CD_α be the L_α distance between a copula density and the uniform copula density $\pi(u, v) = 1$.

$$\mathrm{CD}_\alpha = \int_{I^2} |c(u, v) - 1|^\alpha dudv, \quad \alpha > 0. \tag{4}$$

Combining Eq.4 in Fukumizu et al. (2007) and Eq.(4) here, CD_2 is the theoretical value of HSNIC in the large data limit. Our first result is that, the Copula Distance is self-equitable when $\alpha \geq 1$.

Lemma 1 *The Copula-Distance* CD_α *with* $\alpha \geq 1$ *is self-equitable.*

The proof follows from Theorems S3 and S4 of Kinney and Atwal (2014), since $g(x) = |x - 1|^\alpha$ is convex when $\alpha \geq 1$.

Remark: Schweizer and Wolff (1981) studied a class of dependence measures that are the L_α distance between a joint copula $C(u, v)$ and the uniform copula $\Pi(u, v) = uv$. The L_1, L_2 and L_∞ distance result in, the Wolf's σ, Hoeffding's Φ^2 and Wolf's κ respectively. Schweizer and Wolff (1981) showed that these measures satisfy a modified set of Rényi's Axioms, including Axiom (A6*) weak-equitability. In contrast to the Copula-Distance CD_α (L_α distance based on *copula densities*), these measures are based on the *cumulative distribution functions* and *are not self-equitable*. Since $C(u, v) = Pr(U \leq u, V \leq v)$ is the cumulative distribution function, such measures cumulate the deviation from independence from $u = 0$ to $u = 1$, and do not remain invariant for all nonlinear transformations f in model (1).

2.2 Robust Equitability

To select among the many self-equitable dependence measures, we want to consider additional equitability conditions. Some self-equitable dependence measures may not perform well in practice. For example, Rényi's maximum correlation coefficient (Rcor) satisfies the stronger Rényi's Axiom A6, thus it is also self-equitable. $Rcor(X; Y) = \sup_{f,g} \rho[f(X); g(Y)]$, where ρ is the linear correlation coefficient and the supremum is taken over all Borel-measurable functions f and g. Rcor has a number of major drawbacks, e.g., it equals 1 too often and is generally not effectively estimable (Schweizer and Wolff, 1981; Székely and Rizzo, 2009).

We observe the deficiencies more clearly in another self-equitable measure, the ideal dependence coefficient (IDC): $IDC(X; Y) = 0$ if X and Y are independent and $IDC(X; Y) = 1$ otherwise. IDC satisfies the first six Rényi's Axioms, and is self-equitable. It equals one for all dependent X and Y, providing no distinction of the dependence strength. IDC is only an abstract measure, the estimation of IDC is equivalent to *testing independence* between X and Y. However,

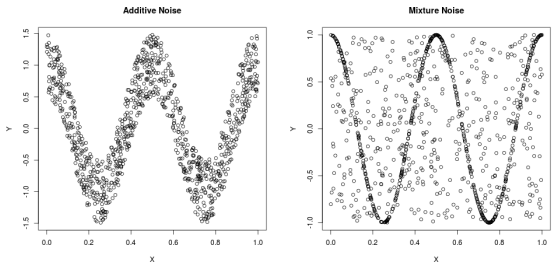


Figure 2: Left: Additive noise for self-equitability. Right: Mixture noise for robust-equitability.

for feature selection, IDC is not helpful at all because it provides no distinction of the dependence strength for different features. It is also hard to estimate. We wish for a new equitability criterion to exclude trivial dependence measures like IDC.

The self-equitability definition focuses on the regression $Y = f(X) + \epsilon$. However, in practice, this additive noise model does not capture all data types. In some cases, for example in sensor measurements, the deterministic signal is hidden in continuous background noise. Figure 2 illustrates these two types of noise. The Left subfigure shows additive noise on a deterministic sinusoidal function. The Right subfigure is the same deterministic signal on a uniform background noise. Mathematically, after the copula transformation, the second mixture noise model is described by a mixture copula: a continuous copula on the unit square I^2 is added to a deterministic signal C_s, which is a singular copula. Any copula can always be separated into a singular component and an absolutely continuous component (Nelsen, 2006, page 27). Independent background noise is represented by taking the absolutely continuous component as the independence copula $\Pi(u, v) = uv$ on I^2. Therefore, with p proportion of hidden deterministic relationship, the copula $C = pC_s + (1 - p)\Pi$. Here C_s is a singular copula representing the deterministic relationship, so that its support $\mathcal{S}$ has Lebesgue measure zero. The equitability in this mixture noise model means that the dependence measure should give the same value for all types of deterministic signal C_s.

Definition 3 *A dependence measure* $D[X; Y]$ *is robust-equitable if and only if* $D[X; Y] = p$ *whenever* (X, Y) *follows a distribution whose copula is* $C = pC_s + (1 - p)\Pi$*, for a singular copula* C_s*.*

Among the self-equitable Copula-Distances ($\alpha \geq 1$), L_1 distance is the special case that does reflect the proportion of deterministic relationship in the mixture copula.

$$CD_1 = p \int_{\mathcal{S}} C(du, dv) + \int_{\mathcal{I}^2 \setminus \mathcal{S}} |(1 - p) - 1|dudv = p(1) + p = 2p.$$

Therefore we define the scaled version of CD_1 as *robust copula dependence (RCD)*

$$RCD = \frac{1}{2}CD_1 = \frac{1}{2}\int_{\mathcal{I}^2} |c(u, v) - 1|dudv. \tag{5}$$

Lemma 2 *The robust copula dependence RCD is robust-equitable.*

Mathematically, RCD is the same as Silvey's Delta (Silvey, 1964):

$$\Delta = \int_{\phi>1} [p(x,y) - p_X(x)p_Y(y)]dxdy,$$

where p_X and p_Y are the marginal probability densities for X and Y, p is the joint probability density for X and Y, and $\phi(x,y) = p(x,y)/[p_X(x)p_Y(y)]$. We write equation (5) in terms of the absolutely continuous copula density for ease of understanding. When part of the copula is singular, the RCD in (5) can be defined as in Silvey (1964), interpreting ϕ as the Radon-Nikodym derivative of the joint distribution with respect to a dominating probability measure which does cover the possibility of singularity. Alternatively, for a mixture copula C, the RCD can be defined as the limit of $\lim_{m\to\infty} RCD(C_m)$ for equation (5) on any sequence of continuous copulas $\{C_1, C_2, ...\}$ that converges to C. The convergence means that $\lim_{m\to\infty} \|C_m - C\|_1 := 2 \lim_{m\to\infty} \sup_A |C_m(A) - C(A)| = 0$, where the supremum is taken over all Borel sets A. A second way of interpretation is helpful in thinking about why CD_α, when $\alpha > 1$, can not be made robust-equitable and why this leads to statistical difficulties in estimation which we will discuss in detail in the next section.

Roughly speaking, for the mixture copula $C = pC_s + (1-p)\Pi$, the copula density for the absolutely continuous component is $c_c(u,v) = 1-p$, while we can imagine $c_s(u,v)$ as an abstract copula density for the singular component such that $\int_{\mathcal{B}} c_s(u,v)dudv := \int_{\mathcal{B}} C(du,dv) = C_s(\mathcal{B})$ for any subset $\mathcal{B} \subset \mathcal{S}$. Since $\mathcal{S}$ has Lebesgue measure zero, $c_s(u,v) = \infty$ for $(u,v) \in \mathcal{S}$ so that c_s is not a proper density, but rather an abstract limit of the sequence $\lim_{m\to\infty} c_{m,s}$. Here for any convergent sequence of continuous copulas $\{C_1, C_2, ...\}$ above, $c_{m,s}(u,v) = c_m(u,v) - c_c(u,v)$ is the continuous copula density that approaches the abstract $c_s(u,v)$. For any open set $\mathcal{B}_\mathcal{O} \supset \mathcal{B}$, $C_s(\mathcal{B}_\mathcal{O}) = \int_{\mathcal{B}_\mathcal{O}} c_s(u,v)dudv := \lim_{m\to\infty} \int_{\mathcal{B}_\mathcal{O}} c_{m,s}(u,v)dudv$, and $C_s(\mathcal{B}) = \lim_{\mathcal{B}_\mathcal{O}\to\mathcal{B}} C_s(\mathcal{B}_\mathcal{O})$. Hence for any $\alpha > 1$, $\int_{\mathcal{B}}[c_s(u,v)]^\alpha dudv = \int_{\mathcal{S}}[c_s(u,v)]^{\alpha-1}c_s(u,v)dudv = \int_{\mathcal{S}} \infty c_s(u,v)dudv = \infty$. So that $CD_\alpha = \infty$ whenever $\alpha > 1$ and $p > 0$. Similarly, $MI = \infty$ for all $p > 0$. They do not distinguish the dependence strength in the mixture distribution according to the signal proportion p, and can not be transformed to be robust-equitable as they over-emphasize the singular component (high copula density region).

If the dependence measure does not equal p exactly but is a monotone function of p, then we can scale it to get a robust-equitable version, and call it weakly-robust-equitable.

Definition 4 *A dependence measure $D[X;Y]$ is weakly-robust-equitable if and only if $D[X;Y]$ is a strictly monotone function of p whenever (X,Y) follows a distribution whose copula is $C = pC_s + (1-p)\Pi$, for a singular copula C_s.*

Lemma 3 *The Copula-Distance CD_α is weakly-robust-equitable if and only if $\alpha \leq 1$.*

When $\alpha > 1$, since $CD_\alpha = \infty$ whenever $p > 0$, those are not weakly-robust-equitable. When $\alpha < 1$, $\int_{\mathcal{S}}[c_s(u,v)]^\alpha dudv = \int_{\mathcal{B}}[c_s(u,v)]^{\alpha-1}c_s(u,v)dudv = 0$ so that the contribution from the singular region $\mathcal{S}$ is zero. In these cases,

$$CD_\alpha = 0 + \int_{\mathcal{I}^2\setminus\mathcal{S}} |(1-p) - 1|^\alpha dudv = p^\alpha$$

is weakly-robust-equitable. And CD_1 is weakly-robust-equitable from Lemma 2.

The self-equitability for additive noise model requires that $\alpha \geq 1$, while the weakly-robust-equitablity in the mixture noise model requires that $\alpha \leq 1$. Hence only $\alpha = 1$ satisfies the equitability condition in both noise models, and resulting in the robust-equitable RCD.

2.2.1 RCD, OTHER EQUITABILITY DEFINITIONS AND RÉNYI'S AXIOM

Having introduced our measure, RCD, and the robust-equitability definition, we can further compare them to the other equitability definitions in the literature. Reshef et al. (2011) considers equitability as the ability of a statistic $\hat{D}$ to approximately reflect the nonlinear R^2 over different relationships. They proposed a statistic MIC and demonstrate numerically such equitability through simulated examples. Kinney and Atwal (2014) propose to formalize such concept for the population parameter $D[X;Y]$ such that an R^2-*equitable* measure equals $g(R^2[f(X),Y])$ in the additive noise model (1) $Y = f(X) + \epsilon$, and showed that no nontrivial dependence measure can satisfy this R^2-equitability. The self-equitability definition is proposed as an alternative. Murrell et al. (2014) pointed out that such impossibility results are due to the non-identifiability due to the specification of ϵ term, allowing ϵ to possibly depend on $f(X)$. Under such specification, for example, a noiseless parabola can be realized as a noisy version of a noiseless linear relationship (Murrell et al., 2014). Reshef et al. (2015b) propose another formal equitability framework through interpretable intervals of a statistic under additive homoscedastic noises for both X and Y.

Our robust-equitability definition shares some common characteristics with both Kinney and Atwal (2014)'s and Reshef et al. (2015b)'s approach respectively. Similar to Kinney and Atwal (2014), our robust-equitability focuses on the population quantity $D[X;Y]$ instead of a statistic $\hat{D}[X;Y]$. This allows proof of theoretical equitability properties for specific dependence measures, as the statistical estimation error $\hat{D} - D$ can be kept as a separate issue. Robust-equitability does have implications on the statistical estimation error bounds which will be discussed in the next section. Self-equitability and robust-equitability focus on the invariance of $D[X;Y]$ as in Rényi's Axiom A6, but for different noise models. Reshef et al. (2015b) and our robust-equitability definition each focuses on a noise model to avoid the non-identifiability issue in Kinney and Atwal (2014)'s model: additive homoscedastic noise and mixture uniform noise respectively. Under each model, there is a clearly identifiable quantity of interest: the nonlinear R^2 and the mixture proportion p respectively.

Furthermore, our RCD satisfies the first five Rényi's Axioms. Particularly, $RCD = 0$ if and only if X and Y are statistically independent, $RCD = 1$ if X and Y are related through a deterministic relationship. And RCD is symmetric in that $RCD[X;Y] = RCD[Y;X]$. Notice that this symmetric property is an appropriate requirement for feature selection with the filtering method mRMR (Peng et al., 2005). Given a data set $\mathbf{X} \in \mathbb{R}^{n\times d}$, where n is the number of samples, d is the number of features, and target variable $Y \in \mathbb{R}^{n\times 1}$, let X_i denote the i-th feature, mRMR finds from the d-dimensional feature space, $\mathbb{R}^d$, a subspace of m features that optimally characterize Y, by solving the following optimization problem:

$$\max_S \frac{1}{|S|} \sum_{X_i\in S} D[X_i;Y] - \frac{1}{|S|^2} \sum_{X_i,X_j\in S} D[X_i;X_j], \tag{6}$$

where S is the optimal feature set, the first term maximizes relevance and the second term minimizes redundancy. Notice that this method only requires bivariate dependence between different features $D[X_i;X_j]$ in addition to the bivariate dependence between each feature and response variable $D[X_i;Y]$. The feature selection results of mRMR with an asymmetric dependence measure would depend on how the features are ordered, which is an undesirable characteristic.

RCD satisfies self-equitability, which is a weakened version of the sixth Rényi's Axiom. The Rényi's Axiom A7 requires the dependence measure to agree with the natural quantity of $|\rho|$ for bivariate Gaussian distributions (which corresponds to a linear regression model). Our RCD does not satisfy that, but instead agree with $|\rho|$ for the mixture noise setting with a linear deterministic relationship, since in that case $p = |\rho|$. Our robust-equitability definition requires the measure equals p exactly, which provides an easy interpretation in that it is an equitable extension of Pearson's correlation $|\rho|$ to all forms of hidden nonlinear deterministic relationships. It is not essential to require the exact equality to p, as equaling to a monotone function of p (weakly-robust-equitability) would enable a robust-equitable version of the measure through a transformation. However, precisely equaling to p is nice due to the above easier interpretation. Notice that $R^2 = \rho^2$ for the additive noise regression model with a linear relationship (bivariate Gaussian distribution). Hence the nonlinear R^2 can be similarly considered as an equitable extension of Pearson's correlation in the additive noise model to all forms of nonlinear regression relationships. However, unlike p, R^2 does not satisfy the symmetric property since regressing Y on X and regressing X on Y do not give the same value.

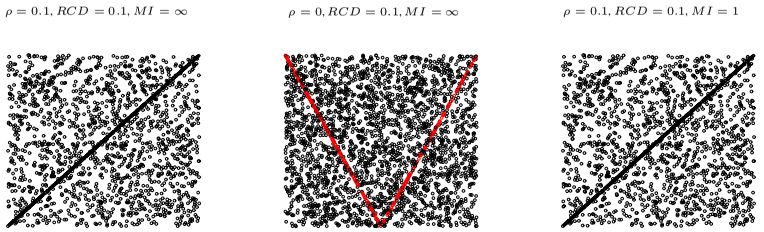
For discrete random variables, Equation (5) corresponds to the Kolmogorov dependence measure in the pattern recognition literature (Vilmansen, 1972, 1973; Ekdahl and Koski, 2006) and also known as the Mortara dependence index (Bagnato et al., 2013). In the discrete case, the measure has a maximum value less than 1. In contrast, $RCD = 1$ when X and Y are deterministically related. For continuous random variables X and Y, Silvey's Delta has been cited only as an abstract concept and no practical estimator was used in the literature for data analysis. The new name, *Robust Copula Dependence* (RCD), emphasizes the fact that it is a robust-equitable copula-based dependence measure.

2.3 Testing Independence Versus Estimation Errors of Dependence Measures

In practice, feature selection is based on an estimator $\hat{D}(X;Y)$ on the data set, since the exact value of the dependence measure $D(X;Y)$ is unknown to the user. Hence the feature selection results are affected by the estimation error $\hat{D} - D$. The estimation error also needs to be studied.

An estimator $\hat{D}[X;Y]$ is often used to test the independence between X and Y. Some studies compare different dependence measures $D[X;Y]$s by the power of independence testing using their corresponding estimators $\hat{D}[X;Y]$s (Reshef et al., 2011; Simon and Tibshirani, 2011; Reshef et al., 2015a). However, while independence testing is related to the estimation of dependence measures $D[X;Y]$, the power of the independence test is not a proper way of comparing dependence measures $D[X;Y]$. In fact, as mentioned in Section 2.2, the independence test corresponds to an estimation of the trivial measure IDC: $IDC(X;Y) = 0$ if X and Y are independent and $IDC(X;Y) = 1$ otherwise. Thus the power comparison is comparing the performance of $\hat{D}[X;Y]$ in estimating $IDC(X;Y)$ rather than estimating its corresponding parameter $D[X;Y]$. Also, good independence test statistics may not have corresponding interpretable dependence measures (Sun and Zhao, 2014).

As Reshef et al. (2015b) pointed out, the estimator $\hat{D}[X;Y]$ for equitable $D[X;Y]$ is most powerful at testing the hypothesis if the signal strength exceeding a threshold $D[X;Y] \geq D_0$, rather than being most powerful at testing independence $D[X;Y] = 0$. Besides simply testing for independence, dependence measures serve another important purpose: ranking the strength of the dependence relationships. For example, in the World Health Organization (WHO) data set in Reshef et al. (2011) the vast majority of the hundreds of the variables show dependence with other variables.



(a) (b) (c)

Figure 3: Hidden in background noise are 10% (red colored) data on a deterministic curve in (a) and (b), on a narrow strip around the line in (c).

So the independence tests do not provide much information there. To achieve sparse representation (feature selection), it is important to pick out the strongest dependence relationships. Equitable dependence measure ensures that the signal strength is reflected, rather than the functional form.

The robust-equitability definitions also have implications on the estimation errors. Some self-equitable measures can equal one too often, the extreme case being the IDC. When the dependence measure $D[X;Y]$ equals one for too many type of distributions, it does not distinguish dependence strength among them, and also makes its estimation difficult. Robust-equitability excludes such measures. In the next section, we theoretically show that robust-equitable RCD is intrinsically much easier to estimate than non-robust-equitable (but only self-equitable) MI and CD_2. This is due to the instability in the theoretical values of MI and CD_2. The following examples illustrate the difference between self-equitability (Figure 3: a versus b) and robust-equitability (Figure 3: a versus c). The self-equitable measures (RCD, MI), unlike ρ, have the same value in Figures 3(a) and 3(b) where each has 10% deterministic data on two different curves. In Figure 3(c), 10% of the data fall around (rather than exactly on) the line in a very small strip of area $0.1/exp(10) = 4.5 \times 10^{-6}$, which is very close to the distribution in Figure 3(a). The robust-equitable RCD values are very similar (differ only in 10^{-6} order) in Figures 3(a) and 3(c), but the MI values changes from ∞ to 1. Since these two almost indistinguishable distributions result in very different theoretical MI values (∞ and 1), no estimator can do well.

3. Statistical Estimation Errors

In this section, we study the estimation errors theoretically, and provide a practical estimator for RCD. In particular, we analyze the statistical estimation error $\hat{D} - D$.

3.1 The Inconsistency Results on Estimation of Mutual Information and CD_2

We theoretically show that MI and CD_2 are much harder to estimate compared to the robust-equitable RCD. We formally quantify the estimation difficulty through the minimax convergence rate over a family $\mathfrak{C}$.

Denote $\mathbf{z} = (u, v)$. Let $\mathfrak{C}$ be the family of continuous copulas with the density satisfying the following Hölder condition on the region where $c(\mathbf{z})$ is bounded above by some constant $M > 1$, denoted as A_M:

$$|c(\mathbf{z}_1) - c(\mathbf{z}_2)| \leq M_1 \|\mathbf{z}_1 - \mathbf{z}_2\|_{l_1}, \tag{7}$$

for a constant M_1 and for all $\mathbf{z}_1 \in A_M$, $\mathbf{z}_2 \in A_M$, and $\|\cdot\|_{l_1}$ denotes the l_1 norm.

The estimation of MI has been studied extensively in the literature. Over all distributions, even discrete ones, no uniform rate of convergence is possible for MI (Antos and Kontoyiannis, 2001; Paninski, 2003). On the other hand, many estimators were shown to converge to MI for every distribution. These results are not contradictory, but rather common phenomena for many parameters. The first result is about the uniform convergence over all distributions, while the second result is about the pointwise convergence for each distribution. The first restriction is too strong, while the second restriction is too weak. The difficulty of estimating a parameter needs to be studied for uniform convergence over a properly chosen family.

As MI is defined through the copula density, it is natural to consider the families generally used in density estimation literature. Starting from Farrell (1972), it is standard to study the minimax rate of convergence for density estimation over the class of functions whose m-th derivatives satisfy the Hölder condition. Since the minimax convergence rate usually is achieved by the kernel density estimator, it is also the optimal convergence rate of density estimation under those Hölder classes. Generally, with the Hölder condition imposed on the m-th derivatives, the optimal rate of convergence for two-dimensional kernel density estimator is $n^{-(m+1)/(2m+4)}$ (Silverman, 1986; Scott, 1992).

Therefore, when studying the convergence of MI estimators, it is very tempting to impose the Hölder condition on the copula density. In fact, imposing the Hölder condition (7) on the whole I^2, Liu et al. (2012) showed that the kernel density estimation (KDE) based MI estimator converges at the parametric rate of $n^{-1/2}$. Pál et al. (2010) also considered similar Hölder condition when they studied the convergence of k-nearest-neighbor (KNN) based MI estimator. However, such a condition is usually too strong for copula density, thus these results cannot fully reflect the true difficulty of MI estimation. When $c(u, v)$ is unbounded, the Hölder condition (7) cannot hold for the region where $c(u, v)$ is big. Hence imposing this Hölder condition (7) on the whole I^2 would exclude many commonly used continuous copula densities (e.g., Gaussian, student-T, etc.) since their densities are unbounded (Omelka et al., 2009; Segers, 2012). Therefore, we impose it only on the region where the copula density is small. Specifically, we assume that the Hölder condition holds only on the region $A_M = \{(u, v) : c(u, v) < M\}$ for a constant $M > 1$. Then this condition is satisfied by all common continuous copulas in the book by Nelsen (2006). For example, all Gaussian copulas satisfy the Hölder condition (7) on A_M for some constants $M > 1$ and $M_1 > 0$. But no Gaussian copulas, except the independence copula Π, satisfy the Hölder condition (7) over the whole $\mathcal{I}^2$.

Theorem 4 *Let $\widehat{MI}$ be any estimator of the mutual information* MI *based on the observations $\mathbf{Z}_1 = (U_1, V_1)$, ..., $\mathbf{Z}_n = (U_n, V_n)$ from a copula distribution $C \in \mathfrak{C}$. And let $\widehat{CD}_\alpha$ be any estimator of the CD_α in equation (4). Then*

$$\begin{aligned} \sup_{C \in \mathfrak{C}} E[|\widehat{MI}(C) - \mathrm{MI}(C)|] &= \infty, \text{ and} \\ \sup_{C \in \mathfrak{C}} E[|\widehat{CD}_\alpha(C) - CD_\alpha(C)|] &= \infty, \text{ for any } \alpha > 1. \end{aligned} \tag{8}$$

The detailed proof is provided in AppendixA. This theorem states that MI and CD_2 cannot be consistently estimated over the family $\mathfrak{C}$. This result does not depend on the estimation method used, as it reflects the theoretical instability of these quantities. There are many estimators for MI: kernel density estimation (KDE) (Moon et al., 1995), the k-nearest-neighbor (KNN) (Kraskov et al., 2004), maximum likelihood estimation of density ratio (Suzuki et al., 2009). However, practitioners are often frustrated by the unreliability of these estimation (Fernandes and Gloor, 2010; Reshef et al., 2011). This theorem provides a theoretical explanation.

Notice that the inconsistency results over this family $\mathfrak{C}$ is not due to the unboundedness of MI and CD_2. They can be transformed into correlation measures with values between 0 and 1 (Joe, 1989): $MIcor = \sqrt{1 - e^{-2MI}}$ and $\phi cor = \sqrt{CD_2/(1 + CD_2)}$. The $MIcor$ is known as the Linfoot correlation in the literature (Speed, 2011). We use the name $MIcor$ to indicate it as the scaled version of MI. The next theorem showed that $MIcor$ cannot be consistently estimated over the family $\mathfrak{C}$ also.

Theorem 5 *Let $\widehat{MIcor}$ be any estimator of $MIcor$ based on the observations $\mathbf{Z}_1 = (U_1, V_1)$, ..., $\mathbf{Z}_n = (U_n, V_n)$ from a copula distribution $C \in \mathfrak{C}$. Then*

$$\sup_{C \in \mathfrak{C}} E[|\widehat{MIcor}(C) - MIcor(C)|] \quge \quad a_2 > 0, \tag{9}$$

for a positive constant a_2.

The detailed proof is provided in Appendix B.

The estimation difficulty of these dependence measures is due to their lack of smoothness related to being not weakly-robust-equitable. Reshef et al. (2015a, Section 4) proved a similar lack of smoothness of MI and $MIcor$, while their proposed statistic MIC may be considered a smoothed version which shows equitable behavior under their framework. Our results are stronger in that: (a) our results establish the statistical difficulty of estimation via minimax rate, and (b) our results apply to a broader class of dependence measures.

3.2 The Consistent Estimation of RCD

The equitability definitions and error analysis above assume a bivariate dependence measure. In this section, we will state the estimation results for a general d-dimensional RCD in equation (5): $RCD = \frac{1}{2}\int_{\mathcal{I}^d} |c(\mathbf{z}) - 1| d\mathbf{z}$ for d-dimensional $\mathbf{z}$. That is, for $\mathbf{z} = (z_1, ..., z_d)$, the copula transformation changes each dimension to uniformly distributed variables $u_j = F_u^{-1}(z_j)$, $j = 1, ..., d$. Then $\int |c(\mathbf{z}) - 1| d\mathbf{z} := \int ... \int |c(u_1, ..., u_d) - 1| du_1 ... du_d$. The robust-equitability definition can be easily changed to the d-dimensional mixture copula with a singular component and the d-dimensional uniform distribution. And, in the d-dimensional case, the calculation above equation (5) still holds so that RCD is robust-equitable. Notice that other equitability definitions such as self-equitability is only defined for the bivariate case, and the filtering feature selection method mRMR also uses only the bivariate dependence.

Mathematically, MI (and $MIcor$) is unstable because it overweighs the region with large density $c(\mathbf{z})$ values. From equation (3), MI is the expectation of $\log[c(\mathbf{z})]$ under the true copula distribution $c(\mathbf{z})$. In contrast, RCD in (5) takes the expectation at the independence case Π instead. Even if $c(\mathbf{z})$ cannot be consistently estimated in the region A_M^c (the complement of A_M), its error contribution to $\widehat{RCD}$ can be bounded. The following theorem, which is proved in Appendix C, shows the result for the KDE estimator for RCD.

Theorem 6 *Let the KDE estimator of the d-dimensional copula density based on observations* $\mathbf{Z}_1, \cdots, \mathbf{Z}_n$ *be*

$$\hat{c}_{kde}(\mathbf{Z}) = \frac{1}{nh^d} \sum_{i=1}^{n} K(\frac{\mathbf{Z}_i - \mathbf{Z}}{h}). \tag{10}$$

We assume the following conditions:

- *The bandwidth* $h \to 0$ *and* $nh^d \to \infty$.
- *The kernel* K *is non-negative and has a compact support in,* $\mathbb{B}_0 = \{\mathbf{Z} : \|\mathbf{Z}\|_{l_2} \leq 1\}$, *the d-dimensional unit ball centered at* 0.
- *The kernel* K *is bounded.* $M_K = \max_{s \in \mathbb{B}_0} K(s)$, $\int_{\mathbb{B}_0} K(s)ds = 1$, $\mu_2^2 = \int_{\mathbb{B}_0} K^2(s)ds < \infty$

Then the plugged-in estimator $\widehat{RCD} = RCD(\hat{c}_{kde})$ *has a risk bound*

$$\sup_{C \in \mathfrak{C}} E[|\widehat{RCD} - RCD|] \leq M_1 h + \frac{\sqrt{2}\mu_2}{\sqrt{n}h^{\frac{d}{2}}} + O(\frac{1}{nh^d}). \tag{11}$$

In addition to the KDE based RCD estimator, we can estimate RCD consistently by plugging in the KNN estimator (Loftsgaarden and Quesenberry, 1965) of the copula density: $\hat{c}(\mathbf{z}) = k/n/A_{r(k,n)}$ using copula based observations $\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_n$. Here $r(k,n)$ is the distance from (d-dimensional) $\mathbf{z}$ to the k-th closest of $\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_n$, and A_r is the volume of the d-dimensional hyper-ball with radius r. Then $\widehat{RCD} = RCD(\hat{c}) = \sum_{\hat{c}(\mathbf{Z}_i)>1}[1 - 1/\hat{c}(\mathbf{Z}_i)]/n$.

Theorem 7 *Assuming* c *in* $\mathfrak{C}$ *has bounded continuous second order derivative in* A_M, $k \to \infty$ *and* $(k/n) \to 0$ *when* $n \to \infty$. *Then the plugged-in KNN estimator* $\widehat{RCD} = RCD(\hat{c})$ *has a risk bound*

$$\sup_{C \in \mathfrak{C}} E[|\widehat{RCD} - RCD|] \leq \tilde{c_1}(\frac{k}{n\epsilon})^{\frac{2}{d}} + \frac{\tilde{c_2}}{\sqrt{k}} + 2\epsilon, \tag{12}$$

for some finite constants $\tilde{c_1}$ *and* $\tilde{c_2}$, *and* $\epsilon = \epsilon(n)$ *is any sequence converging to* 0 *slower than* k/n.

Here, the extra technical assumption on the second order derivative allows a simpler proof (provided in Appendix D) by citing formulas in Mack and Rosenblatt (1979). Without it, RCD can still be estimated consistently as in Theorem 6. The error bound (12) is minimized by $\epsilon = (k/n)^{2/(d+2)}$ and $k = n^{4/(d+6)}$. Hence, in the bivariate ($d = 2$) case, we have $k = O(\sqrt{n})$. Simulations in Appendix E suggests a practical estimator with $k = 0.25\sqrt{n}$.

When RCD is estimated well under a sample size, further increasing the sample size does not change its estimation value much. In contrast, the estimated MI and CD_2 values can continuously change by a large margin as sample size increases, altering the ranking of features, sometimes to the wrong order.

Computational Complexity of KNN-based RCD Estimator The computation of KNN-based RCD is dominated by empirical ranking for each dimension and k-nearest neigbhor search for each sample. The ranking can be solved by mergesort, which costs $\mathcal{O}(dn \log n)$ for d dimensions (Cormen, 2009). The k-nearest neighbor search can be solved using k-d tree construction, which has $\mathcal{O}(kdn \log n)$ complexity when d is small (no larger than 20) (Bentley, 1975). However, the complexity can increase to $\mathcal{O}(kdn^2)$ if d becomes large. Therefore, the overall complexity is $\mathcal{O}(kdn \log n)$ for low dimension data and $\mathcal{O}(kdn^2)$ for high dimension data.

4. Experimental Results

In this section, we empirically verify the properties of RCD in our theoretical analysis.

We first check the estimation errors for RCD in synthetic experiments with additive noise and mixture noise respectively. For each type of noise, we simulate data with several different relationships so as to show the effect of self-equitability and robust-equitability respectively. In particular, we compare the RCD estimator with an MI estimator based on the same density estimation. Due to the non-robust-equitability of MI, in the mixture noise cases, the MI estimator varies widely with the sample sizes. In contrast, RCD converges as sample sizes increases. Therefore, MI may provide misleading ranking of features with unequal sample sizes. Also, the ranking between relationships with the two different noise types are greatly affected by the sample sizes under MI, while ranking under RCD remains relatively stable.

We then conduct several synthetic experiments to illustrate the properties in feature selection, and then show that similar patterns exist on real-world data sets. In Section 4.3, we show that: (1) Non-self-equitable dependence measures may provide misleading ranking under additive noise; (2) Non-robust-equitable dependence measures may provide misleading ranking under mixture noise when features have unequal sample sizes; (3) The ranking by non-robust-equitable dependence measures between the two types of noises are sensitive to sample size. Section 4.4 shows that similar behavior occurs in three real data examples. This confirms that the advantages of self-equitable and robust-equitable dependence measures are not just theoretical, but are real in some practical situations.

Furthermore, we compare the performance of feature selection by the filter method mRMR (Peng et al., 2005) as a feature search strategy and using various dependence measures as measures of relevance and redundancy (refer to Equation (6)). We conducted the mRMR method first on synthetic examples in Section 4.5, to illustrate why non-self-equitability or non-robust-equitability could lead to misleading results. We then perform mRMR on nine benchmark data sets from the UCI Data Repository (Lichman, 2013) in Section 4.6. Notice that the feature selection performance on a particular data set is affected by the type of existing relationships and the type of predictors used. For example, Pearson's correlation with a linear regression predictor should perform best if linear relationship is dominant in a data set. For a fair comparison, we measure performance by 10-fold cross-validated MSE of spline regression, a general nonlinear predictor (Friedman, 1991), using the selected features. Self-equitability and robust-equitability lead to equitable and robust feature selection. Hence RCD should provide stable performance across different types of data, not necessarily best in each situation. However, over many data sets with different types of nonlinear relationships, robust-equitable RCD would provide best average performance as confirmed on these nine benchmark data sets.

There are some parameters to be set for computing various dependence measures. For kernel based measures, we follow the settings used by Fukumizu et al. (2007). For HSNIC, we set the regularization parameter $\epsilon_n = 10^{-5}n^{-3.1}$ to satisfy the convergence guarantee given by Theorem 5 from Fukumizu et al. (2007). As discussed in the previous section, we set $k = 0.25\sqrt{n}$ for the k-NN estimator of MI, RCD and CD_2.

4.1 Estimation Errors and Equitability

In this section, we study the estimation errors of our RCD estimates through synthetic experiments, and examine the equitability effect in ranking features. We first generate data from four different

type	level	Linear		Square Root		Cubic		Quadratic	
		1k	10k	1k	10k	1k	10k	1k	10k
add	0.4	0.35(0.01)	0.38(0.00)	0.36(0.01)	0.38(0.00)	0.36(0.01)	0.38(0.00)	0.35(0.01)	0.37(0.00)
	0.6	0.54(0.01)	0.58(0.00)	0.54(0.01)	0.58(0.00)	0.55(0.01)	0.58(0.00)	0.53(0.01)	0.57(0.00)
	0.8	0.76(0.00)	0.79(0.00)	0.76(0.01)	0.79(0.00)	0.76(0.01)	0.79(0.00)	0.75(0.01)	0.78(0.00)
mix	0.4	0.43(0.02)	0.43(0.01)	0.42(0.02)	0.42(0.01)	0.42(0.02)	0.42(0.01)	0.39(0.02)	0.42(0.01)
	0.6	0.62(0.02)	0.62(0.00)	0.61(0.02)	0.62(0.00)	0.61(0.02)	0.62(0.00)	0.59(0.01)	0.61(0.01)
	0.8	0.81(0.01)	0.81(0.00)	0.80(0.01)	0.81(0.00)	0.80(0.01)	0.81(0.00)	0.78(0.01)	0.80(0.00)

Table 2: The expected values of RCD estimates (with standard deviation in parenthesis) based on 100 simulations, under various functional types, sample sizes, noise types and noise levels.

type	level	Linear		Square Root		Cubic		Quadratic	
		1k	10k	1k	10k	1k	10k	1k	10k
add	0.4	0.77(0.01)	0.77(0.00)	0.77(0.01)	0.77(0.00)	0.75(0.01)	0.75(0.00)	0.00(0.04)	0.00(0.01)
	0.6	0.90(0.00)	0.90(0.00)	0.90(0.01)	0.90(0.00)	0.87(0.01)	0.87(0.00)	0.00(0.04)	0.00(0.01)
	0.8	0.98(0.00)	0.98(0.00)	0.96(0.00)	0.96(0.00)	0.91(0.00)	0.91(0.00)	0.00(0.04)	0.00(0.01)
mix	0.4	0.40(0.03)	0.40(0.01)	0.33(0.03)	0.33(0.01)	0.34(0.03)	0.33(0.01)	-0.01(0.03)	0.00(0.01)
	0.6	0.60(0.03)	0.60(0.01)	0.52(0.03)	0.51(0.01)	0.51(0.03)	0.5(0.01)	0.00(0.03)	0.00(0.01)
	0.8	0.80(0.02)	0.80(0.01)	0.73(0.02)	0.72(0.01)	0.69(0.03)	0.69(0.01)	0.00(0.04)	0.00(0.01)

Table 3: The expected values of ρ estimates (with standard deviation in parenthesis) based on 100 simulations.

functional types: linear, square root, cubic, and quadratic (cases A, C, D, F in Table 11 of Appendix E) with two sample sizes of $n = 1000$ and $n = 10000$ respectively. Also, data with two different noise paradigms and three noise levels are tested with three measures, RCD, (non-self-equitable) ρ and (non-robust-equitable) CD_2.

As we can see from Table 2, the standard deviation of the RCD estimates are small, and the expected value of RCD converges to the true values as sample size increases. The expected values of RCD was similar for the different functional relationships. They are closer to the true values under the mixture noise than under the additive noise. For either noise type, the estimates are very accurate for sample size $n = 10,000$. Although there are some random estimation errors, the RCD estimates would not miss-rank features with moderate difference in true RCD values. That is, it never ranks features with real RCD = 0.2 as more dependent than features with real RCD = 0.4, under samples $n = 1000$ or $n = 10,000$. So the RCD estimates can be used to provide reliable dependence ranking that do not change dramatically under sample sizes $n = 1000$ versus $n = 10,000$.

In contrast, the non-self-equitable Pearson's ρ values depend on the functional relationship. Under the mixture noise, the estimated ρ values are close to the mixture proportion only for the linear relationship. Hence its ranking of features is heavily influenced by the functional relationships. Particularly, it fails to detect the quadratic relationship in the last column of Table-3. Even when 80% of data follows the deterministic quadratic relationship, it is still ranked as less dependent than the features with other functional relationships (even if the other features has only 40% deterministic mixture proportion).

For the non-robust-equitable CD_2 in Table 4, its value changes dramatically under sample sizes $n = 1000$ versus $n = 10,000$, especially under mixture noise. This demonstrates that sample size affects the ranking by CD_2 in contrast to the ranking by the robust-equitable RCD. If the

type	level	Linear		Square Root		Cubic		Quadratic	
		1k	10k	1k	10k	1k	10k	1k	10k
add	0.4	0.94(0.04)	1.23(0.02)	0.96(0.04)	1.28(0.02)	1.08(0.04)	1.45(0.02)	0.88(0.03)	1.19(0.01)
	0.6	1.7(0.04)	2.25(0.02)	1.81(0.05)	2.46(0.02)	2.19(0.06)	3.11(0.03)	1.6(0.04)	2.27(0.02)
	0.8	3.62(0.06)	5.06(0.03)	3.72(0.07)	5.47(0.03)	4.35(0.10)	7.52(0.05)	2.93(0.06)	4.96(0.03)
mix	0.4	1.29(0.09)	2.62(0.07)	1.30(0.09)	2.69(0.072)	1.24(0.10)	2.59(0.06)	0.83(0.06)	1.71(0.04)
	0.6	2.70(0.15)	5.78(0.10)	2.73(0.15)	5.89(0.09)	2.66(0.17)	5.76(0.10)	1.69(0.08)	3.63(0.06)
	0.8	4.75(0.15)	10.24(0.11)	4.79(0.17)	10.34(0.11)	4.75(0.16)	10.24(0.11)	2.87(0.08)	6.27(0.07)

Table 4: The expected values of CD_2 estimates (with standard deviation in parenthesis) based on 100 simulations.

two features have unequal sample sizes, then the feature with larger sample size has a built-in preference by CD_2. For each of the functional types, CD_2 ranks a feature with mixture proportion (of deterministic data) 0.6 but large sample size $n = 10000$ as more dependent than a strongly dependent feature with mixture proportion 0.8 but smaller sample size $n = 1000$. Also, for features with different noise types, their ranking are inconsistent when sample size changes. For example, for the linear relationship under sample size $n = 1000$, CD_2 ranks the feature with RCD = 0.6 mixture noise as less dependent than the feature with RCD = 0.8 additive noise. But when sample size is increased to $n = 10000$, the ranking between these two features reverses. While it is not necessary for other dependence measures to rank features across different noise types in the same order as RCD, the stability of the ranking under different sample sizes is desirable. The non-robust-equitable dependence measures may not provide consistent ranking.

In summary, we observe three advantages of RCD for feature selection, in comparison to other dependence measures. (1) Non-equitable measures such as ρ may prefer certain functional relationships (say, linear), while RCD treat them equitably. (2) Self-equitable but non-robust-equitable measure such as CD_2 prefer features with larger sample size. (3) Self-equitable but non-robust-equitable measure such as CD_2 does not provide stable ranking among features when sample size changes.

4.2 Multivariate Equitability Analysis

We now perform equitability analysis on simulated multivariate data. Following the framework of Reshef et al. (2014); Murrell et al. (2016), we generate noisy data from various functional forms, and plot the estimated dependence measure values against the signal level in Figure 4. Our robust-equitability definition extends the natural signal level to higher dimensional case in the mixture noise model. Here we study three dimensional ($d = 3$) cases of six different nonlinear function relationships, generated with different portion of uniform noise from 0 to 0.9 (with signal portion from 0.1 to 1). The sample size $n = 1000$ is used in this experiment. Figure 5 plots the six three dimensional different nonlinear function relationships at signal level 0.8 (0.2 proportion of uniform noise).

According to Reshef et al. (2014), a measure is more equitable if the length of a band, when cutting each plot in Figure 4 with a horizontal line, is smaller. That is, the measure with smaller bandwidth could capture the dependence purely based on the noise level, and is robust to different (linear and nonlinear) relationships. On the other hand, if the band is very wide, it will give the same score for data with a wide range of noise levels, and hence could not identify strong relationships correctly. Figure 4 shows that RCD is the most equitable, since it has the narrowest bandwidth and

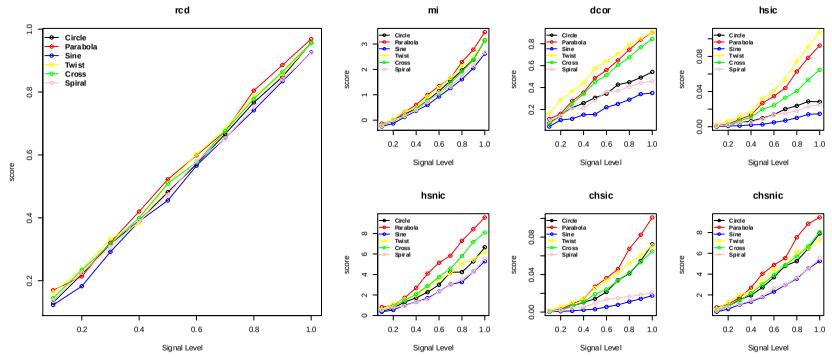


Figure 4: Seven dependence measure values versus signal levels.

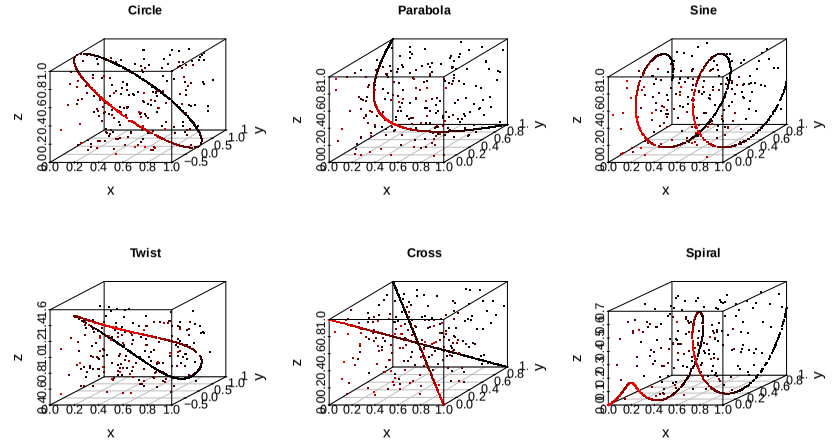


Figure 5: Six function types, circle, parabola, sine/cosine, twisted curve, two cross lines, and a spiral curve, with 20% uniform noise from 0 to 0.9 (with signal portion from 0.1 to 1). The examples in this figure is of signal level 0.8.

lies on the signal level line. The self-equitable mutual information is second best, has relatively narrower bandwidth compared to other measures except RCD.

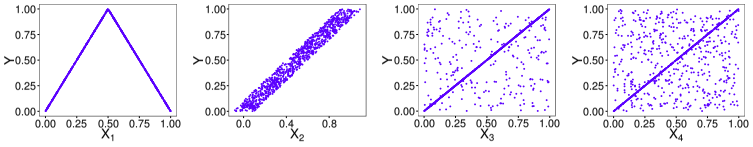


Figure 6: (X_1, Y) has a nonmonotonic relationship with deterministic signal. (X_2, Y) has linear relationship with uniform additive noise with width $d = 0.2$. (X_3, Y) has background noise with a 0.75 linear signal portion. (X_4, Y) is similar to (X_3, Y) but with a 0.5 signal portion.

	X_1		X_2		X_3		X_4	
n	300	10k	300	10k	300	10k	300	10k
ρ_{lin}	0.021	0.00043	0.98	0.98	0.75	0.75	0.51	0.51
HSIC	0.036	0.033	0.095	0.093	0.061	0.057	0.025	0.025
CMMD	0.034	0.034	0.095	0.095	0.060	0.056	0.026	0.025
HSNIC	3.40	3.57	3.63	3.98	3.55	4.08	1.84	1.81
CHSNIC	3.30	3.60	3.59	3.95	3.53	4.07	1.84	1.81
MI	5.06	7.62	2.28	2.37	3.65	5.46	1.82	2.97
CD_2	21.37	102.09	3.75	3.74	24.10	146.73	7.86	44.15
RCD	0.93	0.99	0.77	0.80	0.75	0.76	0.52	0.52
MSE	0.00098	0.0023	0.0033	0.0031	0.037	0.037	0.064	0.063

Table 5: Dependence measure values for synthetic data. Sample sizes $n = 300$ and $n = 10,000$ are considered. Each row corresponds to one type of measure. The MSE is presented in the last row.

4.3 Synthetic Data Sets I: Ranking Features

To compare the performance of each dependence measure in feature selection, we consider four features X_1, X_2, X_3, X_4 and target variable Y as shown in Figure 6. Y has a nonmonotonic but deterministic relationship with X_1 and a linear relationship with X_2 plus some additive noise. In addition, Y has linear relationships with both X_3 and X_4 corrupted by increasing level of continuous background noise. For each feature X_i, we calculate its dependence measure with Y for different sample sizes $n = 300$ and $10,000$. Results are presented in Table 5.

Since X_1 has a deterministic relationship with Y, it should be ranked as more dependent than the other features. X_3 and X_4 has mixture noise with the mixture proportions of 0.75 and 0.5 respectively. We can see that the values learned by RCD are close to those values and correctly ranks X_3 as more dependent than X_4. The last row of Table 5 reports the 10-fold cross-validated mean-squared-error (MSE) of a nonlinear predictor using each feature. Here RCD results are consistent with the MSE results, providing higher scores for those with lower MSE values (more predictive of Y). Now, we inspect the other dependence measures and observe the three issues mentioned in Section 4.1.

Self-equitability. We expect the self-equitable measures to treat linear and nonlinear models equally (i.e., they should prefer X_1 over X_2 because X_1 is purely deterministic while X_2 has some noise). As we can see from Table 5, Pearson correlation coefficient ρ_{lin}, and kernel-based measures prefer

X_2 more than X_1. On the other hand, self-equitable measures MI, CD_2 and RCD were able to rank the features correctly. Although HSNIC and CHSNIC have the same value as CD_2 in the large data limit, empirically they behave similarly to other non-self-equitable kernel-based measures due to slow convergence of their estimators (Reddi and Póczos, 2013).

Selection Correctness in Unequal Sample Sizes. In real applications, some features may have some missing measurements, resulting in unequal number of samples among the various features. In this setting, we still want to compare feature relevance. An ideal dependence measure should not be influenced greatly by unequal sample sizes. Let us take a closer look at MI and CD_2 and on how they rank features X_3 and X_4. Note that X_3 has a stronger signal-to-noise proportion than X_4 ($p = 0.5$ versus $p = 0.75$); thus, ideally, one would like the measure to rank X_3 higher than X_4 as empirically confirmed by the MSE results. The ranking provided by MI and CD_2 is correct when both features have the same sample size, $n = 10,000$. However, if the stronger feature X_3 has missing measurements so that $n = 300$ for X_3, then X_3 is ranked lower than X_4 by CD_2, which would mislead feature selection algorithms. MI will make the same mistake if the sample size for X_4 further increases.

Selection Stability in Different Sample Sizes. Ideally, a measure should not vary too much as the sample size changes. However, we observe that MI, CD_2, and HSNIC's ranking of features X_2, X_3 and X_4 is affected when the sample size is increased. With fixed sample size $n = 300$, MI ranks X_2 as having higher deterministic relationship with Y compared to X_4. However, when the sample size is increased to $n = 10,000$, it reverses the ranking of these features. This is due to its non-robust-equitability and resulting estimation difficulty, as proved in Theorems 4 and 5. Additionally, similar phenomenon appears for CHSNIC and HSNIC on features X_2 and X_3. We observe that when $n = 300$, they rank X_2 as having higher dependence with Y compared to X_3. However, when $n = 10,000$, these rankings are reversed. These inconsistencies may mislead feature selection algorithms.

4.4 Real Data Sets I: Ranking Features

In this subsection, we verify that the equitability properties in Subsection 4.3 are also observed on real data.

Self-equitability. Consider the stock data set from StatLib[1]. This data set provides daily stock prices for ten aerospace companies. Our task is to determine the relative relevance of the stock price of the first two companies (X_1, X_2) in predicting that of the fifth company (Y). The scatter plots of Y against X_1, X_2 are presented in Figure 7. Ideally, self-equitable measures should prefer X_1 over X_2 because the MSE associated with X_1 is lower even though it has a more complex function form. As we can see from Table 8, self-equitable measures MI, CD_2, and RCD all correctly select X_1. While measures that are not self-equitable fail to select the right feature.

Selection Correctness in Unequal Sample Sizes. Consider the KEGG metabolic reaction network data set (Lichman, 2013). Our task is to select the most relevant features in predicting target variable 'Characteristic path length' (Y). The 'Average shortest path' (X_1), 'Eccentricity' (X_2) and 'Closeness centrality' (X_3) are used as candidate features. Observe the ranking of X_1 and X_3 from Table 6, when they have equal sample sizes (either 1000 or 20,000), MI, CD_2 and RCD all rank X_1 as being more relevant than X_3. The MSE values also confirmed that X_1 is more predictive of Y than X_3. However, if there are missing measurements of X_1, then we may need to compare X_1

1. http://lib.stat.cmu.edu/

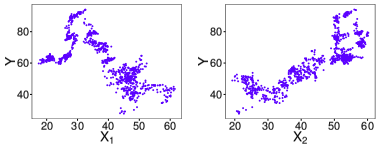


Figure 7: $(X_i, Y), i = 1, 2$ in stock data set

Measures	$X1$	$X2$
ρ_{lin}	-0.68	0.83
HSIC	0.053	0.068
CMMD	0.062	0.073
HSNIC	1.95	2.16
CHSNIC	1.90	1.99
MI	2.06	1.92
CD_2	3.88	3.13
RCD	0.68	0.67
MSE	0.18	0.23

Figure 8: Measures for X_1, X_2 in stock data set

	X_1		X_2		X_3	
n	1k	20k	1k	20k	1k	20k
MI	3.39	3.95	3.23	3.66	2.94	3.54
CD_2	12.05	31.65	10.67	22.44	9.77	28.30
RCD	0.85	0.86	0.82	0.84	0.77	0.80
MSE	0.030	0.028	0.032	0.032	0.14	0.14

Table 6: Dependence measure for three features in metabolic reaction network data set

with 1000 samples and X_3 with 20,000 samples. The feature X_3 with less signal strength but larger sample size is given higher ranking by MI and CD_2, degrading the performance of feature selection algorithms. In contrast, RCD correctly identifies X_1 as being more relevant than X_3 even with the unequal sample size.

Selection Stability in Different Sample Sizes. Ideally, a measure should not vary too much as the sample size changes. However, in Table 6, CD_2's ranking of features X_2 and X_3 is affected by the increase in sample size. If we fix sample size $n = 1000$, CD_2 ranks X_2 as more relevant than X_3 in predicting Y, agreeing with the MSE ranking. However, when the sample size increases to $n = 20,000$, CD_2 will prefer X_3. CD_2 will select the feature X_3 when the sample size is large even though it is less relevant to Y. RCD has the same ranking under both sample sizes.

4.5 Synthetic Data Sets II: mRMR Feature Selection

In this part, we investigate the performance of feature selection for each dependence measure with mRMR (Peng et al., 2005). We generate data from the following additive regression model $Y = 1.5\cos(3\pi X_1) + (1 - 2|2X_2 - 1|)^2 + \epsilon$, where X_1 and X_2 are uniformly distributed on $[0, 1]$, and $\epsilon \sim N(0, 0.05)$. We consider feature selection from twenty features. The first two features are X_1 and X_2. The next six features are noisy versions of X_1 and X_2, with some as mixtures. They are also related to Y. A good feature selection method should select X_1 and X_2 before these more

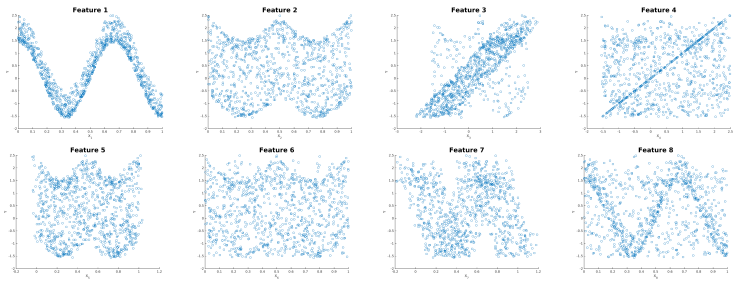


Figure 9: Scatter plots of the features $X_1, \cdots, X_8$ versus Y.

Feature X_1	X_1
Feature X_2	X_2
Feature X_3	25% X_2 and 75% Y with additive noise $U(-0.75, 0.75)$
Feature X_4	20% X_2, 20% Y and 60% background noise
Feature X_5	X_2 with additive noise $U(-0.05, 0.05)$
Feature X_6	50% X_2 and 50% background noise
Feature X_7	X_1 with additive noise $U(-0.2, 0.2)$
Feature X_8	50% X_1 and 50% background noise
Feature X_9,...,X_{20}	pure random noise

Table 7: Features that are used in the additive regression model.

noisy features X_3 to X_8. The rest 12 features X_9 to X_{20} are simply random noise not related to X_1, X_2 or Y. The features are listed in Table 7. And we plot the first eight features versus Y in one simulation run in Figure 9.

We generate data sets with size $n = 1000$, and select the features with mRMR based on the eight dependence measures. We repeat this experiment fifty times and record the order of each feature being selected in each data sets. We then apply spline regression model using the top one to ten selected features and report their respective cross-validated mean square error (MSE). The cross-validated MSE averaged over 50 runs are plotted in Figure 10.

In this synthetic example, RCD yields the lowest MSE. We can see why by looking at the feature selection result in more details. Table 8 reports the features that are most frequently selected as the top one to five features using mRMR. Inversely, Table 9 shows the median order of being selected for each relevant feature X_1, ..., X_8.

From the tables, RCD correctly selects the two true features X_1 and X_2 as the top two features, thus resulting in the lowest MSE curve in Figure 10. The non-self-equitable measures, for this data distribution, incorrectly ranks first the feature X_3 which has linear relationship with Y in parts of the data (75% mixture). Then Pearson's correlation ρ does not rank X_1 and X_2 high because the relationships are nonlinear. Some nonlinear non-self-equitable measures (HSNIC, CMMD, CHSNIC) are able to rank the feature X_1 second, but cannot select feature X_2 which has a nonlinear relationship with Y. That is due to X_2, which also has some dependence with X_3, was penalized when X_3 was selected first. The self-equitable measures MI, CD_2 and RCD, in contrast, correctly rank X_1 first. However, only the robust-equitable RCD ranks X_2 second. The non-robust-equitable MI and CD_2 incorrectly ranks the noisy X_4 second, instead of X_2, due to mishandling of the mixture noise.

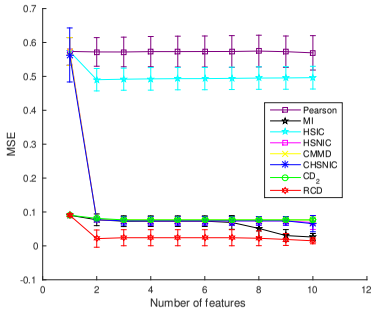


Figure 10: Cross-validated MSE plot with error bar based on 50 repeated runs.

Order of selection	1	2	3	4	5
ρ	X_3(100%)	*	X_4(50%)	*	*
HSIC	X_3(100%)	X_7(96%)	*	*	*
HSNIC	X_3(100%)	X_1(100%)	X_4(78%)	*	*
CMMD	X_3(100%)	X_1(100%)	*	X_4(52%)	*
CHSNIC	X_3(100%)	X_1(100%)	X_4(98%)	*	*
MI	X_1(100%)	X_4(78%)	X_3(90%)	*	*
CD_2	X_1(100%)	X_4(80%)	X_3(80%)	*	*
RCD	X_1(100%)	X_2(82%)	X_3(90%)	*	*

Table 8: The most frequently selected top five features, with the relative frequency in parenthesis. Asterisk indicates one of the random noise features (X_9, ..., X_{20}).

Feature	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
ρ	20	19	1	3.5	18	15	18	16
HSIC	18	20	1	4	19	16	2	17
HSNIC	2	17.5	1	3	20	19	18	10
CMMD	2	20	1	4	18	16	19	17
CHSNIC	2	15.5	1	3	20	19	17	10
MI	1	20	3	2	9	19	13	12
CD_2	1	20	3	2	17	18	16	19
RCD	1	2	3	7	20	19	13	6

Table 9: Median of the order of selection for the first eight features in each dependence measure experiment among fifty repeated runs.

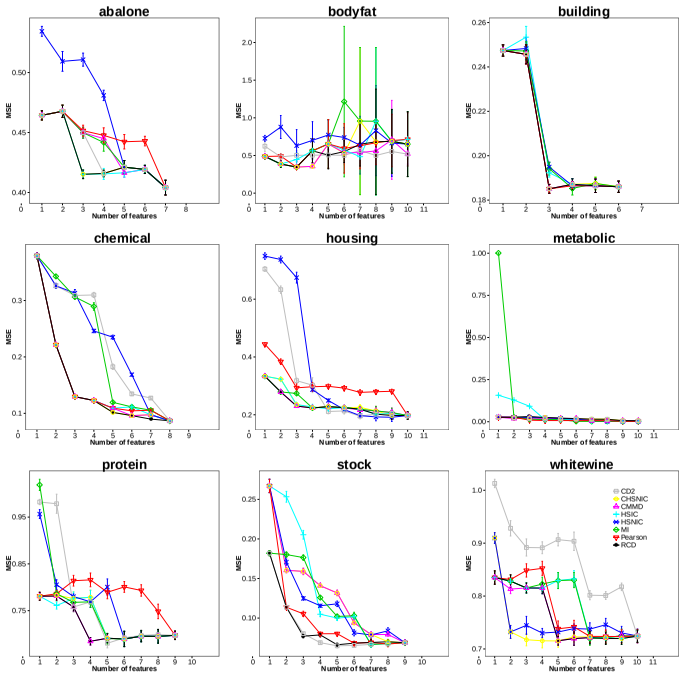


Figure 11: MSE of mRMR-based Feature Selection on nine real-world data sets

4.6 Real Data Sets II: mRMR Feature Selection

Here we used the various dependence measures as measures of relevance and redundancy in the mRMR-based search strategy, and compare the feature selection results on nine real-world data sets. Due to the cubic computational cost of kernel-based measures (HSNIC, CHSNIC), up to 1000 samples are used for each data set. We compare the results of RCD versus other dependence measures by showing plots of 10-fold cross-validated MSE using spline regressor with the features selected by these measures versus the number of selected features in Figure 11.

We used the Kruskal-Wallis test to compare the MSE between different measures. Table 10 lists the top dependence measures in order of their MSE. For each data set, we only include the dependence measures resulting in MSE equivalent to the best MSE (p-value > 0.05 for Kruskal-Wallis test) in the table. We can see that RCD performs as one of the best in 8 out of 9 data sets. Most other measures are worse off in more than half of the data sets. Only CHSNIC and CMMD are close in performance. In particular, CMMD is among the best measures in 5 data sets. CHSNIC performs as one of the best in 6 data sets and actually beats RCD in one data set, *whitewine*. RCD in fact find the best top feature in every data set including *whitewine*. However, the best second feature

Data set	Top Dependence Measures
abalone	**RCD** CHSNIC HSIC
bodyfat	**RCD** CMMD CHSNIC
building	**RCD** CHSNIC Pearson CMMD CD_2
chemical	**RCD** CHSNIC
housing	**RCD** CMMD
metabolic	**RCD** CHSNIC CD_2 CMMD HSNIC Pearson
protein	**RCD** CMMD
stock	**RCD** CD_2
whitewine	CHSNIC

Table 10: Measures ranked by predictive MSE

in *whitewine* was not selected by RCD. Overall, RCD has the best performance in mRMR-based feature selection compared to competing dependence measures.

5. Conclusions and Discussions

As the data size explodes, researchers are studying increasingly complex relationships among features. Restricting the focus on simple linear relationship can miss very informative features. Therefore, how to measure the dependence strength equitably for various functional relationships has attracted recent interest from researchers (Reshef et al., 2011; Kinney and Atwal, 2014; Murrell et al., 2014; Reshef et al., 2015b). This paper provides a theoretical treatment of various equitability definitions, including our proposal of the robust-equitability concept. The robust copula dependence (RCD) is proven to be both self-equitable and robust-equitable. Theoretically we show that RCD is intrinsically easier to estimate than some other self-equitable dependence measures (such as MI and CD_2). Particularly, through minimax rate of convergence, we provide a theoretical explanation for the difficulty of accurately estimating MI which is noted by practitioners. A practical estimator is provided for RCD, which enables its usage in feature selection.

Through theoretical and empirical studies, we have shown that RCD does better in ranking the features according to deterministic signal strengths compared to other dependence measures. The non-self-equitable measures may prefer noisy features with certain types of relationships (e.g., monotonic) over less noisy features with more complex relationships. Self-equitable but non-robust-equitable measures (such as MI and CD_2) overcome this deficiency but have estimation problems, leading to non-robust feature selection particularly when comparing features with unequal sample sizes. RCD can be used in feature selection to overcome these limitations.

Using nonlinear dependence measures, rather than the Pearson's correlation, in high-dimensional data analysis (e.g. independent component analysis) is becoming more popular to deal with possibly non-Gaussian noises. The equitability properties of RCD make it an ideal choice of dependence measure in such applications. Replacing measures such as MI by RCD may lead to more robust results as shown in the examples of Section 4. RCD estimation in high-dimensional case however, similar to MI, may be inaccurate as it involves high-dimensional density estimation. The improvement on nonparametric RCD estimation remains an ongoing research effort, and can lead to wider applications.

Acknowledgments

We would like to acknowledge support for this project from NSF grant CCF-1442728.

Appendix A. Proof of Theorem 4

For simplicity, we focus on the bivariate case (X and Y are each one-dimensional variables). The extension of the proof to the multivariate case is straight forward. We first work on mutual information, then show the similar arguments on the copula distances. To prove the theorem, we use Le Cam (1973)'s method to find the lower bound on the minimax risk of the estimating mutual information MI. To do this, we will use a more convenient form of Le Cam's method developed by Donoho and Liu (1991). Define the module of continuity of a functional T over the class $\mathbf{F}$ with respect to Hellinger distance as in equation (1.1) of Donoho and Liu (1991):

$$w(\varepsilon) = \sup\{|T(F_1) - T(F_2)| : H(F_1, F_2) \le \varepsilon, F_i \in \mathbf{F}\}. \tag{13}$$

Here $H(F_1, F_2)$ denotes the Hellinger distance between F_1 and F_2. Then the minimax rate of convergence for estimating $T(F)$ over the class $\mathbf{F}$ is bounded below by $w(n^{-1/2})$.

We now look for a pair of density functions $c_1(u, v)$ and $c_2(u, v)$ on the unit square for distributions that are close in Hellinger distance but far away in their mutual information. This provides a lower bound on the module of continuity for mutual information MI over the class $\mathfrak{C}$, and hence leads to a lower bound on the minimax risk. We outline the proof next.

We first divide the unit square into three disjoint regions R_1, R_2 and R_3 with $R_1 \cup R_2 \cup R_3 = [0,1]\times[0,1]$. The first density function $c_1(u, v)$ puts probability masses δ, a and $1-a-\delta$ respectively on the regions R_1, R_2 and R_3 each uniformly. The a is an arbitrary small fixed value, for example, $a = 0.01$. For now, we take δ to be another small fixed value. The area of the region is chosen so that $c_1(u, v) = M$ on region R_2 and $c_1(u, v) = M^*$ on region R_1 for a very big M^*. The second density function $c_2(u, v)$, compared to $c_1(u, v)$, moves a small probability mass ε from R_1 to R_2. We will see that the Hellinger distance between c_1 and c_2 is of the same order as ε, but the change in MI is unbounded for big M^*. Hence module of continuity $w(\varepsilon)$ is unbounded for mutual information MI. Therefore the MI can not be consistently estimated over the class $\mathfrak{C}$.

Specifically, the region R_1 is chosen to be a narrow strip immediately above the diagonal, $R_1 = \{(u, v) : -\delta_1 < u - v < 0\}$; and R_2 is chosen to be a narrow strip immediately below the diagonal, $R_2 = \{(u, v) : 0 \le u - v < \delta_2\}$. The remaining region is $R_3 = [0,1]\times[0,1]\setminus(R_1 \cup R_2)$. The values of δ_1 and δ_2 are chosen so that the areas of regions R_1 and R_2 are δ/M^* and a/M respectively. Then clearly $c_1(u, v) = M^*$ on R_1; $c_1(u, v) = M$ on R_2; $c_1(u, v) = (1-a-\delta)/(1-a/M-\delta/M^*)$ on R_3. And $c_2(u, v) = M^* - \varepsilon(M^*/\delta)$ on R_1; $c_2(u, v) = M + \varepsilon(M/a)$ on R_2; $c_2(u, v) = c_1(u, v)$ on R_3. See the Figure 12.

Then we have

$$\begin{aligned} 2H^2(c_1, c_2) &= \int (\sqrt{c_2(u,v)} - \sqrt{c_1(u,v)})^2 du dv \\ &= (\sqrt{M^* - \varepsilon(M^*/\delta)} - \sqrt{M^*})^2 \delta/M^* + (\sqrt{M + \varepsilon(M/a)} - \sqrt{M})^2 a/M \\ &= \delta(\sqrt{1 - \varepsilon/\delta} - 1)^2 + a(\sqrt{1 + \varepsilon/a} - 1)^2 \\ &= \delta(\varepsilon/2\delta)^2 + a(\varepsilon/2a)^2 + o(\varepsilon^2) \\ &= \varepsilon^2(\tfrac{1}{4\delta} + \tfrac{1}{4a}) + o(\varepsilon^2). \end{aligned}$$

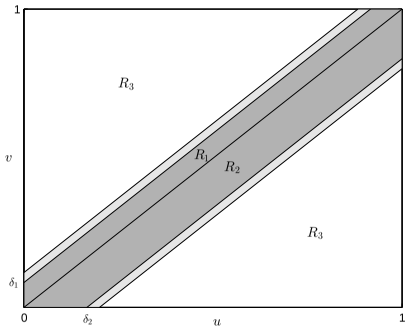


Figure 12: The plot shows the regions R_1, R_2 and R_3. The other two narrow strips neighboring R_1 and R_2 are for the continuity correction mentioned at the end of the proof.

Hence the Hellinger distance is of the same order as ε:

$$H(c_1, c_2) = \varepsilon\sqrt{\frac{1}{8\delta} + \frac{1}{8a}} + o(\varepsilon).$$

On the other hand, the difference in the mutual information is

$$\begin{aligned} &\mathrm{MI}(c_1) - \mathrm{MI}(c_2) \\ =\ & \delta\log(M^*) + a\log(M) - (\delta - \varepsilon)\log[M^* - \varepsilon(M^*/\delta)] - (a + \varepsilon)\log[M + \varepsilon(M/a)] \\ =\ & \varepsilon\log(M^*) - \varepsilon\log(M) - (\delta - \varepsilon)\log(1 - \varepsilon/\delta) - (a + \varepsilon)\log(1 + \varepsilon/a). \end{aligned} \tag{14}$$

Here M, δ and a are fixed constants. Hence when $M^* \to \infty$, this difference in MI also goes to ∞. For example, if we let $M^* = e^{1/(\varepsilon)^2}$, then the module of continuity $w(\varepsilon) \ge O(1/\varepsilon)$. That means, the rate of convergence is at least $O(w(n^{-1/2})) = O(n^{1/2}) \to \infty$. In other words, MI can not be consistently estimated.

Now, let us consider the $\mathrm{CD}_\alpha = \int_{I^2} |c(u, v) - 1|^\alpha du dv$, for $\alpha > 1$, where I^2 is the unit square.

$$\begin{aligned} &\mathrm{CD}_\alpha(c_1) - \mathrm{CD}_\alpha(c_2) \\ =\ & |M^* - 1|^\alpha \delta/M^* + |M^- 1|^\alpha a/M - |M^* - 1 - \varepsilon(M^*/\delta)|^\alpha \delta/M^* + |M - 1 + \varepsilon(M/a)|^\alpha a/M \\ =\ & [|M^* - 1|^\alpha - |M^* - 1 - \varepsilon(M^*/\delta)|^\alpha]\delta/M^* + [|M^- 1|^\alpha - |M - 1 + \varepsilon(M/a)|^\alpha]a/M \\ =\ & \alpha[(M^* - 1)^{\alpha-1} M^*/\delta - (M - 1)^{\alpha-1} M/\alpha]\varepsilon + o(\varepsilon^2). \end{aligned} \tag{15}$$

Again, M, δ and a are fixed constants. Hence when $M^* \to \infty$, this difference in $\mathrm{CD}_\alpha, \alpha > 1$ also goes to ∞. For example, if we let $M^* = (\varepsilon^{-2} + M^\alpha)^{\frac{1}{\alpha-1}} + 1$, then the module of continuity $w(\varepsilon) \ge O(1/\varepsilon)$. Note that $\alpha > 1$ is essential here. That means, the rate of convergence is at least $O(w(n^{-1/2})) = O(n^{1/2}) \to \infty$. In other words, $\mathrm{CD}_\alpha, \alpha > 1$ can not be consistently estimated.

The above outlines the main idea of the proof, ignoring some mathematical subtleties. One is that the example densities c_1 and c_2 are only piecewise continuous on the three regions, but not truly continuous as required for the class $\mathfrak{C}$. This can be easily remedied by connecting the three pieces linearly. Specifically we set the densities $c_i(u,v) = M$, $i = 1, 2$, on the boundary between R_1 and R_3, $\{(u,v) : u - v = -\delta_1\}$, and on the boundary between R_2 and R_3, $\{(u,v) : u - v = \delta_2\}$. Then we use two narrow strips within R_3, $\{(u,v) : -\delta_3 \le u-v \le -\delta_1\}$ and $\{(u,v) : \delta_2 \le u-v \le \delta_4\}$ to connect the constant $c_i(u,v)$ values on the rest of region R_3 with the boundary value $c_i(u,v) = M$ continuously through linear (in $u - v$) $c_i(u,v)$'s on the two strips that satisfies the Hölder condition (7) of the main text. By the Hölder condition, the connection can be made with strips of width at most $(M - 1 + a + \delta)/M_1$. This continuity modification does not affect the calculation of the difference $\text{MI}(c_1) - \text{MI}(c_2)$ or $\text{CD}_\alpha(c_1) - \text{CD}_\alpha(c_2)$ above as c_1 and c_2 only differ on regions R_1 and R_2. Within regions R_1 and R_2, the densities c_1 and c_2 can be further similarly connected continuously linearly in $u - v$. As there is no Hölder condition on A_M^c, the connection within R_1 and R_2 can be as steep as we want. Clearly the order obtained through above calculations will not change if we make these connections very steep so that their effect is negligible.

Another technical subtlety is that the c_1 and c_2 defined above are only densities on the unit square but not copula densities which require uniform marginal distributions. However, it is clear that the marginal densities for c_is are uniform over the interval $(\delta_3, 1 - \delta_4)$ and linear in the rest of interval near the two end points 0 and 1. The copulas densities c_i^*'s corresponding to c_i's can be calculated directly through Sklar's decomposition (1) in the main text. It is easy to see that the order for the module of continuity $w(\varepsilon)$ remains the same for using the corresponding copula densities c_i^*'s.

Appendix B. Proof of Theorem 5

The proof is almost the same as the proof for MI, but need some modification of the pair of least favorable c_1 and c_2 above. The small difference in Hellinger distance of c_1 and c_2 can lead to unbounded difference in $MI(c_1)$ and $MI(c_2)$ since MI is unbounded. After the transformation $MIcor = \sqrt{1 - e^{-2MI}}$ is bounded. The difference between $MIcor(c_1)$ and $MIcor(c_2)$ in the above example is actually small since the MI are big for both c_1 and c_2 (leading to corresponding $MIcor$s close to zero). However, $MIcor$ is also very hard to estimate over the class $\mathfrak{C}$. To see this, we follow the same reasoning above but modify the example of c_1 and c_2. First, we notice that for any pair of densities c_1 and c_2,

$$\begin{aligned} |MIcor(c_1) - MIcor(c_2)| &= |\sqrt{1 - e^{-2MI(c_1)}} - \sqrt{1 - e^{-2MI(c_2)}}| \\ &= |\tfrac{[1-e^{-2MI(c_1)}]-[1-e^{-2MI(c_2)}]}{\sqrt{1-e^{-2MI(c_1)}}+\sqrt{1-e^{-2MI(c_2)}}}| \\ &\ge \tfrac{1}{2}|e^{-2MI(c_1)} - e^{-2MI(c_2)}| \\ &= \tfrac{1}{2}e^{-2MI(c_1)}|1 - e^{-2[MI(c_1)-MI(c_2)]}|. \end{aligned}$$

For the difference $MIcor(c_1) - MIcor(c_2)$ to be the same order of the difference $MI(c_1) - MI(c_2)$, we need to set $MI(c_1)$ at constant order when $\varepsilon \to 0$.

Therefore, we modify the above c_1 to have probability mass $\delta = 2\varepsilon$ in region R_1, varying with the ε value instead of fixed as before. And we set $M^* = e^{1/\varepsilon}$, leading to

$$\begin{aligned} & MI(c_1) \\ = \;& \delta \log(M^*) + a\log(M) + (1 - a - \delta)\log[(1 - a - \delta)/(1 - a/M - \delta/M^*)] \\ = \;& 2 + a\log(M) + (1 - a - 2\varepsilon)\log[(1 - a - 2\varepsilon)/(1 - a/M - 2\varepsilon e^{-1/\varepsilon})], \end{aligned}$$

which converges to a fixed constant $a_1 = 2 + a\log(M) + (1-a)\log[(1-a)/(1-a/M)]$ as $\varepsilon \to 0$. Using (14), recall that $\delta = 2\varepsilon$ and $M^* = e^{1/\varepsilon}$, we have

$$\begin{aligned} & MI(c_1) - MI(c_2) \\ = \;& \varepsilon\log(M^*) - \varepsilon\log(M) - (\delta - \varepsilon)\log(1 - \varepsilon/\delta) - (a + \varepsilon)\log(1 + \varepsilon/a) \\ = \;& 1 - \varepsilon\log(M) - \varepsilon\log(1/2) - (a + \varepsilon)\log(1 + \varepsilon/a), \end{aligned}$$

which converges to 1 as $\varepsilon \to 0$. Hence we have

$$\lim_{\varepsilon\to 0} w(\varepsilon) \ge \lim_{\varepsilon\to 0} \frac{1}{2}e^{-2MI(c_1)}|1 - e^{-2[MI(c_1)-MI(c_2)]}| = \frac{1}{2}e^{-2a_1}(1 - e^{-2(1)}),$$

a positive constant $a_2 = e^{-2a_1}(1 - e^{-2})/2$. Therefore, $MIcor$ can not be estimated consistently over the class $\mathfrak{C}$ either.

Appendix C. Proof of Theorem 6

The first two terms in (11) corresponds to bias and standard deviation of kernel density estimation when the copula density is bounded. When the copula density is unbounded, the kernel density estimation $\hat{c}(\mathbf{Z})$ is not consistent. However, a smaller order $O(\frac{1}{nh^d})$ term bounds the overall error contribution to $\widehat{RCD}$ resulting from $\hat{c}(\mathbf{Z})$ in the unbounded copula density region.

Let $M_2 = \frac{M+1}{2}$, $A_{M_2} = \{\mathbf{Z}|c(\mathbf{Z}) \le M_2\}$, $T_1(c) = \int_{A_{M_2}}(1 - c(\mathbf{Z}))_+ d\mathbf{Z}$, $T_2(c) = \int_{A_{M_2}^c}(1 - c(\mathbf{Z}))_+ d\mathbf{Z}$, $RCD = T_1(c) + T_2(c)$, and $\widehat{RCD} = T_1(\hat{c}) + T_2(\hat{c})$

Firstly, we consider the region A_{M_2} with bounded copula density. Here we calculate the bias and variance of the kernel density estimator using standard methods first.

$$\bar{c}_n(\mathbf{Z}) = E[\hat{c}_{kde}(\mathbf{Z})] = \frac{1}{h^d}\int K(\frac{\mathbf{z} - \mathbf{Z}}{h})c(\mathbf{z})d\mathbf{z} = \int K(s)c(\mathbf{Z} + sh)ds.$$

Hence

$$\begin{aligned} |Bias(\mathbf{Z})| = |\textstyle\int K(s)c(\mathbf{Z} + sh)ds - c(\mathbf{Z})| &\le \textstyle\int_{\mathbb{B}_0} K(s)|c(\mathbf{Z} + sh) - c(\mathbf{Z})|ds \\ &\le \textstyle\int_{\mathbb{B}_0} K(s)M_1 h ds \\ &= M_1 h. \end{aligned} \tag{16}$$

$$\begin{aligned} |Var(\mathbf{Z})| &= \tfrac{1}{n}Var[\tfrac{1}{h^d}K(\tfrac{\mathbf{Z}_1 - \mathbf{Z}}{h})] \le \tfrac{1}{n}E[\tfrac{1}{h^{2d}}K^2(\tfrac{\mathbf{Z}_1 - \mathbf{Z}}{h})] \\ &= \tfrac{1}{nh^d}\textstyle\int_{\mathbb{B}_0} K^2(s)c(\mathbf{Z} + sh)ds \\ &\le \tfrac{1}{nh^d}\textstyle\int_{\mathbb{B}_0} K^2(s)[c(\mathbf{Z}) + M_1 h]ds \\ &= \tfrac{\mu_2^2}{nh^d}[c(\mathbf{Z}) + M_1 h]. \end{aligned} \tag{17}$$

Hence the integrated mean square error of the density estimator $\hat{c}_n(\mathbf{Z})$ over regions A_{M_2} is

$$\begin{aligned} IMSE(\mathbf{Z}) &= \int_{A_{M_2}} [Bias^2(\mathbf{Z}) + Var(\mathbf{Z})] d\mathbf{Z} \\ &\le \int_{A_{M_2}} [M_1^2 h^2 + \tfrac{\mu_2^2}{nh^d}[c(\mathbf{Z}) + M_1 h] d\mathbf{Z} \ \le M_1^2 h^2 + \tfrac{\mu_2^2}{nh^d}[1 + M_1 h] \\ &\le M_1^2 h^2 + \tfrac{2\mu_2^2}{nh^d} \end{aligned} \tag{18}$$

Hence the error of $\widehat{RCD}$ on A_{M_2} is bounded by

$$\begin{aligned} E|T_1(\hat{c}) - T_1(c)| &\le E \int_{A_{M_2}} |(1 - \hat{c}_n(\mathbf{Z}))_+ - (1 - c(\mathbf{Z}))_+| d\mathbf{Z} \\ &\le E \int_{A_{M_2}} |\hat{c}_n(\mathbf{Z}) - c(\mathbf{Z})| d\mathbf{Z} \\ &\le \sqrt{E \int_{A_{M_2}} (\hat{c}_n(\mathbf{Z}) - c(\mathbf{Z}))^2 d\mathbf{Z}} \\ &\le \sqrt{d^2 M_1^2 h^2 + \tfrac{2\mu_2^2}{nh^d}} \\ &\le dM_1 h + \sqrt{2}\mu_2 (\tfrac{1}{nh^d})^{1/2}. \end{aligned}$$

Now we consider the region $A^c_{M_2}$ with unbounded copula density. For $\mathbf{Z} \in A^c_{M_2}$, $\hat{c}(\mathbf{Z})$ does not have a finite variance bound in (17). But we can bound the variance by the expectation $\bar{c}_n(\mathbf{Z}) = E[\hat{c}_n(\mathbf{Z})]$. Let $M_3 = \frac{M_2+1}{2}$, when h small, $\mathbf{Z} \in A^c_{M_2}$ implies $\mathbf{Z} + sh \in A^c_{M_3}$. Hence

$$|Var(\mathbf{Z})| \le \frac{1}{nh^d} \int_{\mathbb{B}_0} K^2(s) c(\mathbf{Z} + sh) ds \le \frac{M_K}{nh^d} \int_{\mathbb{B}_0} K(s) c(\mathbf{Z} + sh) ds = \frac{M_K}{nh^d} \bar{c}_n(\mathbf{Z})$$

Using Chebyshev's inequality,

$$\begin{aligned} E[1_{\{\hat{c}_n(\mathbf{Z}) < 1\}}] = P(\hat{c}_n(\mathbf{Z}) < 1) &\le P(|\bar{c}_n(\mathbf{Z}) - \hat{c}_n(\mathbf{Z})| > \bar{c}_n(\mathbf{Z}) - 1) \\ &\le \frac{Var[\hat{c}_n(\mathbf{Z})]}{[\bar{c}_n(\mathbf{Z}) - 1]^2} \\ &\le \frac{M_K}{nh^d} \frac{\bar{c}_n(\mathbf{Z})}{[\bar{c}_n(\mathbf{Z}) - 1]^2} \ \le \frac{M_K M_4}{nh^d} \end{aligned}$$

where $M_4 = \frac{M_3}{(M_3 - 1)^2}$.

Hence the error of $\widehat{RCD}$ on $A^c_{M_2}$ is bounded by

$$E|T_2(\hat{c}) - T_2(c)| = E[T_2(\hat{c})] \le \int_{A^c_{M_2}} E[1_{\{\hat{c}_n(\mathbf{Z}) < 1\}}] d\mathbf{Z} \le \frac{M_K M_4}{nh^d}$$

Combining the above results:

$$E[|\widehat{RCD} - RCD|] \le M_1 h + \frac{\sqrt{2}\mu_2}{\sqrt{n} h^{d/2}} + \frac{M_K M_4}{nh^d}. \tag{19}$$

This finishes the proof.

Note that we can use any L_p norm ($1 \le p \le \infty$) in the Hölder condition: equation (7). The kernel K is then assumed to have support in the unit ball $\mathbb{B}_0$ corresponding to that L_p norm. The proof remains exactly the same. We in fact will use L_∞ norm in our estimator for computational simplicity. In that case, the unit ball $\mathbb{B}_0 = \{\mathbf{Z} : \|\mathbf{Z}\|_{l_\infty} \le 1\}$ is in fact the d-dimensional cube.

Appendix D. Proof of Theorem 7

Here for $\widehat{RCD} = RCD(\hat{c})$ we use the k-NN estimator (Loftsgaarden and Quesenberry, 1965) of the copula density

$$\hat{c}_{knn}(\mathbf{Z}) = \frac{\frac{k(n)}{n}}{A_{r(k(n),n),\mathbf{Z}}}, \tag{20}$$

where $\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_n$ are the copula based observations, $r(k(n), n)$ is the distance from $\mathbf{Z}$ to the k^{th} closest of $\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_n$ and $A_{r(k(n),n),\mathbf{Z}}$ is the volume of the d-dimensional hyper-ball with radius $r(k(n), n)$.

In the following, without ambiguity, we denote $r(k(n), n)$ by r, and $k(n)$ by k. Hence the volume $A_{r,\mathbf{Z}}$ is $v_d \cdot r^d$, where v_d is the volume of the d-dimensional unit ball $\mathbb{B}_0$. And $\hat{c}_{knn}(\mathbf{Z}) = k/(v_d n r^d)$. For l_2 norm, $v_d = \pi^{d/2}/\Gamma[(d+2)/2]$ where $\Gamma(\cdot)$ denotes the Gamma function.

Moore and Yackel (1977a) showed that, for bounded densities, there is equivalence between the consistency of the KDE density estimator and the consistency of the k-NN estimator. To cite the results of (Moore and Yackel, 1977a), we assume a slightly stronger version of the Hölder condition than (7). That is, we assume that c also has bounded continuous *second order* derivative in A_M. Let $Q(\mathbf{Z}) = tr[\frac{\partial^2 c(\mathbf{Z})}{\partial \mathbf{Z}^2}]$ denote the trace of the Hessian matrix of copula density $c(\mathbf{Z})$. For the d-dimensional vector $\mathbf{Z} = (z_1, ..., z_d)$, the Hessian matrix $\partial^2 c(\mathbf{Z})/\partial \mathbf{Z}^2$ has entries

$$[\frac{\partial^2 c(\mathbf{Z})}{\partial \mathbf{Z}^2}]_{ij} = \frac{\partial^2 c(\mathbf{Z})}{\partial z_i \partial z_j}.$$

Then we rewrite the error bound in Theorem 7 explicitly as

$$\sup_{C \in \mathfrak{C}} E[|\widehat{RCD} - RCD|] \le 2\bar{Q}(\frac{k}{n\epsilon})^{\frac{2}{d}} + \frac{2M}{\sqrt{k}} + 2\epsilon,$$

where $\bar{Q} = \frac{1}{2(d+2)\pi} \Gamma^{2/d}(\frac{d+2}{2}) \sup_{\mathbf{Z} \in A_M} Q(\mathbf{Z})$, and $\epsilon = \epsilon(n)$ is any sequence converging to 0 slower than k/n. We suppress the n from the notation in ϵ without ambiguity as in k and r above.

We shall use the following asymptotic results on k-NN density estimator in Mack and Rosenblatt (1979). Denote $\tilde{Q}(\mathbf{Z}) = \frac{1}{2(d+2)\pi} \Gamma^{2/d}(\frac{d+2}{2}) Q(\mathbf{Z})$. Then

$$\begin{aligned} Bias[\hat{c}_{knn}(\mathbf{Z})] &= \tfrac{\tilde{Q}(\mathbf{Z})}{c(\mathbf{Z})^{2/d}} (\tfrac{k}{n})^{2/d} + O(\tfrac{c(\mathbf{Z})}{k}) + o((\tfrac{k}{n})^{2/d}), \\ Var[\hat{c}_{knn}(\mathbf{Z})] &= \tfrac{c^2(\mathbf{Z})}{k} + o(\tfrac{1}{k}). \end{aligned} \tag{21}$$

These expressions provide control on the error contribution of $\hat{c}(\mathbf{Z})$ to $\widehat{RCD}$ when $c(\mathbf{Z})$ is bounded both from above and from below. Similar to the proof of KDE-based $\widehat{RCD}$, we prove that the error contribution to $\widehat{RCD}$ from the big copula density region is of a smaller order $O(1/k)$. Different from the KDE, the k-NN density estimator also does not have finite bias bound in (21) when the copula density $c(\mathbf{Z})$ is not bounded below. Therefore, we also need to control the error contribution to $\widehat{RCD}$ from the small ($< \epsilon$) copula density region separately.

As before, let M_2 be a constant between 1 and M, say, $M_2 = \frac{M+1}{2}$. We now separate the three regions by copula density: $A^c_{M_2} = \{\mathbf{Z} : c(\mathbf{Z}) > M_2\}$ (big), $A_{M_2,\epsilon} = \{\mathbf{Z} : \epsilon \le c(\mathbf{Z}) \le M_2\}$ (middle) and $A_\epsilon = \{\mathbf{Z} : c(\mathbf{Z}) < \epsilon\}$ (small). Then we can separate RCD into three components

$RCD = T_1(c) + T_2(c) + T_3(c)$: $T_1(c) = \int_{A_{M_2^c}} [1 - c(\mathbf{Z})]_+ d\mathbf{Z}$, $T_2(c) = \int_{A_{M_2,\epsilon}} [1 - c(\mathbf{Z})]_+ d\mathbf{Z}$ and $T_3(c) = \int_{A_\epsilon} [1 - c(\mathbf{Z})]_+ d\mathbf{Z}$.

Firstly, we look at the error bound on $A^c_{M_2}$, the region of big copula density. Similar to the KDE, the error in $\hat{c}_{knn}(\mathbf{Z})$ can be arbitrarily large for $\mathbf{Z} \in A^c_{M_2}$. However, the error only leads to the error in $\widehat{RCD}$ if $\hat{c}_{knn}(\mathbf{Z}) < 1$. From equation (20), $\hat{c}_{knn}(\mathbf{Z}) < 1$ if and only if

$$r > (\frac{k}{nv_d})^{1/d} \stackrel{\text{def}}{=} \bar{r}.$$

This occurs when at most $k-1$ of observations $\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_n$ fall into the ball $\mathbb{B}(\mathbf{Z}; \bar{r})$ which is centered at $\mathbf{Z}$ with radius $\bar{r}$.

Let $\bar{N}(\mathbf{Z})$ denotes the number of observations falling into $\mathbb{B}(\mathbf{Z}; \bar{r})$. Then $\bar{N}(\mathbf{Z})$ follows a binomial distribution with mean $n\bar{p}$, where $\bar{p} = \int_{\mathbb{B}(\mathbf{Z};\bar{r})} c(\mathbf{z}) d\mathbf{z}$. Since $k/n \to 0$, $\bar{r} \to 0$. Hence $M_1 \bar{r} < (M_2 - 1)/2$ when n is large enough. Then the whole ball $\mathbb{B}(\mathbf{Z}; \bar{r})$ is contained in $A^c_{M_3}$ with $M_3 = (M_2 + 1)/2$ as before. Hence $\bar{p} = \int_{\mathbb{B}(\mathbf{Z};\bar{r})} c(\mathbf{z}) d\mathbf{z} \geq M_3 v_d \bar{r}^d = M_3 k/n$. Using Chebyshev's inequality,

$$\begin{aligned} Pr[\hat{c}_{knn}(\mathbf{Z}) < 1] &= E[\mathbb{1}\{\bar{N}(\mathbf{Z}) < k\}] \leq \tfrac{Var[\bar{N}(\mathbf{Z})]}{\{E[\bar{N}(\mathbf{Z})] - k\}^2} = \tfrac{n\bar{p}(1-\bar{p})}{(n\bar{p}-k)^2} \\ &\leq \tfrac{1}{n\bar{p}[1-k/(n\bar{p})]^2} \leq \tfrac{1}{M_3 k[1-1/M_3]^2} \\ &= O(\tfrac{1}{k}). \end{aligned}$$

Hence

$$E|T_1(c) - T_1(\hat{c}_{knn})| = \int_{A^c_{M_2}} E[1_{\{\hat{c}_{knn}(\mathbf{Z}) < 1\}}] d\mathbf{Z} \leq \frac{1}{M_3 k[1 - 1/M_3]^2} = O(\frac{1}{k}).$$

Secondly, we look at the error bound on $A_{M_2,\epsilon}$, the region of middle copula density. Using (21), for $\mathbf{Z} \in A_{M_2,\epsilon}$, the mean squared error of $\hat{c}_{knn}(\mathbf{Z})$ is

$$\begin{aligned} E[(\hat{c}_n(\mathbf{Z}) - c(\mathbf{Z}))^2] &= bias^2(\mathbf{Z}) + Var(\mathbf{Z}) \\ &= [\frac{\tilde{Q}(\mathbf{Z})}{c(\mathbf{Z})^{2/d}} (\frac{k}{n})^{2/d}]^2 + \frac{c^2(\mathbf{Z})}{k} + o((\frac{k}{n})^{4/d} + \frac{1}{k}) \\ &\leq (\frac{\bar{Q}^2}{\epsilon^{4/d}})(\frac{k}{n})^{4/d} + \frac{M_2^2}{k} + o((\frac{k}{n})^{4/d} + \frac{1}{k}). \end{aligned}$$

Hence

$$E|\hat{c}_{knn}(\mathbf{Z}) - c(\mathbf{Z})| \leq \sqrt{E[(\hat{c}_n(\mathbf{Z}) - c(\mathbf{Z}))^2]} \leq \sqrt{2}[\bar{Q}(\frac{k}{n\epsilon})^{2/d} + \frac{M_2}{\sqrt{k}}][1 + o(1)].$$

We get

$$E[T_2(\hat{c}_{knn}) - T_2(c)] \leq E[\int_{A_{M_2,\epsilon}} |\hat{c}_{knn}(\mathbf{Z}) - c(\mathbf{Z})| d\mathbf{Z}] \leq \sqrt{2}[\bar{Q}(\frac{k}{n\epsilon})^{2/d} + \frac{M_2}{\sqrt{k}}][1 + o(1)]. \quad (22)$$

Thirdly, we look at the error bound on A_ϵ, the region of small copula density. From equation (20), $\hat{c}_{knn}(\mathbf{Z}) \geq 2\epsilon$ if and only if

$$r \leq (\frac{k}{n 2\epsilon v_d})^{1/d} \stackrel{\text{def}}{=} r^*.$$

This occurs when at least k of observations $\mathbf{Z}_1, \mathbf{Z}_2, \cdots, \mathbf{Z}_n$ fall into the ball $\mathbb{B}(\mathbf{Z}; r^*)$. Since $k/(n\epsilon) \to 0$, $r^* \to 0$.

Let $N^*(\mathbf{Z})$ denotes the number of observations falling into $\mathbb{B}(\mathbf{Z}; r^*)$. Then $\bar{N}(\mathbf{Z})$ follows a binomial distribution with mean np^*, where $p^* = \int_{\mathbb{B}(\mathbf{Z};r^*)} c(\mathbf{z}) d\mathbf{z}$.

Using Taylor expansion, we have (from last line page 228 in Biau et al. (2011))

$$\int_{\mathbb{B}(\mathbf{Z};r)} c(\mathbf{z}) d\mathbf{z} = c(\mathbf{Z}) v_d r^d + \tilde{Q}(\mathbf{Z}) v_d r^{d+2} + o(r^{d+2}).$$

Therefore, using $r^* \to 0$, we have $p^* = c(\mathbf{Z}) v_d (r^*)^d [1 + o(1)] \leq \epsilon v_d (r^*)^d [1 + o(1)]$ which converges to $k/(2n)$. Hence for n big, $p^* < 0.6k/n$. Using Chebyshev's inequality,

$$\begin{aligned} Pr[\hat{c}_{knn}(\mathbf{Z}) > 2\epsilon] &= E[\mathbb{1}\{N^*(\mathbf{Z}) < k\}] \leq \tfrac{Var[N^*(\mathbf{Z})]}{[k - E(N^*(\mathbf{Z}))]^2} = \tfrac{np^*(1-p^*)}{(k-np^*)^2} \\ &\leq \tfrac{0.6k}{(0.4k)^2} < \tfrac{3}{k} \\ &= O(\tfrac{1}{k}). \end{aligned}$$

Since $c(\mathbf{Z}) \leq \epsilon$, if $\hat{c}_{knn}(\mathbf{Z}) \leq 2\epsilon$, then $|\hat{c}_{knn}(\mathbf{Z}) - c(\mathbf{Z})| \leq 2\epsilon$. Hence

$$\begin{aligned} E|T_3(c) - T_3(\hat{c}_{knn})| &\leq \textstyle\int_{A_\epsilon} E|\hat{c}_{knn}(\mathbf{Z}) - c(\mathbf{Z})| d\mathbf{Z} \leq \int_{A_\epsilon} \{2\epsilon + Pr[\hat{c}_{knn}(\mathbf{Z}) > 2\epsilon]\} d\mathbf{Z} \\ &< 2\epsilon + \tfrac{3}{k} = O(\epsilon + \tfrac{1}{k}). \end{aligned}$$

Finally, when combining the three parts, the terms $O(1/k) = o(1/\sqrt{k}) < (2 - \sqrt{2})/\sqrt{k}$. Hence we arrive at

$$\sup_{C \in \mathfrak{C}} E[|\widehat{RCD} - RCD|] \leq 2[\bar{Q}(\frac{k}{n\epsilon})^{2/d} + \frac{M_2}{\sqrt{k}} + \epsilon], \quad (23)$$

which finished the proof.

Note that we can use other l_p norms, which changes the v_d in the proof to the volume of the unit ball under the corresponding norm. The rate does not change.

We can also prove the consistency under Hölder condition without assuming continuous second derivatives. However, that involve tedious derivation of bias and variance bounds similar to (21) for k-NN density estimators. We provide the simple proof here by citing (21) from Mack and Rosenblatt (1979).

To minimize the error bound in (12), we get $\epsilon = (k/n)^{2/(d+2)}$ and $k = n^{4/(d+6)}$. So in bivariate ($d = 2$) case, we take $k = O(n^{1/2})$. Taking k below the $n^{4/(d+6)}$ rate will make the $O(1/\sqrt{k})$ term dominant in the error bound. In that case, the asymptotic results on the k-NN density estimation states that $\sqrt{k}[\hat{c}(\mathbf{Z}) - c(\mathbf{Z})]/c(\mathbf{Z})$ converge to a standard Gaussian distribution. Then $\sqrt{k}[\widehat{RCD} - RCD]$ converges to an integral of a Gaussian process.

Appendix E. Selection of Tuning Parameter in the Practical Estimator

For a practical estimator for $\widehat{RCD}$, we need to decide the bandwidth in KDE-based estimator or the number of neighbors k in KNN-based estimator. Theorem 4 and Theorem 5 provides the rates. For bivariate case, $h = O(n^{-1/4})$ and $k = O(\sqrt{n})$. To decide the constant coefficient, we used empirical simulations.

First, for KDE estimators, we tested $\widehat{RCD}$ on nine functions (listed in Table 11) with various levels of additive noises. Four sample sizes of $n = 10^2$, 10^3, 10^4 and 10^5 are used.Figure 13 plots

the simulation results using $h = 0.25n^{-1/4}$. We can see that the performance of $\widehat{RCD}$ improves as sample size increases, and gives very accurate estimates for RCD under big sample sizes. For illustration, we showed the plots with bandwidth $h = 0.1n^{-1/4}$ and $h = 0.5n^{-1/4}$ in Figure 14 and Figure 15 respectively. Those bandwidth choices are clearly either too small ($h = 0.1n^{-1/4}$ estimator overshoot in several cases when RCD is small) or too big ($h = 0.5n^{-1/4}$ estimator converges slowly when RCD is large). Hence the bandwidth $h = 0.25n^{-1/4}$ is a good choice.

A	Linear	$y = x$
B	Quadratic	$y = x^2$
C	Square Root	$y = \sqrt{x}$
D	Cubic	$y = x^3$
E	Centered Cubic	$y = 4(x - 1/2)^3$
F	Centered Quadratic	$y = 4x(1 - x)$
G	Cosine (Period 1)	$y = [\cos(2\pi x) + 1]/2$
H	Circle	$(x - 1/2)^2 + y^2 = 1/4$
I	Cross	$y = \pm(x - 1/2)$

Table 11: The function relationships used in Figures 13 - 18.

According to the equivalence results between the KDE and the KNN estimator by (Moore and Yackel, 1977b), the k in the KNN density estimation corresponds to the bandwidth in KDE estimator as $c(\mathbf{z})(2h)^2 = k/n$. As the mean of copula density $c(\alpha)$ is one, $h = 0.25n^{-1/4}$ corresponds to $k = n(2h)^2 = 0.25\sqrt{n}$. The simulation results for KNN-based $\widehat{RCD}$ with $k = 0.25\sqrt{n}$, $k = 0.1\sqrt{n}$ and $k = 0.5\sqrt{n}$ are plotted in Figures 16 - 18. Similar pattern as in KDE-based estimator are observed. Hence we propose the practical KNN-based $\widehat{RCD}$ to use $k = 0.25\sqrt{n}$.

Furthermore, we also checked the KNN-based $\widehat{RCD}$ on the mixture noise setting used in definition 2: a proportion (p) of deterministic function is hidden in independent continuous noise. Six types of deterministic function are used, as listed in Table 12. When $n = 5000$, the $\widehat{RCD}$ is close to the true value p in the simulations. And compared to the two choices of $k = 0.1\sqrt{n}$ and $k = 0.5\sqrt{n}$, $k = 0.25\sqrt{n}$ provide a good balance of approximating the true values when RCD is small or large.

A	Linear	$y = x$
B	Centered Quadratic	$y = 4(x - 1/2)^2$
C	Cosine	$y = \cos(4\pi x)$
D	Cross	$y = \pm x 1_{\{0 \le x \le 1\}}$
E	Circle	$(2x - 1)^2 + y^2 = 1$
F	Cross 2	$y = \pm(x - 1/2)1_{\{0 \le x \le 1\}}$

Table 12: The function relationships used in Figures 19.

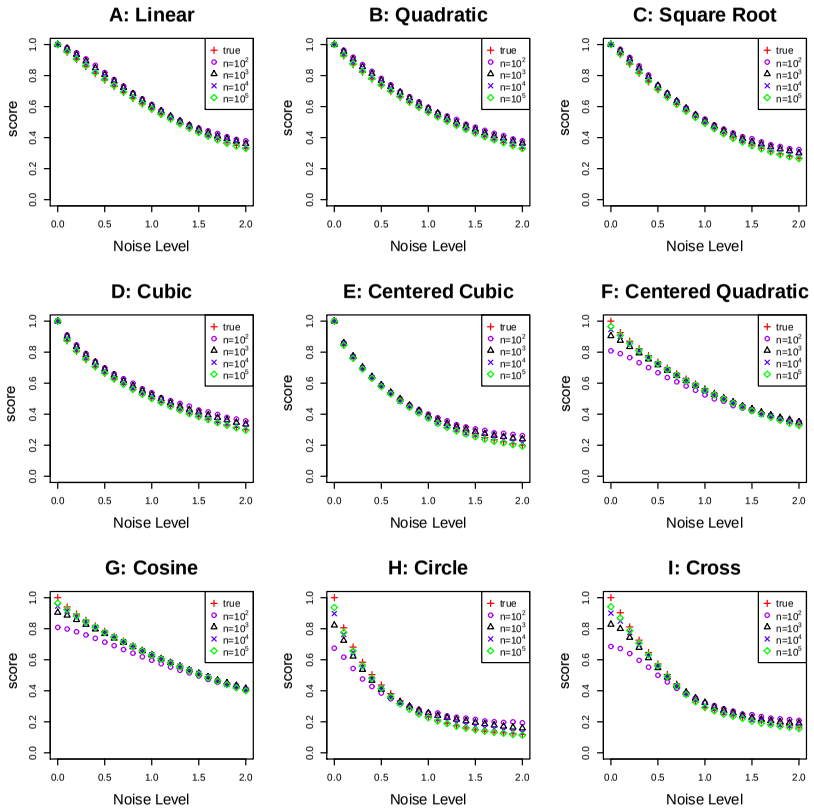


Figure 13: The comparison of RCD with its estimated values under different sample sizes. This estimator uses the square kernel density estimator with bandwidth $h = 0.25n^{-1/4}$.

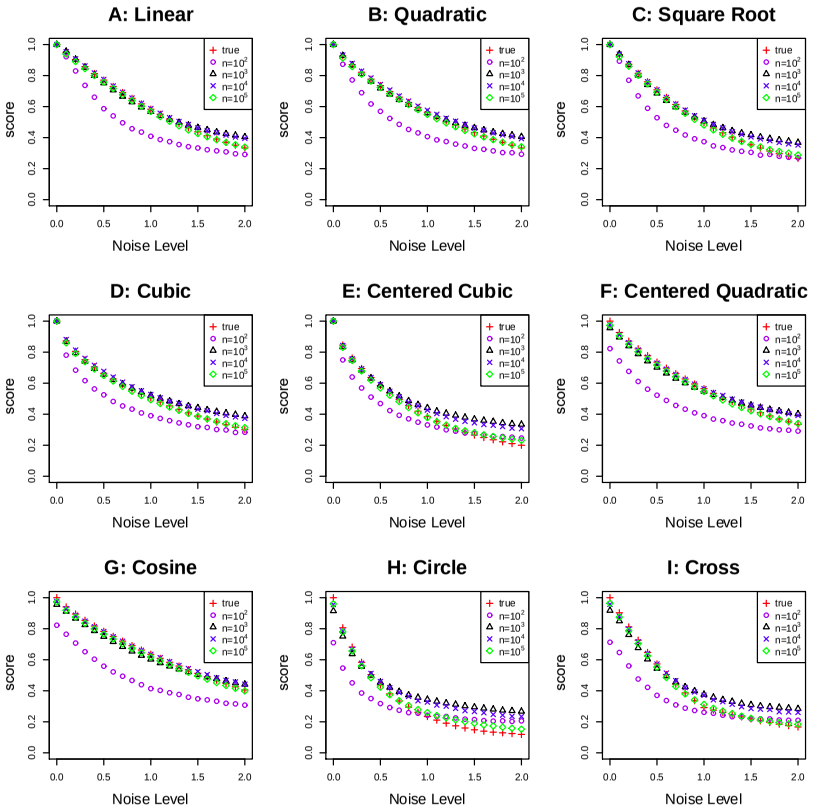


Figure 14: The comparison of RCD with its estimated values under different sample sizes. This estimator uses the square kernel density estimator with bandwidth $h = 0.1n^{-1/4}$.

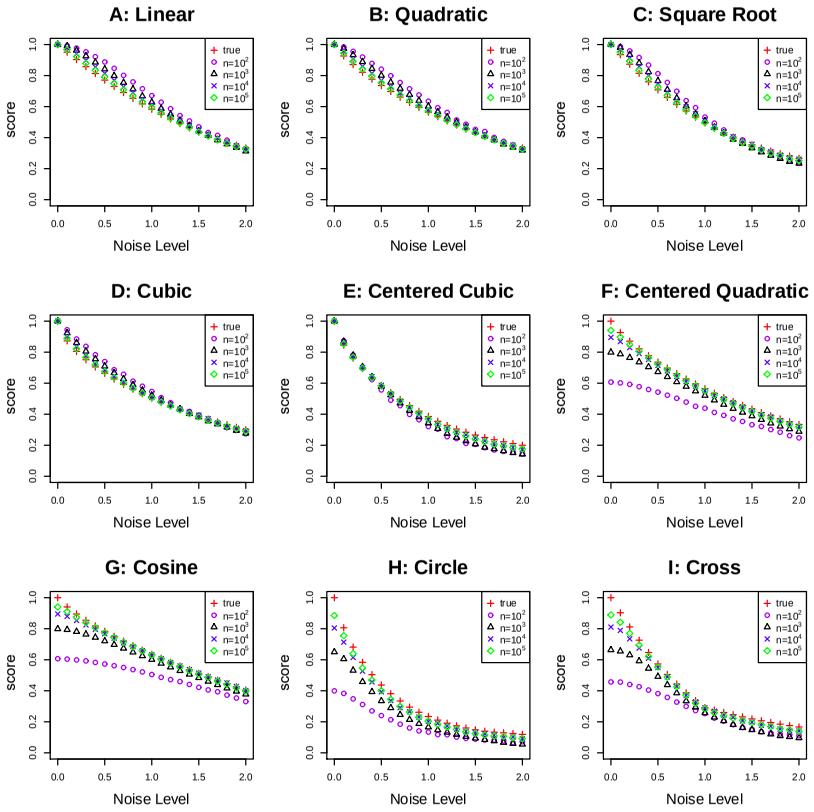


Figure 15: The comparison of RCD with its estimated values under different sample sizes. This estimator uses the square kernel density estimator with bandwidth $h = 0.5n^{-1/4}$.

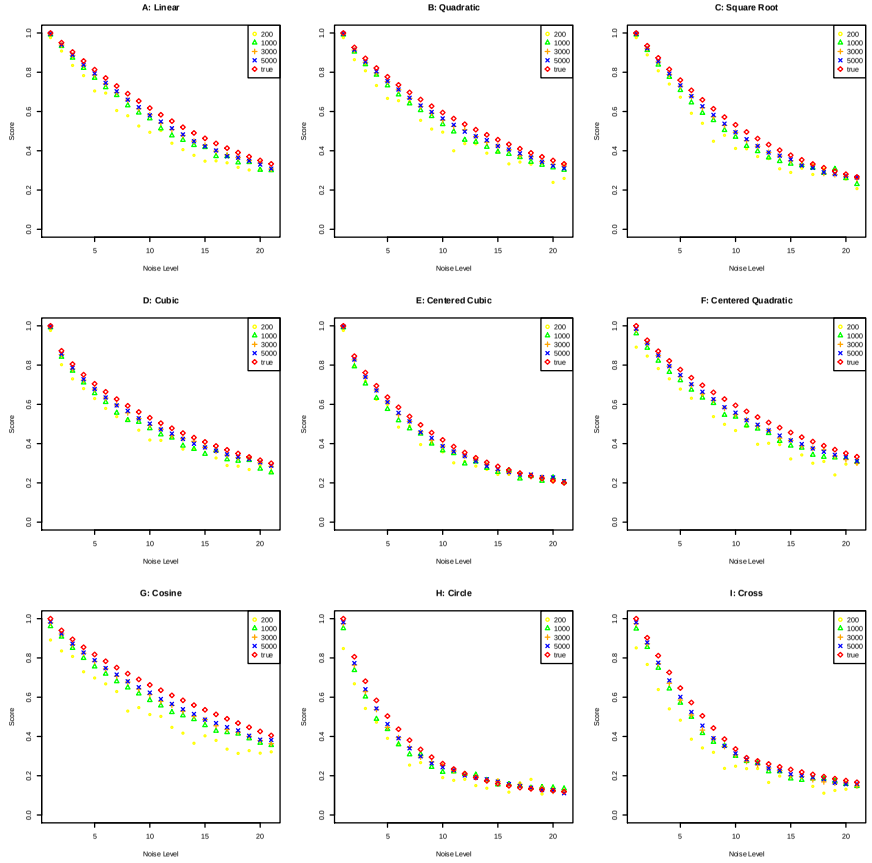


Figure 16: Additive noise with $k = 0.25\sqrt{n}$.

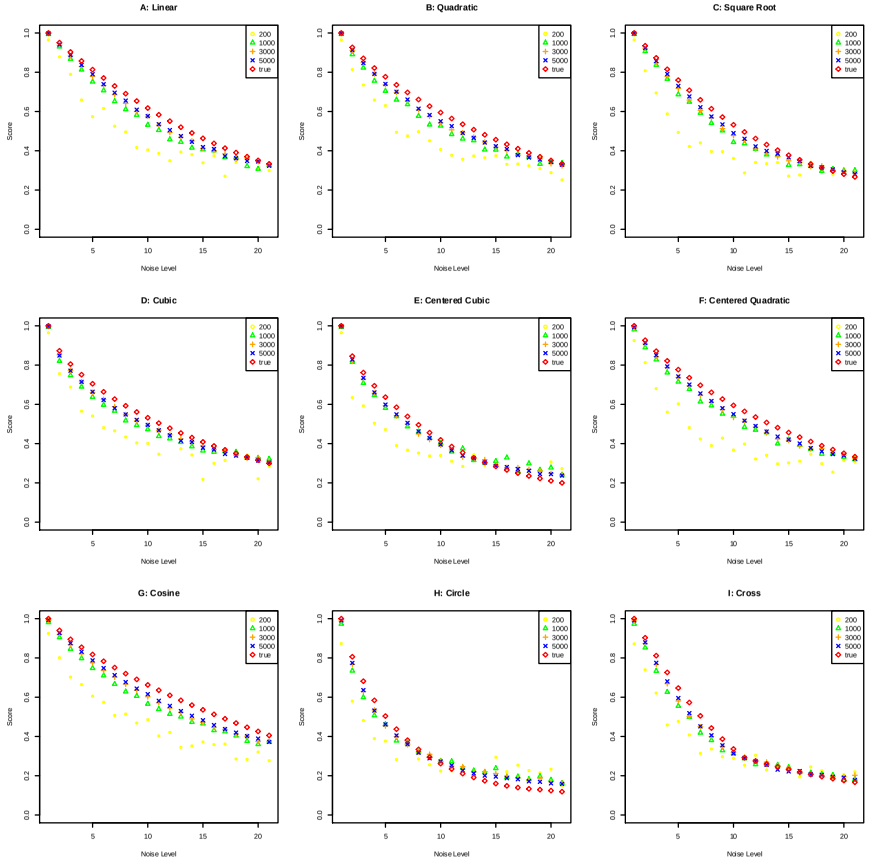


Figure 17: Additive noise with $k = 0.1\sqrt{n}$.

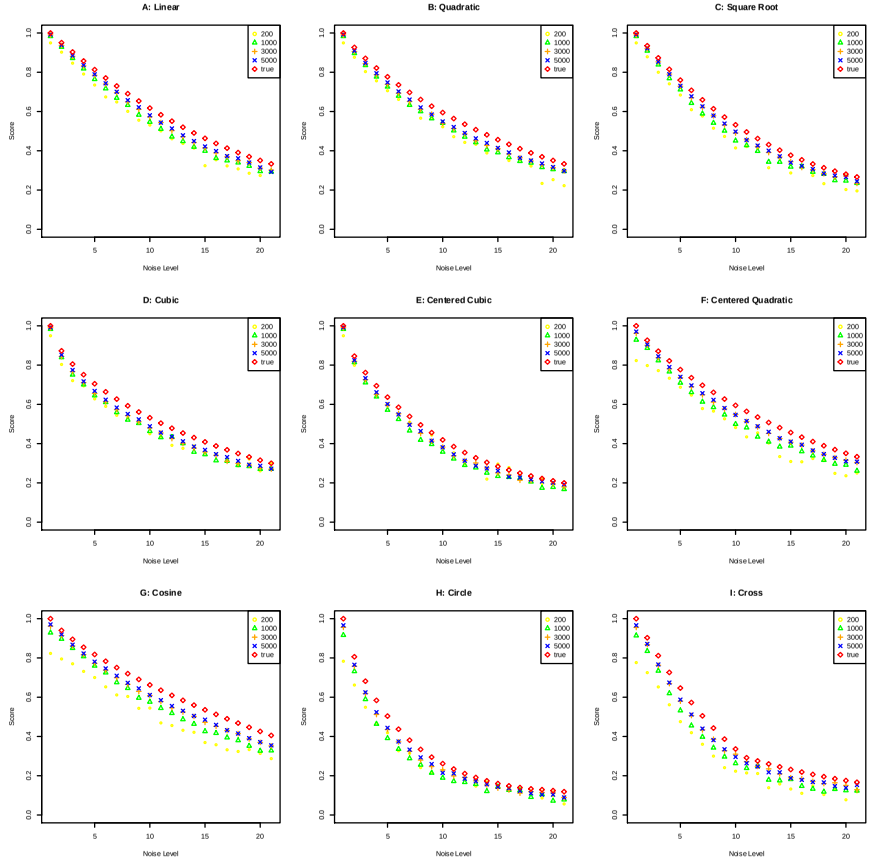


Figure 18: Additive noise with $k = 0.5\sqrt{n}$.

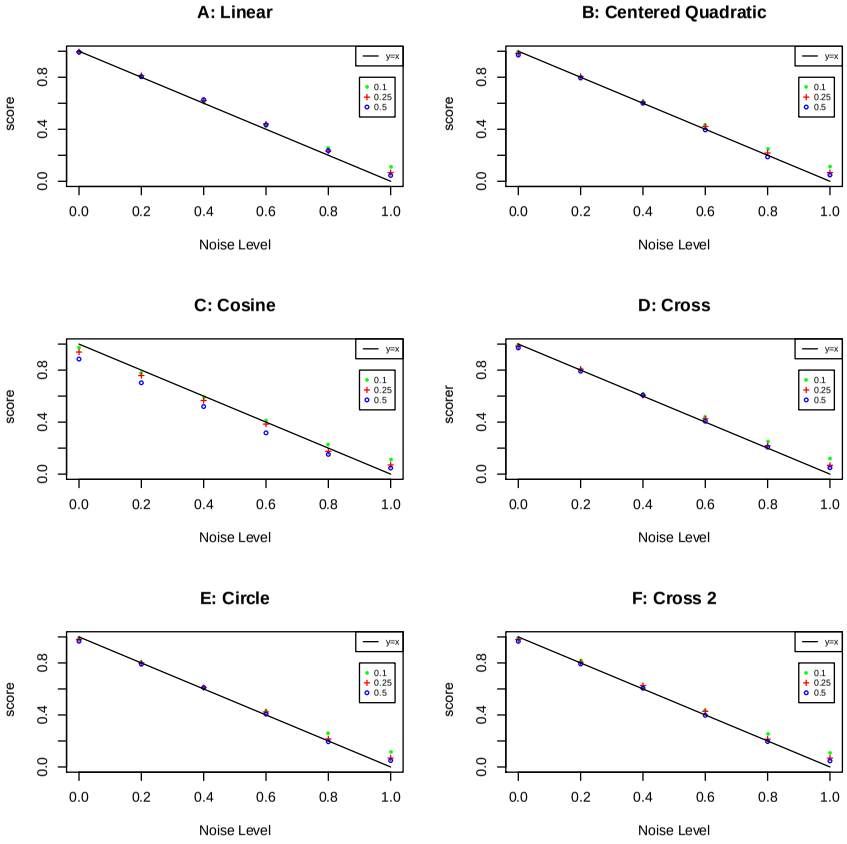


Figure 19: Mixture noise with $k = c\sqrt{n}$, where $c = 0.1, 0.25, 0.5$.

References

András Antos and Ioannis Kontoyiannis. Convergence properties of functional estimates for discrete distributions. *Random Structures & Algorithms*, 19(3-4):163–193, 2001.

Luca Bagnato, Lucio De Capitani, and Antonio Punzo. Testing serial independence via density-based measures of divergence. *Methodology and Computing in Applied Probability*, pages 1–15, 2013. ISSN 1387-5841. doi: 10.1007/s11009-013-9320-4.

Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

Gérard Biau, Frédéric Chazal, David Cohen-Steiner, Luc Devroye, and Carlos Rodríguez. A weighted k-nearest neighbor density estimate for geometric inference. *Electron. J. Statist.*, 5: 204–237, 2011. doi: 10.1214/11-EJS606.

Yale Chang, Yi Li, Adam Ding, and Jennifer Dy. A robust-equitable copula dependence measure for feature selection. In *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Statistics*. Citeseer, 2016.

Thomas H Cormen. *Introduction to algorithms*. MIT press, 2009.

David L. Donoho and Richard C. Liu. Geometrizing rates of convergence, ii. *The Annals of Statistics*, 19(2):pp. 633–667, 1991. ISSN 00905364.

Jennifer G Dy and Carla E Brodley. Feature selection for unsupervised learning. *The Journal of Machine Learning Research*, 5:845–889, 2004.

Magnus Ekdahl and Timo Koski. Bounds for the loss in probability of correct classification under model based approximation. *J. Mach. Learn. Res.*, 7:2449–2480, December 2006. ISSN 1532-4435.

R. H. Farrell. On the best obtainable asymptotic rates of convergence in estimation of a density function at a point. *The Annals of Mathematical Statistics*, 43(1):pp. 170–180, 1972. ISSN 00034851.

Andrew D Fernandes and Gregory B Gloor. Mutual information is critically dependent on prior assumptions: would the correct estimate of mutual information please identify itself? *Bioinformatics*, 26(9):1135–1139, 2010.

Jerome H Friedman. Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67, 1991.

Kenji Fukumizu, Arthur Gretton, Xiaohai Sun, and Bernhard Schölkopf. Kernel measures of conditional dependence. In *NIPS*, volume 20, pages 489–496, 2007.

Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *Algorithmic learning theory*, pages 63–77. Springer, 2005a.

Arthur Gretton, Ralf Herbrich, Alexander Smola, Olivier Bousquet, and Bernhard Schölkopf. Kernel methods for measuring independence. *J. Mach. Learn. Res.*, 6:2075–2129, December 2005b. ISSN 1532-4435.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422, 2002.

Xiaofei He, Deng Cai, and Partha Niyogi. Laplacian score for feature selection. In *Advances in neural information processing systems*, pages 507–514, 2005.

Harry Joe. Relative entropy measures of multivariate dependence. *Journal of the American Statistical Association*, 84(405):157–164, 1989. doi: 10.1080/01621459.1989.10478751.

Justin B Kinney and Gurinder S Atwal. Equitability, mutual information, and the maximal information coefficient. *Proceedings of the National Academy of Sciences*, 111(9):3354–3359, 2014.

Kenji Kira and Larry A Rendell. The feature selection problem: Traditional methods and a new algorithm. In *AAAI*, volume 2, pages 129–134, 1992.

Ron Kohavi and George H John. Wrappers for feature subset selection. *Artificial intelligence*, 97 (1):273–324, 1997.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review E*, 69(6):066138, 2004.

Lucien Le Cam. Convergence of estimates under dimensionality restrictions. *The Annals of Statistics*, pages 38–53, 1973.

M. Lichman. UCI machine learning repository. `http://archive.ics.uci.edu/ml`, 2013.

Han Liu, John D Lafferty, and Larry A Wasserman. Exponential concentration for mutual information estimation with application to forests. In *NIPS*, pages 2546–2554, 2012.

D. O. Loftsgaarden and C. P. Quesenberry. A nonparametric estimate of a multivariate density function. *Ann. Math. Statist.*, 36(3):1049–1051, 06 1965. doi: 10.1214/aoms/1177700079.

Y. P. Mack and M. Rosenblatt. Multivariate k-nearest neighbor density estimates. *Journal of Multivariate Analysis*, 9(1):1–15, 1979.

Young-Il Moon, Balaji Rajagopalan, and Upmanu Lall. Estimation of mutual information using kernel density estimators. *Physical Review E*, 52(3):2318, 1995.

David S. Moore and James W. Yackel. Consistency properties of nearest neighbor density function estimators. *Ann. Statist.*, 5(1):143–154, 01 1977a. doi: 10.1214/aos/1176343747.

David S. Moore and James W. Yackel. Large sample properties of nearest neighbor density function estimators. *Statistical Decision Theory and Related Topics*, II:269–279, 1977b.

Ben Murrell, Daniel Murrell, and Hugh Murrell. R2-equitability is satisfiable. *Proceedings of the National Academy of Sciences*, 111(21):E2160–E2160, 2014.

Ben Murrell, Daniel Murrell, and Hugh Murrell. Discovering general multidimensional associations. *PloS one*, 11(3):e0151551, 2016.

R. B. Nelsen. *An introduction to copulas (Springer series in statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387286594.

Marek Omelka, Irène Gijbels, and Noël Veraverbeke. Improved kernel estimation of copulas: weak convergence and goodness-of-fit testing. *The Annals of Statistics*, 37(5B):3023–3058, 2009.

Dávid Pál, Barnabás Póczos, and Csaba Szepesvári. Estimation of renyi entropy and mutual information based on generalized nearest-neighbor graphs. In *NIPS*, pages 1849–1857, 2010.

Liam Paninski. Estimation of entropy and mutual information. *Neural Computation*, 15(6):1191–1253, 2003.

Hanchuan Peng, Fulmi Long, and Chris Ding. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(8):1226–1238, 2005.

B Poczos, Z Ghahramani, and J Schneider. Copula-based kernel dependency measures. In *International Conference on Machine Learning*, 2012.

Pavel Pudil, Jana Novovičová, and Josef Kittler. Floating search methods in feature selection. *Pattern recognition letters*, 15(11):1119–1125, 1994.

Sashank J Reddi and Barnabás Póczos. Scale invariant conditional dependence measures. In *Proceedings of The 30th International Conference on Machine Learning*, pages 1355–1363, 2013.

A. Rényi. On measures of dependence. *Acta Mathematica Academiae Scientiarum Hungarica*, 10 (3-4):441–451, 1959. ISSN 0001-5954. doi: 10.1007/BF02024507.

David N Reshef, Yakir A Reshef, Hilary K Finucane, Sharon R Grossman, Gilean McVean, Peter J Turnbaugh, Eric S Lander, Michael Mitzenmacher, and Pardis C Sabeti. Detecting novel associations in large data sets. *science*, 334(6062):1518–1524, 2011.

Yakir A. Reshef, David N. Reshef, Pardis C. Sabeti, and Michael Mitzenmacher. Theoretical foundations of equitability and the maximal information coefficient, 2014.

Yakir A Reshef, David N Reshef, Hilary K Finucane, Pardis C Sabeti, and Michael M Mitzenmacher. Measuring dependence powerfully and equitably. *arXiv preprint arXiv:1505.02213*, 2015a.

Yakir A Reshef, David N Reshef, Pardis C Sabeti, and Michael M Mitzenmacher. Equitability, interval estimation, and statistical power. *arXiv preprint arXiv:1505.02212*, 2015b.

B. Schweizer and E. F. Wolff. On nonparametric measures of dependence for random variables. *The Annals of Statistics*, 9(4):pp. 879–885, 1981. ISSN 00905364.

D.W. Scott. *Multivariate density estimation: theory, practice, and visualization*. Wiley Series in Probability and Statistics. Wiley, 1992. ISBN 9780471547709.

Johan Segers. Asymptotics of empirical copula processes under non-restrictive smoothness assumptions. *Bernoulli*, 18(3):764–782, 2012.

Bernard W Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.

S. D. Silvey. On a measure of association. *Ann. Math. Statist.*, 35(3):1157–1166, 09 1964. doi: 10.1214/aoms/1177703273.

Noah Simon and Robert Tibshirani. Comment on detecting novel associations in large data sets by reshef et al, science dec 16, 2011. *Science*, 2011.

Le Song, Alex Smola, Arthur Gretton, Karsten M Borgwardt, and Justin Bedo. Supervised feature selection via dependence estimation. In *Proceedings of the 24th international conference on Machine learning*, pages 823–830. ACM, 2007.

Terry Speed. A correlation for the 21st century. *Science*, 334(6062):1502–1503, 2011. doi: 10.1126/science.1215894. URL `http://www.sciencemag.org/content/334/6062/1502.short`.

Ning Sun and Hongyu Zhao. Putting things in order. *Proceedings of the National Academy of Sciences*, 111(46):16236–16237, 2014. doi: 10.1073/pnas.1418862111. URL `http://www.pnas.org/content/111/46/16236.short`.

Taiji Suzuki, Masashi Sugiyama, and Toshiyuki Tanaka. Mutual information approximation via maximum likelihood estimation of density ratio. In *ISIT*, volume 9, pages 463–467, 2009.

Gábor J Székely and Maria L Rizzo. Brownian distance covariance. *The annals of applied statistics*, pages 1236–1265, 2009.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Vladimir Naumovich Vapnik and Vlamimir Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

Toomas R. Vilmansen. On dependence and discrimination in pattern recognition. *IEEE Transactions on Computers*, 21(9):1029–1031, September 1972. ISSN 0018-9340. doi: 10.1109/TC.1972.5009090.

T.R. Vilmansen. Feature evaluation with measures of probabilistic dependence. *IEEE Transactions on Computers*, 22(4):381–388, April 1973. ISSN 0018-9340. doi: 10.1109/T-C.1973.223725.

Lei Yu and Huan Liu. Efficient feature selection via analysis of relevance and redundancy. *The Journal of Machine Learning Research*, 5:1205–1224, 2004.

Journal of Machine Learning Research 18 (2017) 1-32 Submitted 3/16; Revised 6/17; Published 8/17

Multiscale Strategies for Computing Optimal Transport

Samuel Gerber[1] samuel.gerber@kitware.com
Mauro Maggioni[2,3,4] mauro.maggioni@jhu.edu
[1]*Kitware, NC, U.S.A*
Department of [2]*Mathematics,* [3]*Applied Mathematics,* [4]*Institute for Data Intensive Engineering and Science, Johns Hopkins University, Baltimore, MD, U.S.A.*

Editor: Nikos Vlassis

Abstract

This paper presents a multiscale approach to efficiently compute approximate optimal transport plans between point sets. It is particularly well-suited for point sets that are in high-dimensions, but are close to being intrinsically low-dimensional. The approach is based on an adaptive multiscale decomposition of the point sets. The multiscale decomposition yields a sequence of optimal transport problems, that are solved in a top-to-bottom fashion from the coarsest to the finest scale. We provide numerical evidence that this multiscale approach scales approximately linearly, in time and memory, in the number of nodes, instead of quadratically or worse for a direct solution. Empirically, the multiscale approach results in less than one percent relative error in the objective function. Furthermore, the multiscale plans constructed are of interest by themselves as they may be used to introduce novel features and notions of distances between point sets. An analysis of sets of brain MRI based on optimal transport distances illustrates the effectiveness of the proposed method on a real world data set. The application demonstrates that multiscale optimal transport distances have the potential to improve on state-of-the-art metrics currently used in computational anatomy.

1. Introduction

The study of maps between shapes, manifolds and point clouds is of great interest in a wide variety of applications. There are many data types, e.g. shapes (modeled as surfaces), images, sounds, and many more, where a similarity between a pair of data points involves computing a map between the points, and the similarity is a functional of that map. The map between a pair of data points however contains much more information than the similarity measure alone, and the study of networks of such maps have been successfully used to organize, extract functional information and abstractions, and help regularize estimators in large collections of shapes (M. Ovsjanikov and Guibas, 2010; Qixing Huang, 2014; Huang and Guibas, 2013). The family of maps to be considered depends on the type of shape, manifold or point cloud, as well as on the choice of geometric features to be preserved in a particular application. These considerations are not restricted to data sets where each point is naturally a geometric object: high-dimensional data sets of non-geometric nature, from musical pieces to text documents to trajectories of high-dimensional stochastic dynamical systems, are often mapped, via feature sets, to geometric objects. The considerations above therefore apply to a very wide class of data types.

©2017 Samuel Gerber and Mauro Maggioni.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-108.html.

In this paper we are interested in the problem where each object is a point cloud – a set of points in $\mathbb{R}^D$ – and will develop techniques for computing maps from one point cloud to another, in particular in the situation where D is very large, but the point clouds are close to being low-dimensional, for example they may be samples from a d-dimensional smooth manifold $\mathcal{M}$ ($d \ll D$). The two point clouds may have a different number of points, and they may arise from a sample of a low-dimensional manifold $\mathcal{M}$ perturbed by high-dimensional noise (for more general models see the works by Little et al. (2012), Maggioni et al. (2016) and Liao and Maggioni (2016)). In this setting we have to be particularly careful in both the choice of maps and in their estimation since sampling and noise have the potential to cause significant perturbations.

We find optimal transport maps rather well-suited for these purposes. They automatically handle the situation where the two point clouds have different cardinality, they handle in a robust fashion noise, and even changes in dimensionality, which is typically ill-defined, for point clouds arising from real-world data (Little et al., 2012). Optimal transport has a very long history in a variety of disciplines and arises naturally in a wide variety of contexts, from optimization problems in economics and resource allocation, to mathematics and physics, to computer science (e.g. network flow algorithms). Thus, applications of optimal transport range from logistics and economics (Beckmann, 1952; Carlier et al., 2008), geophysical models (Cullen, 2006), image analysis (Rubner et al., 1998; Haker et al., 2004) to machine learning (Cuturi and Doucet, 2014; Cuturi and Avis, 2014). Despite these widespread applications, the efficient computation of optimal transport plans remains challenging, especially in complex geometries and in high dimensions.

For point sets the optimal transport problem can be solved by a specialized linear program, the minimum network flow problem (Ahuja et al., 1993; Tarjan, 1997). The minimum network flow problem has been extensively studied in the operations research community and several fast algorithms exist. However, these algorithms, at least on desktop hardware, do not scale beyond a few thousand source and target points. Our framework extends the applications of these algorithms to problem instances several orders of magnitude larger, under suitable assumptions on the geometry of the data. We exploit a multiscale representation of the source and target sets to reduce the number of variables in the linear program and quickly find good initial solutions, as illustrated in Figure 1. The optimal transport problem is solved (with existing algorithms) at the coarsest scale and the solution is propagated to the next scale and refined. This process is repeated until the finest scale is reached. This strategy, discussed in detail in Section 3, is adaptable to memory limitations and speed versus accuracy trade-offs. For some of the refinement strategies, it is guaranteed to converge to the optimal solution.

Our approach draws from a varied set of work that is briefly summarized in Section 2. The proposed approach generalizes and builds on previous and concurrently developed hierarchical methods (Glimm and Henscheid, 2013; Schmitzer and Schnörr, 2013; Schmitzer, 2015; Oberman and Ruan, 2015). The work in this paper adds the following contributions:

- The description of a general multiscale framework for discrete optimal transport that can be applied in conjunction with a wide range of optimal transport algorithms.
- A set of propagation and refinement heuristics, including approaches that are similar and/or refine existing ones (Glimm and Henscheid, 2013; Oberman and Ruan, 2015; Schmitzer, 2015) as well as novel ones. In particular we propose a novel propagation

strategy based on capacity restrictions of the network flow problem at each scale. This new approach proves to be very efficient and accurate in practice. Overall, the heuristics result empirically in a linear increase in computation time with respect to data set size.

- An implementation in the R package *mop* that allows the combination of multiple heuristics to tailor speed and accuracy to the requirements of particular applications.

Compared to other linear programming based approaches, the multiscale approach results in a speedup of up to multiple orders of magnitude in large problems and permits to solve approximately transportation problems of several orders of magnitudes larger than previously possible. Comparing to PDE based approaches is difficult and PDE based methods are limited to low-dimensional domains and specific cost metrics. The proposed framework is demonstrated on several numerical examples and compared to the state-of-the-art approximation algorithm by Cuturi (2013).

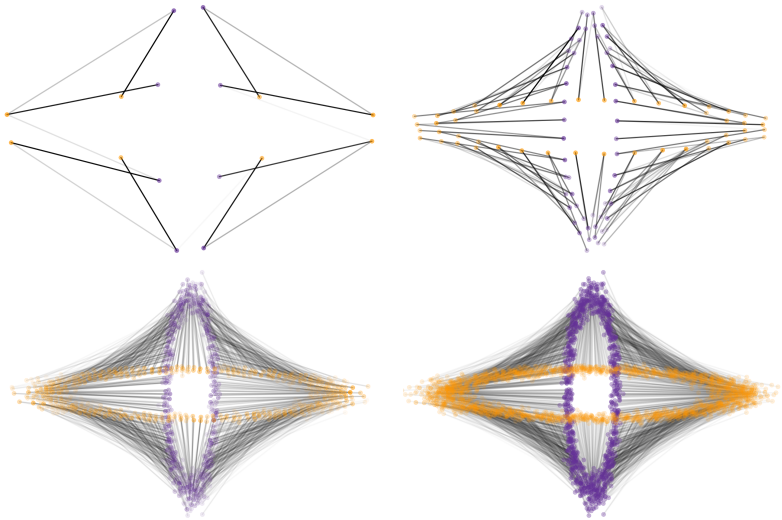


Figure 1: Optimal transport between two noisy elliptical shapes in 2-D. We visualize coarse-to-fine approximations to the source (purple dots) and target (yellow dots) point clouds, together with the optimal transportation plan at each scale (gray edges). The intensity of the points is proportional to mass – at coarser scales each point represents an agglomerate of points at finer scales, and its mass is the sum of the masses of the points it represents. Similarly the intensity of the lines is proportional to the amount of mass transported along the line. Notice that the optimal transport formulation permits multiple lines exiting a source point or entering a target point.

2. Background

Optimal transport is the problem of minimizing the cost of moving a source probability distribution to a target probability distribution given a function that assigns costs to moving mass from source to target locations. The classical formulation by Monge (1781) considers the minimization over mass preserving mappings. Later Kantorovitch (1958) considered the same problem but as a minimization over couplings between the two probability measures, which permits to split mass from a single source across multiple target locations. More precisely, for two probability measures μ and ν on probability spaces $\mathbf{X}$ and $\mathbf{Y}$ respectively, a coupling of μ and ν is a measure π on $\mathbf{X} \times \mathbf{Y}$ such that the marginals of π are μ and ν, i.e. $\pi(A \times \mathbf{Y}) = \mu(A)$ for all (μ-measurable) sets $A \subseteq \mathbf{X}$ and $\pi(\mathbf{X} \times B) = \nu(B)$ for all (ν-measurable) sets $B \subseteq \mathbf{Y}$. We denote by $\mathcal{C}(\mu, \nu)$ the set of couplings between μ and ν. Informally, we may think of $d\pi(x, y)$ as the amount of infinitesimal mass to be transported from source x to destination y, with the condition that π is a coupling guaranteeing that the source mass is distributed according to μ and the destination mass is distributed according to ν. Such couplings always exist: we always have the trivial coupling $\pi = \mu \times \nu$. The trivial coupling is uninformative, every source mass $d\mu(x)$ is transported to the same target distribution ν. In Monge's formulation the coupling is restricted to the special form $\pi(x, y) = \delta_{T(x)}$ where δ_y is the Dirac-δ measure with mass at $y \in \mathbf{Y}$, and T is a function $\mathbf{X} \to \mathbf{Y}$: in this case the coupling is "maximally informative" in the sense that there is a function mapping each source x to a single destination y; in particular the mass $d\mu(x)$ at x is not split into multiple portions that are shipped to different target y's.

To define optimal transport and optimal couplings, we need a cost function $\mathsf{c}(x, y)$ on $\mathbf{X} \times \mathbf{Y}$ representing the work or cost needed to move a unit of mass from x to y. Then for every coupling π we may define the cost of π to be

$$\mathbb{E}_{(X,Y)\sim\pi}[\mathsf{c}(X, Y)]\,, \tag{1}$$

with (X, Y) being a pair of random variables distributed according to π. An optimal coupling π minimizes this cost over all choices of couplings. When seeking an optimal transportation plan, the above becomes

$$\mathbb{E}_{X\sim\mu}[\mathsf{c}(X, T(X))] = \int_{\mathbf{X}} \mathsf{c}(x, T(x)) d\mu(x)\,,$$

and one minimizes over all (measurable) functions $T : \mathbf{X} \to \mathbf{Y}$. One often designs the cost function c in an application-dependent fashion, and the above framework is extremely general. When $\mathbf{X} = \mathbf{Y}$ is a metric space with respect to a distance ρ (with suitable technical conditions relating the metrics and the measures μ, ν that we will not delve into, referring the reader to Villani (2009)), then distinguished choices for the cost are those that are related to the metric structure. The natural choice of $\mathsf{c}(x, y) = \rho(x, y)^p$, for some $p > 0$ leads to the definition of the Wasserstein-Kantorovich-Rubinstein metric on the space of probability measures on $\mathbf{X}$:

$$W_p(\mu, \nu) := \min_{\pi\in\mathcal{C}(\mu,\nu)} \left(\int_{\mathbf{X}} \int_{\mathbf{X}} \rho(x, y)^p d\pi(x, y) \right)^{\frac{1}{p}}. \tag{2}$$

Computational solutions to optimal transport split roughly in two settings: Approaches based on the solution of partial differential equations derived from the continuous optimal

transport formulation, briefly discussed in Section 2.1 and, more relevant to this paper, combinatorial optimization methods to directly solve for a discrete optimal transport plan discussed in Section 2.2.

2.1 Continuous Optimal Transport

In case that at least the source distribution admits a density, and when the cost function is the squared Euclidean distance, the optimal coupling is deterministic, i.e. there exists a transport map, and the optimal solution is the gradient of a convex function (Brenier, 1991). This has been exploited to solve the optimal transport problem by numerical partial differential equation approaches (Benamou and Brenier, 2000; Angenent et al., 2003; Haker et al., 2004; Iollo and Lombardi, 2011; Papadakis et al., 2013; Benamou et al., 2014).

An alternative formulation proposed by Aurenhammer et al. (1998) shows that the optimal transport from a source density to a target distribution of a set of weighted Dirac delta's can be solved through a finite dimensional unconstrained convex optimization. Mérigot (2011) proposes a multiscale approach for the formulation of Aurenhammer et al. (1998).

Both the numerical PDE based approaches as well as the unconstrained convex optimization require a discretization of the full domain which is generally not feasible for higher dimensional domains. For arbitrary cost functions and distributions the optimal transport problem does typically not result in a deterministic coupling and can not be solved through PDE based approaches.

2.2 Discrete Optimal Transport and Linear Programming

For two discrete distributions $\mu = \sum_1^n w(x_i)\delta(x_i)$ and $\nu = \sum_1^m v(y_i)\delta(y_i)$ with $\sum w(x_i) = \sum v(y_i) = 1$ the optimal transport problem is equivalent to the linear program

$$\min_{\pi} \sum_{\substack{i=1,\dots,n \\ j=1,\dots,m}} \mathtt{c}(x_i, y_j)\pi(x_i, y_j) \quad \text{s.t.} \quad \begin{cases} \sum_j \pi(x_i, y_j) = \mu(\{x_i\}) = w(x_i) \\ \sum_i \pi(x_i, y_j) = \nu(\{y_j\}) = v(y_j) \\ \pi(x_i, y_j) \geq 0 \end{cases} . \tag{3}$$

The solution is called an optimal coupling π^*, and the minimum value attained at π^*, is called the cost of π^*, or the optimal cost of the transport problem, and is denoted by $\mathtt{cost}(\pi^*) = \sum_{i,j} \mathtt{c}(x_i, y_j)\pi^*(x_i, y_j)$. The constraints enforce that π is a coupling. The variables $\pi(x_i, y_j)$ correspond to the amount of mass transported from source x_i to target y_j, at cost $\mathtt{c}(x_i, y_j)$. The linear constraints are of rank $n + m - 1$: when $n + m - 1$ of the constraints are satisfied, either the n constraints of the source density or the m constraints of the target density are satisfied. Since the sum of outgoing mass is equal to the sum of incoming mass, i.e. $\mu(\mathbf{X}) = \nu(\mathbf{Y})$, it follows that all constraints must be satisfied. The optimal solution lies, barring degeneracies, on a corner of the polytope defined by the constraints, i.e. is a basic feasible solution. This implies that exactly $n + m - 1$ entries of the optimal coupling π are non-zero, i.e. π is a sparse matrix. The optimal coupling is a Monge transport map if and only if all the mass of a source x_i is transported to exactly one target location. A Monge solution does not exist for every optimal transport problem, in fact a small perturbation of μ will always suffice to make a Monge solution impossible.

For source and target data sets with the same cardinality and equally weighted δ-functions, Kantorovich's optimal transport problem reduces to an assignment problem, whose solution is a Monge optimal transport map. In this special case, the optimal transport problem can be efficiently solved by the Hungarian algorithm (Kuhn, 1955). The assignment problem results in a degenerate linear program since only n entries are non-zero (instead of $2n - 1$). We can therefore think of optimal transport as a robust version of assignments. This can also be seen from the point of view of convexity: in the assignment problem, π is a permutation matrix deciding to which y_j each x_i is transported. The convex hull of permutation matrices is exactly the set of doubly-stochastic matrices, to which a coupling π belongs as a consequence of the constraints in (3).

For point sets with different cardinalities and/or points with different masses the optimal transport problem can be solved by a linear program and is a special case of the minimum cost network flow problem. The minimum cost flow problem is well studied and a number of algorithms (Ford and Fulkerson, 1956; Klein, 1967; Cunningham, 1976; Goldberg and Tarjan, 1987; Bertsekas and Tseng, 1988; Orlin, 1997) exist for its solution. This discrete solution approach is not constrained to specific cost functions and can work with arbitrary cost functions.

However, the linear programming approach neglects possibly useful geometric properties of the measures and the cost function. Our work makes assumptions about the underlying geometry of the measure spaces and the associated cost function, and in this way is a mixing of the low-dimensional "geometric PDE" approaches with the discrete non-geometric optimization approaches. It exploits the geometric assumptions to relieve the shortcomings of either approach, namely it scales to high-dimensional data, provided that the intrinsic dimension is low in a suitable sense, and does not require a mesh data structure. At the same time we use the geometry of the data to speed up the linear program, which per–se does not leverage geometric structures.

The refinement strategies of the proposed multiscale approach add subsets of paths among all pairwise paths at each subsequent scale to improve the optimal transport plan. This strategy of adding paths, is akin to column generation approaches (Desrosiers and Lübbecke, 2005). Column generation, first developed by Dantzig and Wolfe (1960) and Ford and Fulkerson (1956), reduces the number of variables in a linear program by solving a smaller linear program on a subset of the original variables and introducing new variables on demand. However, the proposed approach exploits the geometry of the problem instead of relying on an auxiliary linear program (Dantzig and Wolfe, 1960) or shortest path computations (Ford and Fulkerson, 1956) to detect the entering variables.

2.3 Approximation Strategies

In the computer vision literature, the Earth Movers distance or equivalently the Wasserstein-1 distance, which is simply the cost of the optimal coupling, is a successful similarity measure for image retrieval (Rubner et al., 1998). In this application the transport plan is not of interest but only the final transport cost. For this purpose Indyk and Thaper (2003), Shirdhonkar and Jacobs (2008) and Andoni et al. (2008) developed algorithms that compute an approximate cost but do not yield a transport plan. Some of these approaches are based on the dual formulation of optimal transport, which involves testing against Lipschitz func-

tions, and observing that Lipschitz functions may be characterized by decay properties of their wavelet coefficients. In this sense these approaches are multiscale as well.

To speed up computations in machine learning applications Cuturi (2013) proposes to *smooth* transport plans by adding a maximum entropy penalty to the optimal transport formulation. The resulting optimization problem is efficiently solved through matrix scaling with Sinkhorn fixed-point iterations. Because of the added regularization term, the solution will in general be different from the optimal transportation. It may however be the case that these particular (or perhaps other) regularized solutions are better suited for certain applications.

2.4 Related Work

Very recently a number of approaches have been proposed to solve the optimal transport in a multiscale fashion (Glimm and Henscheid, 2013; Schmitzer and Schnörr, 2013; Schmitzer, 2015; Oberman and Ruan, 2015). Glimm and Henscheid (2013) design an iterative scheme to solve a discrete optimal transport problem in reflector design and propose a heuristic for the iterative refinements based on linear programming duality. This iterative scheme can be interpreted as a multiscale decomposition of the transport problem based on geometry of the source and target sets. The proposed potential refinement strategy extends the heuristic proposed by Glimm and Henscheid (2013) to guarantee optimal solutions and adds a more efficient computation strategy: Their approach requires to check all possible variables at the next scale. In Section 3.4.1 we introduce a variation of the approach by Glimm and Henscheid (2013) by adding a branch and bound strategy to avoid checking all variables, and an iterative procedure that guarantees optimal solutions.

Schmitzer and Schnörr (2013) propose a multiscale approach on grids that uses a refinement strategy based on spatial neighborhoods, akin to the neighborhood refinement described in Section 3.4.2. Schmitzer (2015) uses a multiscale approach to develop a modified auction algorithm with guaranteed worst case complexity and optimal solutions. We use data structures that enable us to quickly construct neighborhoods, even for point clouds that live in high-dimensions, but have low-intrinsic dimension; we also exploit these structures to not compute all the pairwise possible costs, as well as the candidate neighborhoods in our propagation steps. This can result in substantial savings in the scaling of the algorithm, from $|\mathbf{X}|^3$ to just $|\mathbf{X}|\log|\mathbf{X}|$. We are therefore able to scale to larger problem, solving on a laptop problems an order of magnitude larger than those in Oberman and Ruan (2015) for example, and showing linear scaling on a large range of scales. Schmitzer (2015) use the c–cyclical monotonicity property of optimal transport plans (Villani, 2009, Chapter 5) to construct "shielding neighborhoods" that permit to exclude paths from further consideration. The idea of shielding neighborhoods is combined with a multiscale strategy that permits to quickly refine initial neighborhood estimates.

Finally, in our work we emphasize that the multiscale construction is not only motivated by its computational advantage, but also as a way of revealing possibly important features of the optimal transportation map. As we show in Section 5, features collected from the multiscale optimal transportation maps leads to improved predictors for brain conditions. More generally, we expect multiscale properties of optimal transportation maps to be useful

in a variety of learning tasks; the connections between learning and optimal transportation are still a very open field, to be explored and exploited.

3. Multiscale Optimal Transport

Solving the optimal transport problem for two point sets $\mathbf{X}$ and $\mathbf{Y}$ directly requires $|\mathbf{X}||\mathbf{Y}|$ variables, or paths to consider. In other words, the number of paths along which mass can be transported grows quadratically in the number of points and quickly yields exceedingly large problems. The basic premise of the multiscale strategy is to solve a sequence of transport problems based on increasingly accurate approximations of the source and target point set. The multiscale strategy helps to reduce the problem size at each scale by using the solution from the previous scale to inform which paths to include in the optimization at the next finer scale. Additionally, the solution at the previous scale helps to find a good initialization for the current scale which results in fewer iterations to solve the reduced size linear program.

The multiscale algorithm (see Algorithm 1 and Figure 2 for a visual illustration) comprises of three key elements:

(I) A way of ***coarsening*** *the sets* of source points $\mathbf{X}$ and measure μ in a multiscale fashion, yielding a chain

$$(\mathbf{X},\mu) =: (\mathbf{X}_J,\mu_J) \to (\mathbf{X}_{J-1},\mu_{J-1}) \to \cdots \to (\mathbf{X}_j,\mu_j) \to \cdots \to (\mathbf{X}_0,\mu_0) \qquad (4)$$

connecting the scales from fine to coarse, with $\mathbf{X}_j$ of decreasing cardinality as the scale decreases, and the discrete measure μ_j "representing" a coarsification of μ at scale j, with $\mathrm{supp}(\mu_j) = \mathbf{X}_j$ (the support of μ_j is the set of points with positive measure). Similarly for $\mathbf{Y}$ and ν we obtain the chain

$$(\mathbf{Y},\mu) =: (\mathbf{Y}_J,\nu_J) \to (\mathbf{Y}_{J-1},\nu_{J-1}) \to \cdots \to (\mathbf{Y}_j,\nu_j) \to \cdots \to (\mathbf{Y}_0,\nu_0) \qquad (5)$$

This coarsening step is described in Section 3.1 and the resulting multiscale family of transport problems is discussed in Section 3.2.

(II) A way of ***propagating*** *a coupling* π_j solving the transport problem $\mu_j \rightarrowtail \nu_j$ at scale j to a coupling π_{j+1} at scale $j+1$. This is described in Section 3.3.

(III) A way of ***refining*** *the propagated solution* to the optimal coupling at scale j. This is described in Section 3.4.

3.1 The Coarsening Step: Multiscale Approximations to X, μ and c

To derive approximation bounds for the error of the multiscale transport problem at each scale we rely on the notion of a regular family of multiscale partitions formally described in Section 3.1.1. The multiscale partition is used to define approximations to μ and $\mathtt{c}$ at all scales. An integral part of the definitions is that the constructions can be interpreted as a tree, with all nodes at a fixed height corresponding to one scale of the multiscale partitioning.

Algorithm 1: Multiscale Discrete Optimal Transport

Input: Two discrete measures μ and ν, and a stopping scale $J_0 \leq J$
Output: Multiscale family of transport plans $(\pi_j : \mu_j \rightarrowtail \nu_j)_{j=0}^{J_0}$
Construct multiscale structures $\{\{\mathbf{X}_j, \mu_j)\}_{j=0}^{J}$ and $\{\{\mathbf{Y}_j, \nu_j)\}_{j=0}^{J}$.
Let $\tilde{\pi}_0$ be an arbitrary coupling $\mu_0 \rightarrowtail \nu_0$.
for $j = 0 \dots J_0 - 1$ **do**
  Refine initial guess $\tilde{\pi}_j$ to the optimal coupling $\pi_j : \mu_j \rightarrowtail \nu_j$.
  Propagate π_j from scale j to scale $j+1$, obtaining a coupling $\tilde{\pi}_{j+1}$

We start with some notation needed for the definition of the multiscale partitions. Let $(\mathbf{X}, \rho, \mu)$ be a measure metric space with metric ρ and finite measure μ. Without loss of generality assume that $\mu(\mathbf{X}) = 1$. The metric ball of center z and radius r is $B_z(r) = \{x \in \mathbf{X} : \rho(x, z) < r\}$. We say that $\mathbf{X}$ has doubling dimension d if every ball $B_z(r)$ can be covered by at most 2^d balls of radius $r/2$ (Assouad, 1983). Furthermore, a space has a doubling measure if $\mu(B_z(r)) \asymp r^d$, i.e. if there exist a constant c_1 such that for every $z \in \mathbf{X}$ and $r > 0$ we have $c_1^{-1} r^d \leq \mu(B_z(r)) \leq c_1 r^d$. Here and in what follows, we say that $f \asymp g$ if there are two constants $c_1, c_2 > 0$ such that for every z in the domain of both functions f, g we have $c_1 f(z) \leq g(z) \leq c_2 f(z)$ (and therefore a similar set of inequalities holds with the roles of f and g swapped), and we say that f and g have the same order of magnitude. Having a doubling measure implies having a doubling metric, and up to changing the metric to an equivalent one, one may choose the same d in the doubling condition for the metric and in that for the measure: we assume this has been done from now on. This family of spaces is rather general, it includes regular domains in $\mathbb{R}^D$, as well as smooth compact manifolds $\mathcal{M}$ endowed with volume measures.

3.1.1 Multiscale Approximations to $\mathbf{X}$ and μ

A regular family of multiscale partitions, with scaling parameter $\theta > 1$, is a family of sets $\{\{C_{j,k}\}_{k=1}^{K_j}\}_{j=0}^{J}$, where j denotes the scale and k indexes the sets at scale j, such that:

(i) the sets $\{C_{j,k}\}_{k=1}^{K_j}$ form a partition of $\mathbf{X}$, i.e. they are disjoint and $\cup_{k=1}^{K_j} C_{j,k} = \mathbf{X}$;

(ii) either $C_{j+1,k'}$ does not intersect a $C_{j,k}$, or it is completely contained in it;

(iii) there exists a constant $A > 0$ such that for all j, k we have the diameter $\mathrm{diam}(C_{j,k}) \leq A\theta^{-j}$;

(iv) each $C_{j,k}$ contains a "center" point $c_{j,k}$ such that $B_{c_{j,k}}(\theta^{-j}) \subseteq C_{j,k}$.

To ease the notation we will fix $\theta = 2$ in what follows, but the constructions and results hold, mutatis mutandis, for general $\theta > 1$. The properties (i) and (ii) above imply that there exists a tree $\mathcal{T}$, with nodes at scale j (i.e. at distance j from the root) in bijection with $\{C_{j,k}\}_{k=1}^{K_j}$, such that $(j+1, k')$ is a child of (j, k) if and only if $C_{j+1,k'}$ is contained in $C_{j,k}$. Moreover properties (iii) and (iv), together with the properties of spaces with a doubling measure, imply that $\mu(C_{j,k}) \asymp 2^{-jd}$ and $K_j \asymp 2^{jd}$. These partitions are classical in harmonic analysis, mimicking dyadic cubes in Euclidean space, and they have recently

been used to construct multiscale decompositions of data sets in high-dimensions (Little et al., 2012; Allard et al., 2012; Iwen and Maggioni, 2013; Chen et al., 2012). We say that $C_{j+1,k'}$, or even $(j+1, k')$, is a child of $C_{j,k}$ (respectively, of (j,k)) if $C_{j+1,k'} \subseteq C_{j,k}$, and that such $C_{j,k}$ (respectively (j,k)) is a parent of $C_{j+1,k'}$ (resp. $(j+1, k')$).

Given two discrete sets $\mathbf{X}$ and $\mathbf{Y}$ in a doubling metric space of homogeneous type with dimension d, we construct the corresponding families of multiscale partitions $\{\{C_{j,k}^{\mathbf{X}}\}_{k=1}^{K_j^{\mathbf{X}}}\}_{j=0}^{J}$ and $\{\{C_{j,k}^{\mathbf{Y}}\}_{k=1}^{K_j^{\mathbf{Y}}}\}_{j=0}^{J}$ (we assume the same range of scales to keep the notation simple). The construction of the multiscale approximations is illustrated in Figure 2.

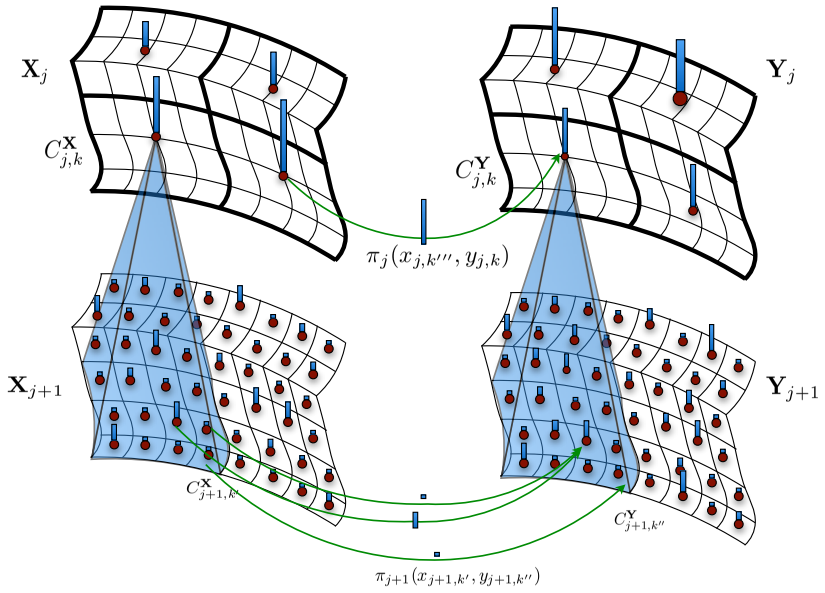


Figure 2: An illustration of the multiscale framework. The coarsening step constructs from a set of weighted points $\mathbf{X}_{j+1}$ at scale $j+1$ a smaller set of weighted points at scale j (blue bars). An equivalent coarsening is performed on the target point set $\mathbf{Y}_{j+1}$ Neighboring points at scale $j+1$ are combined into a single representative at scale j with mass equivalent to the weights of the combined points. The optimal transport plan (green arrows) is solved at the coarser scale j, then propagated to scale $j+1$ and refined.

The space $\mathbf{X}_j$ will be the partition at scale j, namely the set $\{C_{j,k}^{\mathbf{X}}\}_{k=1}^{K_j^{\mathbf{X}}}$, of cardinality $K_j^{\mathbf{X}}$. The measures μ and ν may be coarsened in the natural way, by letting μ_j be defined

recursively on $\mathbf{X}_j$ by

$$\mu_j(C^{\mathbf{X}}_{j,k}) = \sum_{(j+1,k') \text{ child of } (j,k)} \mu_{j+1}(C^{\mathbf{X}}_{j+1,k'}) \,, \tag{6}$$

and similarly for ν_j. These are in fact projections of these measures, and may also be interpreted as conditional expectations with respect to the σ-algebra generated by the multiscale partitions. We can associate a point $\overline{c}^{\mathbf{X}}_{j,k}$ to each $C^{\mathbf{X}}_{j,k}$ in various ways, by "averaging" the points in $C^{\mathbf{X}}_{j+1,k'}$, for $(j+1,k')$ a child of (j,k). At the finest scale we may let $\overline{c}^{\mathbf{X}}_{J,k} = c^{\mathbf{X}}_{J,k}$ (this being a "center" for $C_{J,k}$ as in item (iv) in the definition of multiscale partitions), and then recursively we defined the coarser centers step from scale $j+1$ to scale j in one of the following ways:

(i) If the metric space is also a vector space a natural definition of $x_{j,k} := \overline{c}^{\mathbf{X}}_{j,k}$ is a weighted average of the $\overline{c}^{\mathbf{X}}_{j+1,k'}$ corresponding to children:

$$\overline{c}^{\mathbf{X}}_{j,k} = \sum_{(j+1,k') \text{ child of } (j,k)} \mu_{j+1}(\{\overline{c}^{\mathbf{X}}_{j+1,k'}\}) \overline{c}^{\mathbf{X}}_{j+1,k'} \,.$$

(ii) In general we can define $x_{j,k} := \overline{c}^{\mathbf{X}}_{j,k}$ as the point

$$\overline{c}^{\mathbf{X}}_{j,k} = \mathrm{argmin}_{c \in \mathbf{X}} \sum_{(j+1,k') \text{ child of } (j,k)} \rho^p(c, \overline{c}^{\mathbf{X}}_{j+1,k'}) \,,$$

for some $p \geq 1$, typically $p = 1$ (median) or $p = 2$ (Fréchet mean).

Of course similar constructions apply to the space $\mathbf{Y}_j$, yielding points $y_{j,k} := \overline{c}^{\mathbf{Y}}_{j,k}$. We discuss algorithms for these constructions in Section 4.1.1.

3.1.2 Coarsening the cost function $\mathtt{c}$

The multiscale partition provides several ways to coarsen the cost function: for every $x_{j,k}$ and $y_{j,k'}$ we consider

($\mathtt{c}$-i) the pointwise value

$$\mathtt{c}_j(c^{\mathbf{X}}_{j,k}, c^{\mathbf{Y}}_{j,k'}) := \mathtt{c}(x^{\mathbf{X}}_{j,k}, y^{\mathbf{Y}}_{j,k'}) \,, \tag{7}$$

where $x_{j,k}$ and $y_{j,k'}$ are defined in any of the ways above;

($\mathtt{c}$-ii) the local average

$$\mathtt{c}_j(c^{\mathbf{X}}_{j,k}, c^{\mathbf{Y}}_{j,k'}) := \operatorname*{argmin}_{\alpha} \sum_{x \in C^{\mathbf{X}}_{j,k}, y \in C^{\mathbf{Y}}_{j,k'}} (\alpha - \mathtt{c}(x,y))^2 = \frac{\sum_{x \in C^{\mathbf{X}}_{j,k}, y \in C^{\mathbf{Y}}_{j,k'}} \mathtt{c}(x,y)}{|C^{\mathbf{X}}_{j,k}||C^{\mathbf{Y}}_{j,k'}|} \,; \tag{8}$$

($\mathtt{c}$-iii) the local weighted average

$$\begin{aligned} \mathtt{c}_j(c^{\mathbf{X}}_{j,k}, c^{\mathbf{Y}}_{j,k'}) &:= \operatorname*{argmin}_{\alpha} \sum_{x \in C^{\mathbf{X}}_{j,k}, y \in C^{\mathbf{Y}}_{j,k'}} (\alpha - \mathtt{c}(x,y))^2 \, \pi^*_{j-1}(x_{j-1,k_1}, y_{j-1,k'_1}) \\ &= \frac{\sum_{x \in C^{\mathbf{X}}_{j,k}, y \in C^{\mathbf{Y}}_{j,k'}} \mathtt{c}(x,y) \pi^*_{j-1}(x_{j-1,k_1}, y_{j-1,k'_1})}{\sum_{x \in C^{\mathbf{X}}_{j,k}, y \in C^{\mathbf{Y}}_{j,k'}} \pi^*_{j-1}(x_{j-1,k_1}, y_{j-1,k'_1})} \,, \end{aligned}$$

where π^*_{j-1} is the optimal or approximate transportation plan at scale $j-1$, defined in (9); k_1 is the unique index for which $C^{\mathbf{X}}_{j,k} \subseteq C^{\mathbf{X}}_{j-1,k_1}$ and k'_1 is the unique index for which $C^{\mathbf{Y}}_{j,k} \subseteq C^{\mathbf{Y}}_{j-1,k'_1}$.

3.2 Multiscale Family of Optimal Transport Problems

With the definitions of the multiscale family of coarser spaces $\mathbf{X}_j$ and $\mathbf{Y}_j$, corresponding measures μ_j and ν_j, and corresponding cost $\mathtt{c}_j$, we may consider, for each scale j, the following optimal transport problem:

$$\pi^*_j := \operatorname*{argmin}_{\pi} \sum_{\substack{k=1,\dots,K^{\mathbf{X}}_j \\ k'=1,\dots,K^{\mathbf{Y}}_j}} \mathtt{c}_j(x_{j,k}, y_{j,k'}) \pi(x_{j,k}, y_{j,k'}) \text{ s.t. } \begin{cases} \sum_{k'} \pi(x_{j,k}, y_{j,k'}) = \mu_j(\{x_{j,k}\}) & \forall k \in K^{\mathbf{X}}_j \\ \sum_{k} \pi(x_{j,k}, y_{j,k'}) = \nu_j(\{y_{j,k'}\}) & \forall k' \in K^{\mathbf{Y}}_j \\ \pi(x_{j,k}, y_{j,k'}) \geq 0 \end{cases} \tag{9}$$

The problems in this family are related to each other, and to the optimal transportation problem in the original spaces. We define the cost of a coupling as

$$\mathtt{cost}(\pi_j) = \sum_{\substack{k=1,\dots,K^{\mathbf{X}}_j \\ k'=1,\dots,K^{\mathbf{Y}}_j}} \mathtt{c}_j(x_{j,k}, y_{j,k'}) \pi_j(x_{j,k}, y_{j,k'}) \,. \tag{10}$$

The cost of the optimal coupling π^*_j at scale j is provably an approximation to the cost of the optimal coupling π^* (which is equal to π^*_J):

Proposition 1 *Let π^* be the optimal coupling, i.e. the solution to (3), and π^*_j the optimal coupling at scale j, i.e. the solution to (9). Define*

$$E_j(\pi^*) := \sum_{\substack{k=1,\dots,K^{\mathbf{X}}_j \\ k'=1,\dots,K^{\mathbf{Y}}_j}} \sum_{\substack{x \in C^{\mathbf{X}}_{j,k} \\ y \in C^{\mathbf{Y}}_{j,k'}}} \left(\mathtt{c}_j(x_{j,k}, y_{j,k'}) - \mathtt{c}(x,y)\right) \pi^*(x,y) \,. \tag{11}$$

Then

$$\mathtt{cost}(\pi^*_j) \leq \mathtt{cost}(\pi^*) + E_j(\pi^*) \,, \tag{12}$$

and if $\mathtt{c}_j = \mathtt{c}$ and $\mathtt{c}$ is Lipschitz with constant $||\nabla \mathtt{c}||_\infty$, we have

$$\mathtt{cost}(\pi^*_j) \leq \mathtt{cost}(\pi^*) + 2^{-j} A ||\nabla \mathtt{c}||_\infty \,, \tag{13}$$

where A is such that $\max_{k,k'} \{\mathrm{diam}(C^{\mathbf{X}}_{j,k}), \mathrm{diam}(C^{\mathbf{Y}}_{j,k'})\} \leq A \cdot 2^{-j}$.

Proof Consider the coupling π_j induced at scale j by the optimal coupling π^*, defined by

$$\pi_j(x_{j,k}, y_{j,k'}) = \sum_{x \in C^{\mathbf{X}}_{j,k}, y \in C^{\mathbf{Y}}_{j,k'}} \pi^*(x,y) \,.$$

First of all, since $\{C^{\mathbf{X}}_{j,k}\}_k$ and $\{C^{\mathbf{Y}}_{j,k'}\}_{k'}$ are partitions, it is immediately verified that π_j is a coupling. Secondly, observe that

$$\begin{aligned}
\mathtt{cost}(\pi_j) &= \sum_{\substack{k=1,\dots,K^{\mathbf{X}}_j \\ k'=1,\dots,K^{\mathbf{Y}}_j}} \mathtt{c}_j(x_{j,k}, y_{j,k'})\pi_j(x_{j,k}, y_{j,k'}) = \sum_{\substack{k=1,\dots,K^{\mathbf{X}}_j \\ k'=1,\dots,K^{\mathbf{Y}}_j}} \sum_{\substack{x\in C^{\mathbf{X}}_{j,k} \\ y\in C^{\mathbf{Y}}_{j,k'}}} \mathtt{c}_j(x_{j,k}, y_{j,k'})\pi^*(x,y) \\
&= \sum_{\substack{x\in\mathbf{X} \\ y\in\mathbf{Y}}} \mathtt{c}(x,y)\pi^*(x,y) + \sum_{\substack{k=1,\dots,K^{\mathbf{X}}_j \\ k'=1,\dots,K^{\mathbf{Y}}_j}} \sum_{\substack{x\in C^{\mathbf{X}}_{j,k} \\ y\in C^{\mathbf{Y}}_{j,k'}}} \big(\mathtt{c}_j(x_{j,k}, y_{j,k'}) - \mathtt{c}(x,y)\big)\,\pi^*(x,y) \\
&= \mathtt{cost}(\pi^*) + \underbrace{\sum_{\substack{k=1,\dots,K^{\mathbf{X}}_j \\ k'=1,\dots,K^{\mathbf{Y}}_j}} \sum_{\substack{x\in C^{\mathbf{X}}_{j,k} \\ y\in C^{\mathbf{Y}}_{j,k'}}} \big(\mathtt{c}_j(x_{j,k}, y_{j,k'}) - \mathtt{c}(x,y)\big)\,\pi^*(x,y)}_{=:E_j(\pi^*)} .
\end{aligned}$$

Since $\mathtt{cost}(\pi^*_j) \le \mathtt{cost}(\pi_j)$ (since π^*_j is optimal), we obtain (12). When $\mathtt{c}_j = \mathtt{c}$ and $\mathtt{c}$ is Lipschitz with constant $||\nabla \mathtt{c}||_\infty$, we have

$$\begin{aligned}
E_j(\pi^*) &\le \sum_{\substack{k\,:\,x\in C^{\mathbf{X}}_{j,k} \\ k'\,:\,y\in C^{\mathbf{Y}}_{j,k'}}} \sum_{\substack{k=1,\dots,K^{\mathbf{X}}_j \\ k'=1,\dots,K^{\mathbf{Y}}_j}} ||\nabla \mathtt{c}||_\infty \cdot ||(x_{j,k}, y_{j,k'}) - (x,y)||\pi^*(x,y) \\
&\le \sum_{\substack{k\,:\,x\in C^{\mathbf{X}}_{j,k} \\ k'\,:\,y\in C^{\mathbf{Y}}_{j,k'}}} \sum_{\substack{k=1,\dots,K^{\mathbf{X}}_j \\ k'=1,\dots,K^{\mathbf{Y}}_j}} ||\nabla \mathtt{c}||_\infty 2^{-j} A\pi^*(x,y) \\
&\le 2^{-j} A||\nabla \mathtt{c}||_\infty ,
\end{aligned}$$

with A as in the claim. ∎

In the discrete, finite case that we are considering, $\mu_j \to \mu$ and $\nu_j \to \nu$ trivially since $\mu_J = \mu$ and $\nu_J = \nu$ by construction. If μ and ν were continuous, and at least when $\mathtt{c}(x,y) = \rho(x,y)^p$ for some $p \ge 1$, then if μ and ν have finite p-moment (i.e. $\int_{\mathbf{X}} \rho(x,x_0)^p d\mu$ and similarly for ν), we would obtain convergence of a subsequence of $\mathtt{c}(\pi^*_j)$ to $\mathtt{c}(\pi^*)$ by general results (e.g. as a simple consequence of Lemma 4.4. in (Villani, 2009)).

Note that the approximations do not guarantee that the transport plans are close in any other sense but their cost. Consider the arrangement in Figure 3, the transport plans are ϵ-close in cost but the distances between the target locations of the sources are far no matter how small ϵ gets.

3.3 Propagation Strategies

The approximation bounds in Section 3.2 show that the optimal solution π_j at scale j is $|E_{j+1} - E_j|$ close to optimal solution π^*_{j+1} at scale $j+1$. This suggests that the solution at scale j can provide a reasonably close to optimal initialization for scale $j+1$.

As proposed by Glimm and Henscheid (2013) the solution at a given scale can be interpolated at the next scale (or finer discretization). The most direct approach to initialize the

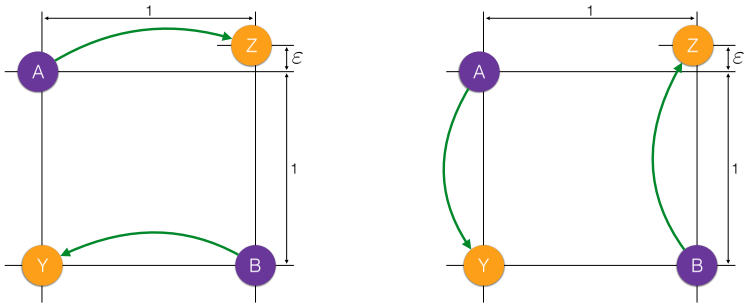


Figure 3: An example that illustrates that closeness in cost does not indicate closeness of the transport plan. The transport plans (green arrows) between the sources A and B (purple) and the targets Y and Z (orange) in (a) and (b) are ϵ-close but their respective target locations are very far.

transport problem at scale $j+1$ given the solution π^*_j at scale j is to distribute the mass $\pi^*_j(x_k, y_{k'})$ equally to all combinations of paths between children(x_k) and children$(y_{k'})$.

This propagation strategy results in a reduction in the number of iterations required to find an optimal solution at the subsequent scale. This warm-starting alone is often not sufficient, however. At the finest scale the problems still requires the solution of a problem of size $O(n^2)$. This quickly reaches memory constraints with $\Omega(10^4)$ points, and a single iteration of Newton's method or a pivot step of a linear program becomes prohibitively slow. Thus, we consider reducing the number of variables, which substantially speeds up the algorithm, albeit we may lose guarantees on its computational complexity and/or its ability to achieve arbitrary accuracy, so that only numerical experiments will support our constructions. These reductions are achieved by considering only a subset $\mathbf{R}_{j+1}$ of all possible paths $\mathbf{A}_{j+1}$ at scale $j+1$.

To distinguish the optimal solution on the reduced set of paths at scale j from the optimal solution over all paths we introduce some notation. Let $\mathbf{A}_{j+1}$ be the set of all possible paths between sources and targets at scale $j+1$. Let $\mathbf{R}_{j+1} \subseteq \mathbf{A}_{j+1}$ be the set of paths propagated from the previous solution (e.g. children of mass-bearing paths found at scale j). Let $\pi^*_{j+1}|_{\mathbf{P}}$ be the optimal solution to the transport problem restricted to a set of paths $\mathbf{P} \subset \mathbf{A}_{j+1}$. With this notation, $\pi^*_{j+1}|_{\mathbf{A}_{j+1}} = \pi^*_{j+1}$. The optimal coupling $\pi^*_{j+1}|_{\mathbf{R}_{j+1}}$ on the reduced set of paths problem does not need to match the optimal coupling π^*_{j+1} on all paths. However $\pi^*_{j+1}|_{\mathbf{R}_{j+1}}$ does provide a starting point for further refinements discussed in Section 3.4.

3.3.1 Simple Propagation

The most direct approach to reduce the number of paths considered at subsequent scales is to include only paths at scale $j+1$ whose endpoints are children of endpoints of mass-

bearing paths at scale j. The optimal solution at scale j has exactly $K_j^{\mathbf{X}} + K_j^{\mathbf{Y}} - 1$ paths with non-zero weight. Thus, the number of paths at scale $j+1$ reduces to $O(C^2(K_j^{\mathbf{X}}+K_j^{\mathbf{Y}}))$, where C is the maximal number of children of any node at scale j. In particular, $C \asymp 2^d$ for a doubling space of dimension d. This reduces the number of variables from "quadratic", $O(K_j^{\mathbf{X}} K_j^{\mathbf{Y}})$, to linear, $O(K_j^{\mathbf{X}} + K_j^{\mathbf{Y}})$. This propagation strategy by itself, however, often leads to a dramatic loss of accuracy in both the cost the transportation plan, and the transportation plan itself.

3.3.2 Capacity Constraint Propagation

This propagation strategy solves a modified minimum flow problem at scale j in order to include additional paths at scale $j+1$ that are likely to be included in the optimal solution π^*_{j+1}. This is achieved by adding a capacity constraint to the mass bearing paths at scale j in the optimal coupling $\pi^*_j|_{\mathbf{R}_j}$: The amount of mass of a mass bearing path $\pi^*_j|_{\mathbf{R}_j}(x_{j,k}, y_{j,k'})$ is constrained to $\lambda \min\left(\mu_j(x_{j,k}), \nu_j(y_{j,k'})\right)$ with λ random uniform on $[0.1, 0.9]$. The randomness is introduced to avoid degenerate constraints. The solution of this modified minimum–flow problem forces the inclusion of n_c additional paths, where n_c is the number of constraints added. There are various options for adding capacity constraints, we propose to constrain all mass bearing paths of the optimal solution at scale $\pi^*_j|_{\mathbf{R}_j}$. The capacity constrained problem thus results in a solution with twice the number of paths as in the coupling $\pi^*_j|_{\mathbf{R}_j}$. The solution of the capacity constrained minimum flow problem is propagated as before to the next scale.

To increase the likelihood of including paths required to find an optimal solution at the next scale, the capacity constrained procedure can be iterated multiple times. Each time the mass bearing paths in the modified solution are constrained and a new solution is computed. Each iteration doubles the number of mass bearing paths and the number of iterations controls how many paths are propagated to the next scale. Thus, the capacity constraint propagation strategy bounds the number of paths considered in the linear program. The optimal transport plan on a source set $\mathbf{X}$ and $\mathbf{Y}$ results in a linear program with $|\mathbf{X}| + |\mathbf{Y}|$ constraints and $|\mathbf{X}||\mathbf{Y}|$ variables and the optimal transport plan has $|\mathbf{X}| + |\mathbf{Y}| - 1$ mass bearing paths. It follows that the capacity constraint propagation strategy considers linear programs with at most $O\left(2^i(|\mathbf{X}| + |\mathbf{Y}|)\right)$ constraints, where i is the number of iterations of the capacity propagation scheme. This results in a significant reduction in problem size, since at each scale we only consider a number of paths scaling as $O(|\mathbf{X}| + |\mathbf{Y}|)$ instead of $O(|\mathbf{X}||\mathbf{Y}|)$.

3.4 Refinement Strategies

Solving the reduced transport problem at scale $j+1$, propagated from scale j, can not guarantee an optimal solution at scale $j+1$. Propagating a sub-optimal solution further may lead to an accumulation of errors. This section describes strategies to refine the reduced transport problem to find closer approximations or even optimal transport plans at each scale. These refinement strategies are essentially batch column generation methods (Desaulniers et al., 2002), that take advantage of the multiscale structure.

3.4.1 Potential Refinement

This refinement strategy exploits the potential functions, or dual solution, to determine additional paths to include given the currently optimal solution on the reduced set of paths from the propagation. The dual formulation of the optimal transport can be written as:

$$\max_{\phi,\psi} \sum_{i=1,..,n} \mu(\{x_i\})\phi(x_i) - \sum_{j=1,..,m} \nu(\{y_j\})\psi(y_j) \quad \text{s.t.} \quad \phi(x_i) - \psi(y_j) \leq \mathsf{c}(x_i, y_j)\,. \tag{14}$$

The functions ϕ and ψ are called dual variables or potential functions. From the dual formulation it follows that at an optimal solution the *reduced cost* $\mathsf{c}(x,y) - \phi(x) + \psi(y)$ is larger or equal to zero. This also follows from the Kantorovich duality of optimal transport (Villani, 2009, Chapter 5).

The potential refinement strategy uses the potential functions ϕ and ψ from the solution of the reduced problem to determine which additional paths to include. If the solution on the reduced problem is not optimal on all paths, then there exist paths with negative reduced cost. Thus, we check the reduced cost between all paths and include the ones negative reduced cost. A direct implementation of this strategy would require to check all possible paths between the source and target points. To avoid checking all pairwise paths between source and target point sets at the current scale, we introduce a branch and bound procedure on the multiscale structure that efficiently determines all paths with negative reduced cost.

Let $\phi^*_j|_{\mathbf{P}}$ and $\psi^*_j|_{\mathbf{P}}$ the dual variables, i.e., the potential functions, at the optimal solution $\pi^*_j|_{\mathbf{P}}$ on the set of paths $\mathbf{P}$ for the source and target points, respectively. Define $\mathbf{V}_j(\pi^*_j|_{\mathbf{P}})$ as the set of paths with non-positive reduced cost with respect to $\phi^*_j|_{\mathbf{P}}$ and $\psi^*_j|_{\mathbf{P}}$

$$\mathbf{V}_j(\pi^*_j|_{\mathbf{P}}) = \left\{\pi_j(x_k, y_{k'}) \in \mathbf{A}_j : \mathsf{c}(x_k, y_{k'}) - \phi^*_j|_{\mathbf{P}}(x_k) - \psi^*_j|_{\mathbf{P}}(y_{k'}) \leq 0\right\}\,.$$

The *potential* refinement strategy now refines the propagated solution $\pi^*_j|_{\mathbf{R}_j}$ by including all negative reduced cost paths $\mathbf{Q}^0_j = \mathbf{V}_j(\pi^*_j|_{\mathbf{R}_j})$. Let $\pi^*_j|_{\mathbf{Q}^0_j}$ be the associated optimal transport. This new solution changes the potential functions which in turn may require to include additional paths. Thus the potential refinement strategy can be iterated with $\mathbf{Q}^i_j = \mathbf{V}_j(\pi^*_j|_{\mathbf{Q}^{i-1}_j})$ leading to monotonically decreasing optimal transport plans $\pi^*_j|_{Q^i_j}$. Since a solution is optimal if and only if all reduced cost are ≥ 0 this iterative strategy converges to the optimal solution.

The set of paths with negative reduced cost given $\phi^*_j|_{\mathbf{P}}$ and $\psi^*_j|_{\mathbf{P}}$ are determined by descending the tree and excluding nodes that can not contain any negative reduced cost paths. This requires bounds on the potential functions for any node at scale smaller than j. The bound is achieved by storing at each target node of the multiscale decomposition the maximal $\psi^*_j|_{\mathbf{P}}$ of any of its descendants at scale j. Now for each source node x_k the target multiscale decomposition is descended, but only towards nodes at which a potential negative reduced cost can exist.

Depending on the properties of $\phi^*_j|_{\mathbf{P}}$ and $\psi^*_j|_{\mathbf{P}}$, this refinement strategy may reduce the number of required cost function evaluations drastically. In empirical experiments the total number of paths considered in the iterative potential refinement was typically reduced to $O(2^d(K_j^{\mathbf{X}} + K_i^{\mathbf{X}}))$, with d being the doubling dimension of the data set.

The potential refinement strategy is also employed by Glimm and Henscheid (2013) without the branch and bound procedure. The shielding neighborhoods proposed by Schmitzer (2015) similarly uses the potentials to define which variables are needed to define a sparse optimal transport problem and also suggests to iterate the neighborhood shields in order to arrive at an optimal solution.

3.4.2 Neighborhood Refinement

This section presents a refinement strategy that takes advantage of the geometry of the data. The approach is based on the heuristic that most paths at the next scale are suboptimal due to boundary effects when moving from one scale to the next, induced by the sharp partitioning of the space at scale j. Such artifacts from the multiscale structure are mitigated by including paths between spatial neighbors in the source and target locations of the optimal solution on the propagated paths. This refinement strategy is also employed by Oberman and Ruan (2015).

Let $\mathbf{N}_j(\pi_j, r)$ be the set of paths such that the source and target of any path are within radius r of the source and target of a path with non-zero mass transfer in π_j. The neighborhood refinement strategy is to expand the reduced set of paths using the union of paths in the current reduced set and its neighbors:

$$\mathbf{E}_j = \mathbf{R}_j \cup \mathbf{N}_j\left(\pi_j^*|_{\mathbf{R}_j}, r\right)$$

When moving from one scale to the next the cost of any path can change at most two times the radius r of the parent node which suggests to set the radius of the neighborhood in consideration as two times the node radius.

This heuristics does not guarantee an optimal solution, but does reduce the number of paths to consider at scale j to $O(q_r^2(K_j^{\mathbf{X}} + K_j^{\mathbf{Y}}))$ with q_r being the number of neighbors within radius $2^{-j}r$, for a doubling space with dimension d, $q_r \asymp r^d$.

The neighborhood strategy requires to efficiently compute the set of points within a ball of a given radius and location. Depending on the multiscale structure there are different ways to compute the set of neighbors. A generic approach that does not depend on any specific multiscale structure is to use a branch and bound strategy. For this approach, each node requires an upper bound $u(c_i)$ on the maximal distance from the representative of node c_i to any of its descendants. Using these upper bounds the tree is searched, starting from the root node, while excluding any child nodes for which $\mathtt{c}(x, c_i) - u(c_i) > r$ from further consideration. For multiscale structures such as regular grids more efficient direct computations are possible. For this paper we implemented the generic branch and bound strategy that works with any multiscale structure.

3.5 Remark on Errors

The error induced by this multiscale framework stems from two sources. First, if $J_0 < J$, i.e., the optimal transport problem is only solved up to scale J_0, the solution at scale J_0 has a bounded approximation error, as detailed in Section 3.2. By solving the transport problem only up to a certain scale permits to trade-off computation time versus precision with guaranteed approximation bounds. However, to speed up computation we rely on heuristics that, depending on the refinement strategy, yield solutions at each scale that might not be optimal. This second type error is difficult to quantify; however, for the potential refinement strategy that we introduce Section 3.4.1, an optimal solution can be guaranteed. The propagation and refinement strategies introduced in Sections 3.3 and 3.4 permit trade-offs between accuracy and computational cost.

4. Numerical Results and Comparisons

We tested the performance of the proposed multiscale transport framework with respect to computational speed as well as accuracy for different propagation and refinement strategies on various source and target point sets with different properties.

The multiscale approach is implemented in C++ with an R front end in the package *mop*[1]. For comparisons, we also implemented the Sinkhorn distance approach (Cuturi, 2013) in C++ with an R front end in the package *sinkhorn*[1]. Our C++ implementation uses the Eigen linear algebra library (Guennebaud et al., 2010), which resulted in faster runtimes than the MATLAB implementation by Cuturi (2013).

4.1 Implementation Details

The R package *mop* provides a flexible implementation of the multiscale framework and is adaptable to any multiscale decomposition that can be represented by a tree. The package permits to use different optimization algorithms to solve the individual transport problem. Currently the multiscale framework implementation supports the open source library GLPK (Makhorin, 2013) and the commercial packages MOSEK (Andersen and Andersen, 2013) and CPLEX (IBM, 2013) (both free for academic use). MOSEK and CPLEX support a specialized network simplex implementation that runs 10-100 times faster than a typical primal simplex implementation. Both the MOSEK and CPLEX network simplex run at comparable times, with CPLEX slightly faster in our experiments. Furthermore CPLEX supports starting from an advanced initial basis for the network simplex algorithm which improves the multiscale run-times significantly. Thus, the numerical test are all run using the CPLEX network simplex algorithm.

4.1.1 Algorithms for Constructing Multiscale Point Set Representations

Various approaches exist to build the multiscale structures described in Section 3.1, such as hierarchical clustering type algorithms (Ward Jr, 1963), or in low dimensions constructions such as quad and oct-trees (Finkel and Bentley, 1974; Jackins and Tanimoto, 1980) or kd-trees (Bentley, 1975) are feasible. Data structures developed for fast nearest neighbor queries, such as navigating nets (Krauthgamer and Lee, 2004) and cover trees (Beygelzimer et al., 2006) induce a hierarchical structure on the data sets with guarantees on partition size and geometric regularity of the elements of the partition at each scale, under rather general assumptions on the distribution of the points. The complexity of cover trees (Beygelzimer et al., 2006) is $O(C^d Dn \log n)$, for some constant C, where $n = |\mathbf{X}|$, d is the doubling dimension of $\mathbf{X}$, and D is the cost of computing a distance between a pair of points. Therefore the algorithm is practical only when the intrinsic dimension is small, in which case they are provably adaptive to such intrinsic dimension. The optimal transport approach

1. available on `https://bitbucket.org/suppechasper/optimaltransport`

does not rest on a specific multiscale structure and can be adapted to application-dependent considerations. However, the properties of the multiscale structure, i.e., depth and partition sizes, do affect run-time and approximation bounds.

In our experiments we use an iterative K-means strategy to recursively split the data into subsets. The tree is initialized using the mean of the complete data set as the root node. Then K-means is run resulting in K children. This procedure is recursively applied for each leaf node, in a breadth first fashion, until a desired number of of leaf nodes, a maximal leaf radius or the leaf node contains only a single point. For the examples shown we select $K = 2^d$ with d the dimensionality of the data set. For high-dimensional data K could be set to an estimate of the intrinsic dimensionality. Since in all experiments we are equipped with a metric structure we use the pointwise coarsening of the cost function as in Equation (7). The reported results include the computation time for building the hierarchical decomposition, which is, however, negligible compared to solving the transport problem at all scales.

4.1.2 Multiscale Transport Implementation for Point Sets

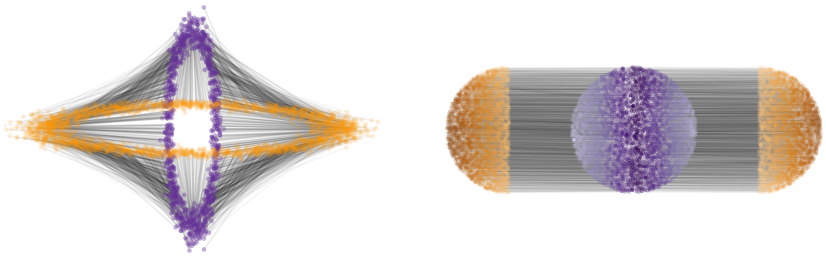
Algorithm 2 details the steps for computing multiscale optimal transport plans using the newtork simplex for solving the optimization problems at each scale and iterated K–means to construct the multiscale structures.

Algorithm 2: Point Set based Multiscale Optimal Transport Implementation

Input: Source point set $\mathbf{X} = \{x\}_{i=0}^{N}$.
Target point set $\mathbf{Y} = \{y\}_{i=0}^{M}$.
A propagation strategy p.
A refinement strategy r.
Output: Multiscale family of transport plans $(\pi_j : \mathbf{X}_j \rightarrowtail \mathbf{Y}_j)_{j=0}^{J}$
Construct multiscale point sets $\{\mathbf{X}_j\}_{j=0}^{N}$ and $\{\mathbf{Y}_j\}_{j=0}^{M}$ using iterated K–means.
if $N < M$ **then**
  Add scales $\{\mathbf{X}_j\}_{j=N+1}^{J}$ by repeating the last scale.
if $N > M$ **then**
  Add scales $\{\mathbf{Y}_j\}_{j=M+1}^{J}$ by repeating the last scale.
Set $J = \max(N, M)$
Form the measures μ_j and ν_j as in equation (6).
Compute optimal transport π_0 at coarsest scale with the network simplex.
for $j = 1 \dots J$ **do**
  Propagate the π_{j-1} from scale $j-1$ to scale j using the propagation strategy p, obtaining a set of paths $\mathbf{S}_j = p(\pi_{j-1}$ at scale j
  Use the network simplex algorithm to solve for optimal transport on the set of paths $\mathbf{S}_j$ yielding a coupling $\tilde{\pi}_j$
  Create the refined set of paths $\mathbf{R}_j = r(\tilde{\pi}_j)$ using the refinement strategy r.
  Use the network simplex algorithm to solve for optimal transport on the set of paths $\mathbf{R}_j$ yielding the optimal coupling π_j on the paths $\mathbf{R}_j$.

4.2 Propagation and Refinement Strategy Performance

Figure 5 illustrates the behaviour of the different propagation and refinement strategies on two examples: The ellipses example in Figure 1 and Caffarelli's smoothness counter example described in Villani (2009, Chapter 12) and illustrated in Figure 4. The ellipse example consists of two uniform samples (source and target data set) of size n from the unit circle with normal distributed noise added with zero mean and standard deviation 0.1. The source data sample is then scaled in the x-Axis by 1.3 and in the y-Axis by 0.9, while the target data set is scaled in the x-Axis by 0.9 and in the y-Axis by 1.1. Caffarelli's example consists of two uniform samples on $[-1, 1]^2$ of size n. Any points outside the unit circle are then discarded. Additionally, the target data sample is split along the x-Axis at 0 and shifted by $+2$ and -2 for points with positive and negative x-Axis values, respectively. We tested 7



(a) (b)

Figure 4: Optimal transport plans on the (a) ellipse and (b) Caffarelli data sets with 5000 points for the source and target point set each. The optimal transport plans are indicated by black lines with transparency indicating the amount of mass being transported from source to target.

different strategies of different combinations of propagation strategies from Section 3.3 with refinement strategies from Section 3.4:

1. (ICP) Iterated capacity propagation (Section 3.3.2) with 0 to 5 iterations with no refinements. Note iterated capacity with 0 iterations is equivalent to simple propagation (Section 3.3.1).
2. (NR) Simple propagation (Section 3.3.1) with neighborhood refinement (Section 3.4.2) with radius factor ranging from 0.5 to 2.5 in 0.5 increments.
3. (ICP + NR) Iterated capacity propagation with 1 to 5 iterations combined with neighborhood refinement with radius factor fixed to 1.
4. (CP + NR) A single iteration of capacity propagation combined with neighborhood refinement with radius factor from 0.5 to 2.5 with 0.5 increments.
5. (IPR) Simple propagation with iterated potential refinement (Section 3.4.1) with 1 to 5 iterations.
6. (ICP + PR) Iterated capacity propagation with 1 to 5 iterations combined with a single potential refinement step.
7. (CP + IPR) A single iteration of capacity propagation combined with 1 to 5 iterations of potential refinement.

Figure 5 shows that almost all strategies have less than one percent relative error. The exception is the simple propagation with no refinements applied (i.e. iterated capacity propagation with no iterations). The randomness of the iterated capacity constrained algorithm can results in worse results despite a larger number of iterations. However, it will always perform better than using the simple propagation strategy. The neighborhood refinement strategy improves the results significantly, but after a radius factor of one the improvements start to level out. The potential refinement strategy finds the optimal solution when iterated a few times. Combining the refinement strategy with capacity propagation reduces relative error and computation time. The computation time is reduced because the capacity propagation yields a closer initialization to the optimal solution. The combination of potential refinement strategy and capacity propagation has the additional benefit that the branch and bound strategy is more efficient since fewer comparisons need to be made when checking for paths and smaller linear programs have to be solved in each refinement step.

For very small relative error ($< 10^{-11}$) the optimal solution is sometimes not achieved.

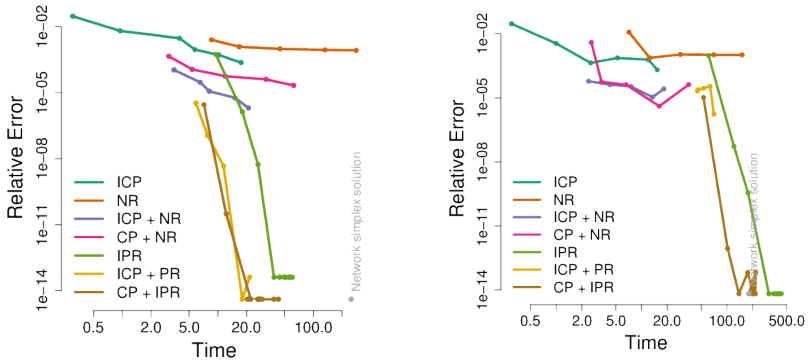


Figure 5: A comparison of different multiscale propagation and refinement strategies on the (a) ellipse and (b) Caffarelli data sets with 5000 points for the source and target point set each. For a description of the different strategies see text, the lines in the relative error graphs indicate increasing number of iterations or radius factor for the different strategies. Both the time and accuracy axes are in logarithmic scale. All strategies, except capacity propagation with 0 iterations (ICP) and neighborhood refinement (NR) with radius factor 0.5, find solution with relative error less than one percent. The capacity propagation strategies can find optimal solutions up to numerical precision.

This is due to tolerance settings in the network simplex algorithm which are on the order of 10^{-11}. Thus, depending how the network simplex approaches the optimum it might stop at different basic feasible solutions.

Figure 5 shows that the computation time and relative error for problem of equal sizes depends on the type of problem. An important aspect of the problem type is the ratio of

the transport distance to the diameters of the source and target point sets. To illustrate this effect we computed optimal transport plans for two data sets with source and target distributions uniform on $[0,1]^2$. In the first case the distributions are perfectly overlapping and in the second case the target distributions shifted by 2 units in the x direction. Figure 6 shows that for large ratios the relative error is typically much smaller. This is expected since variations in the transport plan only change the cost marginally. For small ratios, i.e. source and target distributions that are almost identical, a small variation in the transport plan leads to a large relative error. Another observation is the following: the potential strategy

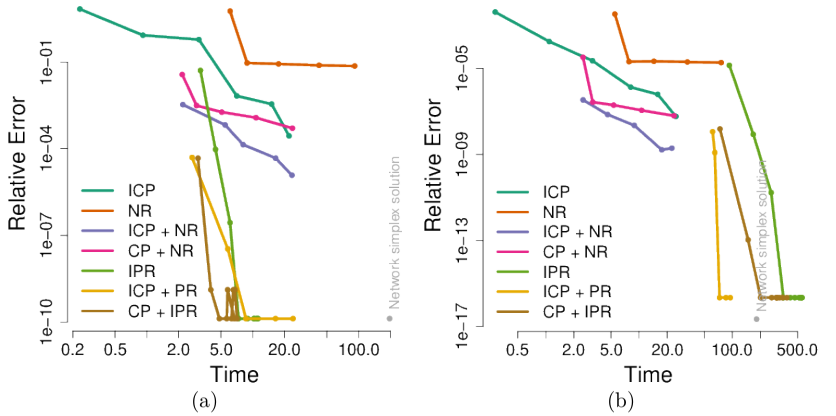


Figure 6: The effect of distance to support size ratio on computation time and relative error for the different multiscale propagation and refinement strategies source and target point sets of 5000 points each sampled uniformly from a square of side length one and ground truth transport distance (a) 0 and (b) 2. Both the time and accuracy axes are in logarithmic scale.

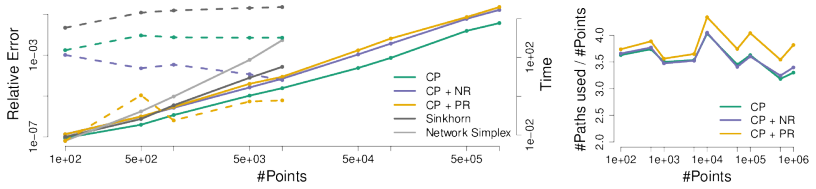
is less effective for transport problems where most mass is transported very far, relative to the distances within source and target point set. This is because a small change in the length of a path can include many possible source and target locations and the transport polytope has many suboptimal solutions with similar cost. If on the other hand most mass is transported on the order of the nearest neighbor distances within the source and target point set, there are many fewer possible paths within a small change in path length and the transport polytope is "steep" in the direction of the cost function.

4.3 Comparison to Network Simplex and Sinkhorn Transport

In this section we compare the CPLEX network simplex algorithm (IBM, 2013) and the Sinkhorn approach (Cuturi, 2013) to three different multiscale strategies:

1. (CP) Capacity propagation strategy using a single iteration with no further refinements.

2. (CP + NR) Capacity propagation combined with neighborhood refinement with radius factor fixed to one and a single capacity constraint iteration.
3. (CP + PR) Capacity propagation combined with potential refinement with a single iteration each.



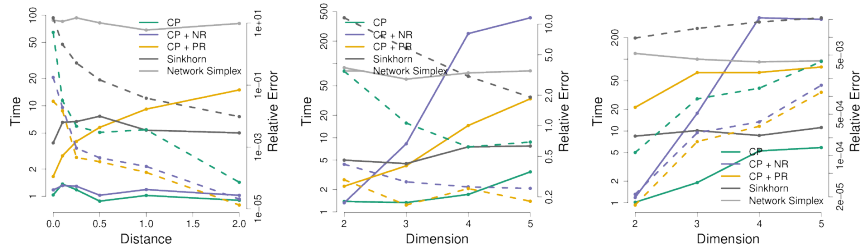
(a) (b)

Figure 7: (a) A comparison of computation time (solid) and relative error (dashed) with respect to the network simplex solution for the CPLEX network simplex, Sinkhorn distance and the proposed multiscale strategy with increasing number of points. (b) The number of total paths considered is roughly constant with four times the number of points in the problem, i.e., the multiscale approach results in only a linear increase in problem size instead of quadratic for a direct approach. The number of points on the x-axis denotes the number of source points $|\mathbf{X}|$ which is approximately equal to the number of target points $|\mathbf{Y}|$. The Sinkhorn approach is competitive in computation time for smaller problems. For larger problems only the multiscale strategies outperform the Sinkhorn approach quickly, and are the only algorithms that remain viable.

Figure 7 shows computation time, relative error and problem size for increasing sample size on the ellipse data set. The Sinkhorn transport employs a regularization term and is thus not expected to converge to the actual transport distance. The comparison of the relative error provides an indication of how far the regularization strays from the true transport distance. For small size problems the Sinkhorn transport is as comparable in speed to the multiscale approach.. The multiscale approach outperforms both competitors for moderate data sizes of a few thousand source and target points. The multiscale approach scales roughly linear in the problem size while both the Sinkhorn and network simplex scale approximately quadratically. The capacity propagation without refinement runs an order of magnitude faster than including refinements and results in relative errors less than one percent for more than a few hundred points. The network simplex and Sinkhorn approach run out of memory on a 16GB computer for problems larger than around $2 \cdot 10^4$ constraints and about 10^8 variables. The proposed multiscale framework results in a linear increase in problem size for the propagation and refinement strategies tested and, on the same 16GB computer, can solve instances with $2 \cdot 10^6$ constraints (source and target points) in a few hours, which would result in about 10^{12} variables (paths) for the full optimal transport problem. The computation times are comparable to the results reported by Oberman and Ruan (2015); Schmitzer (2015) on examples on regular grids that result in similar sized problems.

The relative error and computation time depend again on the type of problem. Figure 8(a) shows computation time and relative error on transport problems on source and target data sets sampled uniformly from $[0,1]^2$ with different shifts in the target distribution. If required, the relative error can be reduced using multiple capacity propagation and potential refinement iterations as illustrated in Figure 6. In these experiments we set the tolerance parameter for Sinkhorn to $1e^{-5}$, and tried also $1e^{-2}$ to see if this would result in significant speed ups, with minimal accuracy loss, but it resulted in only approximately a 10% speedup, and no fundamental change in the behavior for large problem sizes. The tolerance parameter for CPLEX is set to $1e^{-8}$. The approach by Schmitzer (2015) is similar to the refinement property strategy, we expect that computation time grows similarly as for the potential strategy for problems with large transport distances compared to the source and target support size.

The final experiment tests the performance with respect to the dimensionality of the source and target distributions. We used two uniform distributions $[0,1]^d$ with d varying from 2 to 5 and the target shifted such that the actual transport distance is 0 and 2.



(a) (b) (c)

Figure 8: Computation time (solid) and relative error (dashed) with respect to (a) changes transport distance to support size ratio in two dimensions and (b,c) dimensionality with ground truth transport distance (b) 0 and (c) 2. The source and target distributions are uniform on a square of side length 1 with approximately 5000 points each. The potential refinement increases proportional to the transport distance to support ratio. The neighborhood strategy is less effective in higher dimensions, due to the curse of dimensionality, but performs better than the Sinkhorn approach. The capacity propagation strategy is less affected by the dimensionality of the problem.

5. Application to Brain MR Images

An important building block for the analysis of brain MRI populations is the definition of a metric that measures how different two brain MRI are. A mathematically well motivated and popular approach for distance computations between brain images is based on large deformation diffeomorphic metric mappings (LDDMM (Miller et al., 2002)). Here we explore optimal transport distance as an alternative metric for comparing brain images.

To solve for the optimal transport map between two 3D brain images we extract for each image a point cloud from the intensity volumes. Each point represents a voxel as a point in 3-dimensional space, the location of the voxel. The mass of the point is equal to the intensity value of the voxel, normalized to sum to one over all points. For illustration, Figure 9 shows a single slice extracted from the original volumes and optimal transport maps between the two slices. This 2D problem resulted in point set of approximately $20,000$ points.

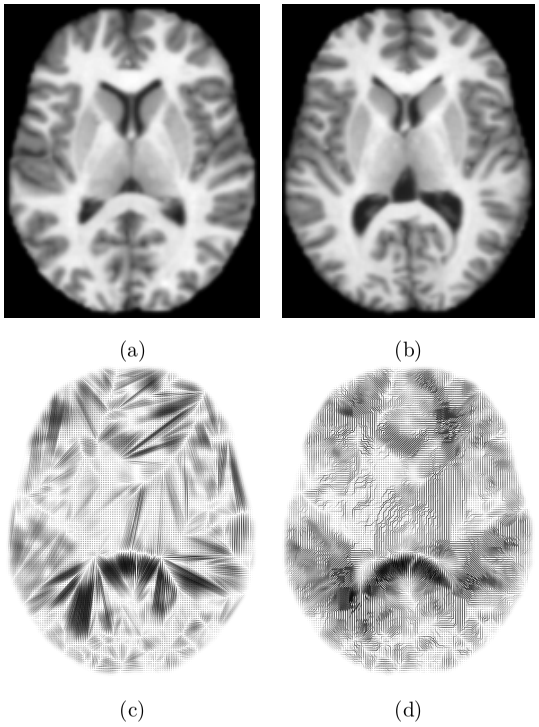


Figure 9: Slice of (a) source and (b) target brain image and optimal transport map for (c) Euclidean and (d) squared Euclidean cost function. For the squared Euclidean cost the optimal transport solutions typically prefer to move many locations small distances, i.e. shift mass among nearest neighbors. For images the neighborhoods are on regular grids resulting in a staircase appearance of the transport plan.

To compare the optimal transport distance to the LDDMM distance we compare how well the distances can be used to predict clinical parameters. Using the pairwise distances, we employ classical multidimensional scaling (MDS) to embed the pairwise distances into

Model	Residual	R^2	F–statistic	p–value
age $= a_0 + \sum_{i=1}^{5} a_i l_i$	**10.5**	**0.82**	**404.9**	$< \epsilon$
age $= a_0 + \sum_{i=1}^{3} a_i x_i$	10.87	0.82	639.5	$< \epsilon$
age $= a_0 + \sum_{i=1}^{5} a_i z_i$	12.04	0.78	297	$< \epsilon$
age $= a_0 + a_1 z_1^1 + a_2 z_2^1 + a_3 z_1^2 + a_4 z_2^2 + a_5 z_1^3 + a_6 z_2^4 + a_7 z_5^4 + a_8 z_1^5$	10.9	0.82	239	$< \epsilon$
MMSE $= a_0 + a_1$age	3.59	0.06	15.82	9.3e-05
MMSE $= a_0 + a_1 l_1$	3.40	0.16	43.13	3.3e-10
MMSE $= a_0 + a_1 x_1$	3.36	0.18	50.30	1.6e-11
MMSE $= a_0 + a_1 z_1 + a_2 z_3 + a_3 z_5$	3.38	0.17	16.31	1.2e-09
MMSE $= a_0 + a_1 z_2^2 + a_2 z_2^3 + a_3 z_4^3 + a_4 z_5^4 + a_5 z_1^5 + a_6 z_3^5 + a_7 z_3^6$	**3.14**	**0.30**	**14.03**	**4.3e-15**
CDR $= a_0 + a_1$age	0.27	0.25	144.5	$< \epsilon$
CDR $= a_0 + a_1 l_1$	0.26	0.34	223.9	$< \epsilon$
CDR $= a_0 + a_1 x_1$	0.25	0.36	248.5	$< \epsilon$
CDR $= a_0 + a_1 z_1 + a_2 z_3 + a_3 z_5$	0.26	0.35	77.5	$< \epsilon$
CDR $= a_0 + a_1 z_4^3 + a_2 z_1^5 + a_3 z_3^5 + a_4 z_5^5 + a_5 a_3^6$	**0.25**	**0.38**	**53.1**	$< \epsilon$

Table 1: Optimal linear regression models from the OASIS data set, for age, mini mental state examination (MMSE) and clinical dementia rating (CDR). The PCA coordinates from the Euclidean distances are denoted with l_i, the diffeomorphic manifold coordinates with x_i, the transport coordinates by z_i and the multiscale transport coordinates with z_i^j from coarsest $j = 1$ to finest $j = 6$ scale. An entry with "$< \epsilon$" denotes quantities smaller than machine precision. The best results are indicated in bold.

Euclidean space. In this embedding each point corresponds to a 3D brain image and thus provides a coordinate system for the relative locations between the brain MRI's. The Euclidean structure of the embedding permits to use standard statistical tools to form prediction models, in our case linear regression.

Within this framework we compare Euclidean, LDDMM and optimal transport distances as the input to the multidimensional scaling step. Note, for the Euclidean distance, i.e. treating each brain MRI as a point in Euclidean space, classical multidimensional scaling is equivalent to a principal component analysis. For the comparisons we used the OASIS brain database (HHMI et al.). The OASIS database consists of T1 weighted MRI of 416 subjects aged 18 to 96. One hundred of the subjects over the age of 60 are diagnosed with mild to moderate dementia. The images in the OASIS data set are already skull-stripped, gain-field corrected and registered to the atlas space of Talairach and Tournoux (1988) with a 12-parameter affine transform. Associated with the data are several clinical parameters. For the linear regression from the MDS embeddings we restrict our attention to the prediction of age, mini mental state examination (MMSE) and clinical dementia rating (CDR).

The MDS computation requires pairwise distances. To speed up computations we reduce the number of points by downsampling the volumes to size $44 \times 52 \times 44$. This results, discarding zero intensity voxels, in point clouds of approximately $40,000$ points for each brain MRI. A single distance computation with capacity propagation takes on the order of 10 seconds resulting in a total computation time of around 2 weeks for all pairwise distances. For embedding the optimal transport distance we consider two approaches:

1. A five dimensional MDS embedding based on the transport cost at the finest scale.

2. A multiscale embedding using multiple five dimensional MDS embeddings, one for the transport cost at each scale.

From the embeddings we build linear regression models, using the embedding coordinates as independent variables and the clinical variables as dependent variables. As in Gerber et al. (2010) we use the Bayesian information criterion (BIC) on all regression subsets to extract a models that trade-off complexity, i.e. number of independent variables, with the quality of fit. Table 1 shows the results of the optimal transport, with squared Euclidean cost, distances compared to the results reported in Gerber et al. (2010). The transport based approach shows some interesting behaviours. The single scale model performs worse on age while performing similar on MMSE and CDR. The multiscale transport models perform similar to the LDDMM approach except for MMSE where it almost doubles the explained variance R^2. This suggests that the multiscale approach captures information about MMSE not contained in a single scale and prompts further research of multiscale based models to predict clinical parameters.

6. Conclusion

The multiscale framework extends the size of transport problems that can be solved by linear programming by one to two orders of magnitude in the size of the point sets.

The framework is flexible, the linear program at each scale can be solved by any method. Depending on the refinement strategy a dual solution would need to be constructed as well. The method can also be applied to solve the linear assignment problem. The solution at the finest scale will be binary if $n = m$, however, at intermediate steps a binary solution is not guaranteed unless at each scale the source and target sets have the same number of points. This could be enforced during the construction of the multiscale structures. As currently defined the method requires point set inputs for the multiscale constructing of the transport problems. However, one could design methods that construct such a multiscale structure from a cost matrix only. However, for large problems pairwise computations of all costs is typically prohibitively expensive.

The multiscale transport framework provides several options for further research. The framework can be combined with various regularizations, e.g. depending on scale and location and induces a natural multiscale decomposition of the transport map. This decomposition can be used to extract information about relevant scales in a wide variety of application domains. We are currently investigating the use of such multiscale decompositions for representation and analysis of sets of brain MRI.

The capacity constraint propagation approach suggests a different venue for further exploration. The capacity propagation strategy does not hinge on a geometrical notion of neighborhoods and is a suitable candidate to extend the multiscale framework to non-geometric problems. Studying the interdependency between the cost function, hierarchical decomposition of the transport problem (or possibly more generic linear programs) and the efficiency of the capacity constraint propagation is a challenging but interesting problem.

Acknowledgments

We thank the referees for their constructive feedback. This work was supported, in part, by NIH/NIBIB and NIH/NIGMS via 1R01EB021396-01A1: Slicer+PLUS: Point-of-Care Ultrasound.

References

Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993. ISBN 0-13-617549-X.

William K Allard, Guangliang Chen, and Mauro Maggioni. Multi-scale geometric methods for data sets ii: Geometric multi-resolution analysis. *Applied and Computational Harmonic Analysis*, 32(3):435–462, 2012.

Erling D. Andersen and Knud D. Andersen. The MOSEK optimization software. http://www.mosek.com/, 2013.

Alexandr Andoni, Piotr Indyk, and Robert Krauthgamer. Earth mover distance over high-dimensional spaces. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 343–352. Society for Industrial and Applied Mathematics, 2008.

Sigurd Angenent, Steven Haker, and Allen Tannenbaum. Minimizing flows for the monge–kantorovich problem. *SIAM journal on mathematical analysis*, 35(1):61–97, 2003.

Patrice Assouad. Plongements lipschitziens dans $\mathbf{R}^p$. *Bulletin de la Société Mathématique de France*, 111:429–448, 1983.

Franz Aurenhammer, Friedrich Hoffmann, and Boris Aronov. Minkowski-type theorems and least-squares clustering. *Algorithmica*, 20(1):61–76, 1998.

Martin Beckmann. A continuous model of transportation. *Econometrica*, 20(4), 1952.

Jean-David Benamou and Yann Brenier. A computational fluid mechanics solution to the monge-kantorovich mass transfer problem. *Numerische Mathematik*, 84(3):375–393, 2000.

Jean-David Benamou, Brittany D Froese, and Adam M Oberman. Numerical solution of the optimal transportation problem using the monge–ampère equation. *Journal of Computational Physics*, 260:107–126, 2014.

Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

Dimitri P Bertsekas and Paul Tseng. Relaxation methods for minimum cost ordinary and generalized network flow problems. *Operations Research*, pages 93–114, 1988.

Alina Beygelzimer, Sham Kakade, and John Langford. Cover trees for nearest neighbor. In *Proceedings of the 23rd international conference on Machine learning*, pages 97–104. ACM, 2006.

Yann Brenier. Polar factorization and monotone rearrangement of vector-valued functions. *Communications on pure and applied mathematics*, 44(4):375–417, 1991.

Guillaume Carlier, Chloé Jimenez, and Filippo Santambrogio. Optimal transportation with traffic congestion and wardrop equilibria. *SIAM Journal on Control and Optimization*, 47(3):1330–1350, 2008.

Guangliang Chen, M. Iwen, Sang Chin, and M. Maggioni. A fast multiscale framework for data in high-dimensions: Measure estimation, anomaly detection, and compressive measurements. In *Visual Communications and Image Processing (VCIP), 2012 IEEE*, pages 1–6, 2012. doi: 10.1109/VCIP.2012.6410789.

Michael JP Cullen. *A mathematical theory of large-scale atmosphere/ocean flow*. World Scientific, 2006.

W.H. Cunningham. A network simplex method. *Mathematical Programming*, 11(1):105–116, 1976.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transportation. *Advances in Neural Information Processing Systems*, pages 2292–2300, 2013.

Marco Cuturi and David Avis. Ground metric learning. *Journal of Machine Learning Research*, 15(February):533–564, 2014.

Marco Cuturi and Arnaud Doucet. Fast computation of wasserstein barycenters. *International Conference on Machine Learning*, pages 685–693, 2014.

George B Dantzig and Philip Wolfe. Decomposition principle for linear programs. *Operations research*, 8(1):101–111, 1960.

Guy Desaulniers, Jacques Desrosiers, and Marius M Solomon. *Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems*. Springer, 2002.

Jacques Desrosiers and Marco E Lübbecke. *A primer in column generation*. Springer, 2005.

Raphael A. Finkel and Jon Louis Bentley. Quad trees a data structure for retrieval on composite keys. *Acta informatica*, 4(1):1–9, 1974.

Lester Randolph Ford and DR Fulkerson. Solving the transportation problem. *Management Science*, 3(1):24–32, 1956.

Samuel Gerber, Tolga Tasdizen, P. Thomas Fletcher, Sarang Joshi, and Ross Whitaker. Manifold modeling for brain population analysis. *Medical Image Analysis*, 14(5):643 – 653, 2010.

Tilmann Glimm and Nick Henscheid. Iterative scheme for solving optimal transportation problems arising in reflector design. *ISRN Applied Mathematics*, 2013, 2013.

A. Goldberg and R. Tarjan. Solving minimum-cost flow problems by successive approximation. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, STOC '87, pages 7–18, New York, NY, USA, 1987. ACM.

Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

Steven Haker, Lei Zhu, Allen Tannenbaum, and Sigurd Angenent. Optimal mass transport for registration and warping. *International Journal of Computer Vision*, 60(3):225–240, 2004.

HHMI, NRG, and BIRN. Open access series of imaging studies. `http://www.oasis-brains.org`.

Qixing Huang and Leonidas Guibas. Consistent shape maps via semidefinite programming. In *Computer Graphics Forum*, volume 32, 2013.

IBM. Ilog cplex. http://www.ilog.com/products/cplex/, 2013.

P. Indyk and N. Thaper. Fast image retrieval via embeddings. *3rd International Workshop on Statistical and Computational Theories of Vision*, 2003.

Angelo Iollo and Damiano Lombardi. A lagrangian scheme for the solution of the optimal mass transfer problem. *Journal of Computational Physics*, 230(9):3430 – 3442, 2011.

Mark A. Iwen and Mauro Maggioni. Approximation of points on low-dimensional manifolds via random linear projections. *Inference & Information*, 2(1):1–31, 2013. arXiv:1204.3337v1, 2012.

Chris L Jackins and Steven L Tanimoto. Oct-trees and their use in representing three-dimensional objects. *Computer Graphics and Image Processing*, 14(3):249–270, 1980.

Lo Kantorovitch. On the translocation of masses. *Management Science*, 5(1):1–4, 1958.

Morton Klein. A primal method for minimal cost flows with applications to the assignment and transportation problems. *Management Science*, 14(3):205–220, 1967.

Robert Krauthgamer and James R Lee. Navigating nets: simple algorithms for proximity search. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 798–807. Society for Industrial and Applied Mathematics, 2004.

Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Wenjing Liao and Mauro Maggioni. Adaptive geometric multiscale approximations for intrinsically low-dimensional data. *arXiv preprint arXiv:1611.01179*, 2016.

Anna V. Little, Mauro Maggioni, and Lorenzo Rosasco. Multiscale geometric methods for data sets I: Multiscale SVD, noise and curvature. Technical report, MIT-CSAIL-TR-2012-029/CBCL-310, MIT, Cambridge, MA, September 2012.

F. Memoli M. Ovsjanikov, Q. Merigot and L. Guibas. One point isometric matching with the heat kernel. *Proc. Eurographics Symposium on Geometry Processing (SGP)*, 2010.

Mauro Maggioni, Stanislav Minsker, and Nate Strawn. Multiscale dictionary learning: Non-asymptotic bounds and robustness. *Journal of Machine Learning Research*, 17(1):43–93, January 2016. ISSN 1532-4435.

Andrew Makhorin. GLPK (GNU linear programming kit). http://www.gnu.org/software/glpk, 2013.

Quentin Mérigot. A multiscale approach to optimal transport. In *Computer Graphics Forum*, volume 30, pages 1583–1592. Wiley Online Library, 2011.

Michael I Miller, Alain Trouvé, and Laurent Younes. On the metrics and euler-lagrange equations of computational anatomy. *Annual review of biomedical engineering*, 4(1):375–405, 2002.

Gaspard Monge. *Mémoire sur la théorie des déblais et des remblais*. De l'Imprimerie Royale, 1781.

Adam M Oberman and Yuanlong Ruan. An efficient linear programming method for optimal transportation. *arXiv preprint arXiv:1509.03668*, 2015.

James B Orlin. A polynomial time primal network simplex algorithm for minimum cost flows. *Mathematical Programming*, 78(2):109–129, 1997.

N. Papadakis, G. Peyré, and E. Oudet. Optimal Transport with Proximal Splitting. *ArXiv e-prints*, april 2013.

Leonidas Guibas Qixing Huang, Fan Wang. Functional map networks for analyzing and exploring large shape collections. *ACM Transactions on Graphics (TOG)*, 33(4), 2014.

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. A metric for distributions with applications to image databases. In *Computer Vision, 1998. Sixth International Conference on*, pages 59–66. IEEE, 1998.

Bernhard Schmitzer. A sparse multi-scale algorithm for dense optimal transport. *arXiv preprint arXiv:1510.05466*, 2015.

Bernhard Schmitzer and Christoph Schnörr. A hierarchical approach to optimal transport. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 452–464. Springer, 2013.

S. Shirdhonkar and D.W. Jacobs. Approximate earth mover's distance in linear time. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008.

J. Talairach and P. Tournoux. *Co-planar stereotaxic atlas of the human brain. 3-Dimensional proportional system: an approach to cerebral imaging*. Thieme, New York, 1988.

Robert E. Tarjan. Dynamic trees as search trees via euler tours, applied to the network simplex algorithm. *Mathematical Programming*, 78(2):169–177, 1997.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

Journal of Machine Learning Research 18 (2017) 1-46 Submitted 3/16; Revised 5/17; Published 8/17

Non-parametric Policy Search with Limited Information Loss

Herke van Hoof HERKE.VANHOOF@MAIL.MCGILL.CA
School of Computer Science, McGill University,
McConnell Eng Bldg, Room 318, 3480 University St, Montreal, Quebec, Canada

Gerhard Neumann GNEUMANN@LINCOLN.AC.UK
Lincoln Centre for Autonomous Systems, Lincoln University, Lincoln, United Kingdom

Jan Peters PETERS@IAS.TU-DARMSTADT.DE
Intelligent Autonomous Systems Institute, Technische Universität Darmstadt, Darmstadt, Germany
Robot Learning Lab, Max Planck Institute for Intelligent Systems, Tübingen, Germany

Editor: Michael Bowling

Abstract

Learning complex control policies from non-linear and redundant sensory input is an important challenge for reinforcement learning algorithms. Non-parametric methods that approximate values functions or transition models can address this problem, by adapting to the complexity of the data set. Yet, many current non-parametric approaches rely on unstable greedy maximization of approximate value functions, which might lead to poor convergence or oscillations in the policy update. A more robust policy update can be obtained by limiting the information loss between successive state-action distributions. In this paper, we develop a policy search algorithm with policy updates that are both robust and non-parametric. Our method can learn non-parametric control policies for infinite horizon continuous Markov decision processes with non-linear and redundant sensory representations. We investigate how we can use approximations of the kernel function to reduce the time requirements of the demanding non-parametric computations. In our experiments, we show the strong performance of the proposed method, and how it can be approximated efficiently. Finally, we show that our algorithm can learn a real-robot under-powered swing-up task directly from image data.

Keywords: Reinforcement Learning, Kernel Methods, Policy Search, Robotics

1. Introduction

Learning continuous valued control policies directly from sensory input presents a major obstacle to applying reinforcement learning (RL) methods effectively in realistic settings. In such settings, there exist two major problems. First, the dimensionality of sensory inputs often makes discretization of the state space infeasible. Secondly, in such settings, the amount of data that can be used to learn policies is often severely limited, increasing the bias in policy updates which can lead to oscillations or divergence (Mannor et al., 2007; Peters et al., 2008).

For the first problem, algorithms have been developed that rely on human-designed features for value function approximations or specialized parametric policies (Kaelbling et al., 1996; Kober et al., 2013; Bartlett, 2003). However, the applicability of such methods is limited in non-linear, redundant sensory domains where defining good features to linearly approximate value functions or policies is non-trivial. Alternative methods use deep neural networks to represent value functions or policies non-linearly (Mnih et al., 2015; Schulman et al., 2015; Lillicrap et al., 2016; Mnih et al., 2016). These representations have many parameters, and thus tend to require sampling many data points.

Recently, there has been a lot of progress towards avoiding the dependence on engineered features by using non-parametric techniques. Such techniques often use kernel functions to implicitly define a (possibly infinite) features space, replacing the manual definition of features. In contrast to task-specific hand-tuned feature spaces, many popular kernels are applicable to a large number of problems as the resulting representation can adapt to the complexity of the data. Non-parametric techniques have successfully been used in value-function methods, for example by Grünewälder et al. (2012b), Nishiyama et al. (2012), and Kroemer and Peters (2011). An overview of related work on non-parametric methods is given in Section 4.3 on page 32. Such methods generally require the inversion of matrices that grow with the number of data points, which limits their applicability. Another shortcoming is that these methods are still susceptible to the problem of data scarcity.

The problem of data scarcity is aggravated by the lack of a notion of the sampled data or sampling policy in many reinforcement learning approaches. At any point in the learning process, knowledge about the learning problem tends to be represented in two ways. First, the outcomes of recently sampled state-action pairs are stored in memory. Secondly, the current sampling policy implicitly represents the state-action pairs sampled longer ago (as well as the initial policy). Any policy update that causes the policy to represent new samples better might cause the policy to represent older samples less well, thus losing information. Many methods update policies or value functions without regard to the sampling policy. Since the recently sampled state-action pairs stored in, e.g., a mini-batch or replay memory are usually not sufficient to completely characterize the learning problem, such updates can result in a critical loss of essential information.

As a result, choosing an improved policy purely based on sampled returns often yields fast but premature convergence to a sub-optimal policy. Such updates favor biased solutions that eliminate states in which, by chance, only bad actions have been tried out. This problem is known as *optimization bias* (Mannor et al., 2007). Optimization biases may appear in both on- and off-policy reinforcement learning methods due to under-sampling (e.g., if we cannot sample sufficiently many of the state-action pairs prescribed by a policy, we will over-fit), model errors or even the policy update step itself.

In an on-line setting, many methods address this problem implicitly by staying close to the previous policy. For example, policy gradient methods allow only small incremental policy changes. The Fisher information metric—that occurs in policy updates using the natural policy gradients (Kakade, 2002; Peters and Schaal, 2008)—can be seen as a Taylor expansion of the loss of information or *relative entropy* between the path distributions generated by the original and the updated policy (Bagnell and Schneider, 2003a). Instead of bounding this Taylor approximation, we can explicitly bound the relative entropy between successive state-action distributions, leading to our Relative Entropy Policy Search (REPS) algorithm.[1] We discuss related policy search methods that limit the information loss in Section 4.1 on page 29.

1. This paper draws from our earlier conference papers (Peters et al., 2010; van Hoof et al., 2015a,b).

©2017 Herke van Hoof, Gerhard Neumann and Jan Peters.

License: CC-BY 4.0, see https://creativecommons.org/licenses/by/4.0/. Attribution requirements are provided at http://jmlr.org/papers/v18/16-142.html.

In this paper, we propose a method based on this insight, that allows us to compute new policies given a data distribution for both off-policy and on-policy reinforcement learning. We start from the optimal control problem statement subject to the constraint that the loss of information is bounded. For continuous domains, where a suitable set of features is often not available, we develop a non-parametric version of this algorithm. This algorithm uses general kernels to define (possibly infinite) feature spaces implicitly, and considers ways to efficiently approximate this method to make it applicable to large data sets.

In our experiments, we show that our method outperforms other non-parametric methods on a reaching task and an under-powered swing-up task. We also show that our method performs well compared to neural-network based methods on a variant of the puddle world task with a 400-dimensional redundant input representation. Furthermore, we evaluate different approximations to process larger data sets efficiently. Finally, we show that using such an approximation, a real-robot pendulum swing-up task can be learned from vision data.

1.1 Problem Statement and Notation

In a Markov decision process (MDP), an agent in state $\mathbf{s}$ selects an action $\mathbf{a} \sim \pi(\mathbf{a}|\mathbf{s})$ according to a (possibly stochastic) policy π and receives a reward $\mathcal{R}_\mathbf{s}^\mathbf{a} \in \mathbb{R}$. We will assume continuous state-action spaces: $\mathbf{s} \in \mathcal{S} = \mathbb{R}^{D_\mathrm{s}}$, $\mathbf{a} \in \mathcal{A} = \mathbb{R}^{D_\mathrm{a}}$. If the Markov decision process is ergodic, for each policy π, there exists a stationary distribution $\mu_\pi(\mathbf{s})$ such that $\int_\mathcal{S}\int_\mathcal{A} \mathcal{P}_{\mathbf{ss}'}^\mathbf{a}\pi(\mathbf{a}|\mathbf{s})\mu_\pi(s)\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s} = \mu_\pi(\mathbf{s}')$, where $\mathcal{P}_{\mathbf{ss}'}^\mathbf{a} = p(\mathbf{s}'|\mathbf{a},\mathbf{s})$. The goal of a reinforcement learning agent is to choose a policy such that the joint state-action distribution $p_\pi(\mathbf{s},\mathbf{a}) = \mu_\pi(\mathbf{s})\pi(\mathbf{a}|\mathbf{s})$ maximizes the average reward $J(\pi) = \int_\mathcal{S}\int_\mathcal{A} \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})\mathcal{R}_\mathbf{s}^\mathbf{a}\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s}$.

The goal of relative entropy policy search is to obtain policies that maximize the expected reward $J(\pi)$ while bounding the information loss with respect to reference distribution $q(\mathbf{s},\mathbf{a})$, i.e.,

$$\max_{\pi,\mu_\pi} \quad \iint_{\mathcal{S}\times\mathcal{A}} \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})\mathcal{R}_\mathbf{s}^\mathbf{a}\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s}, \tag{1}$$

$$\text{s. t.} \quad \iint_{\mathcal{S}\times\mathcal{A}} \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s} \quad = 1, \tag{2}$$

$$\forall s' \quad \iint_{\mathcal{S}\times\mathcal{A}} \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})\mathcal{P}_{\mathbf{ss}'}^\mathbf{a}\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s} = \mu_\pi(\mathbf{s}'), \tag{3}$$

$$\mathrm{KL}(\pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})||q(\mathbf{s},\mathbf{a})) \ \leq \epsilon, \tag{4}$$

where Eqs. (1)–(3) specify the general reinforcement learning objective (Eq. 1) with the constraints that $\pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})$ is a distribution (Eq. 2) and μ_π is the stationary distribution under policy π (Eq. 3). Equation (4) specifies the additional bound on the KL divergence, where

$$\mathrm{KL}(p(x)||q(x)) = \int p(x)\log(p(x)/q(x))\mathrm{d}x.$$

The reference distribution q is usually set to the state-action distribution induced by previous policies, where the initial explorative policy is a wide, uninformed distribution such as a zero-centered Gaussian with a larger variance. In each iteration, the policy is adapted

to maximize the expected reward while respecting the constraint on the KL divergence. Thus, as learning progresses, the policy typically slowly converges towards a deterministic reward-maximizing policy.

In this paper, we aim at developing a reinforcement learning algorithm for continuous state-action MDPs with non-linear and redundant state representations. We assume the transition and reward models of the MDP are unknown. Furthermore, we will concentrate on infinite-horizon problems.

2. Stable Policy Updates for Stochastic Continuous MDPs

The relative entropy policy search optimization problem in Eqs. (1)–(4) defines controller updates that maximize the expected average reward under a bound on the information loss. However, for continuous systems with stochastic dynamics and non-parametric controllers, it is not straightforward to solve the optimization problem directly. In this section, we explain the steps to obtain a practical algorithm. First, we show how to find the dual of the optimization problem. Subsequently, we discuss how this problem can be solved for stochastic systems that are continuous and non-linear. After that, we discuss how to relax the assumption of ergodicity of the MDP by transforming the average reward MDP into a discounted reward MDP. Solving the optimization problem results in a new optimal policy that is, however, only defined on the current set of samples. Therefore, we discuss how the sample-based optimal policy can be generalized to the entire state space. Finally, we discuss how to set the hyper-parameters of the different steps of our method with minimal manual tuning.

2.1 Finding the Dual Problem

To find the dual to the optimization problem in Eqs. (1)–(4), first, we formulate the Lagrangian. For every constraint, we introduce a Lagrangian multiplier. Because Eq. (3) represents a continuum of constraints, we introduce a corresponding continuous state-dependent Lagrangian multiplier $V(\mathbf{s})$. Instead of summing the contributions for each constraint in the Lagrangian, here, we have to integrate instead. Analogously to the sampling distribution $q(\mathbf{s},\mathbf{a})$, we will write $p_\pi(\mathbf{s},\mathbf{a}) = \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})$ for the proposed state-action distribution to keep the exposition brief. Therefore, the Lagrangian

$$\begin{aligned} L(p,\eta,V,\lambda) = &\iint_{\mathcal{A}\times\mathcal{S}} p_\pi(\mathbf{s},\mathbf{a})\mathcal{R}_\mathbf{s}^\mathbf{a}\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s} + \int_\mathcal{S} V(\mathbf{s}')\left(\iint_{\mathcal{A}\times\mathcal{S}} p_\pi(\mathbf{s},\mathbf{a})\mathcal{P}_{\mathbf{ss}'}^\mathbf{a}\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s} - \mu_\pi(\mathbf{s}')\right)\mathrm{d}\mathbf{s}' \\ &+ \lambda\left(1 - \iint_{\mathcal{A}\times\mathcal{S}} p_\pi(\mathbf{s},\mathbf{a})\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s}\right) + \eta\left(\epsilon - \iint_{\mathcal{A}\times\mathcal{S}} p_\pi(\mathbf{s},\mathbf{a})\log\frac{p_\pi(\mathbf{s},\mathbf{a})}{q(\mathbf{s},\mathbf{a})}\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s}\right). \end{aligned}$$

Using the identity $\mu_\pi(\mathbf{s}) = \int_A p_\pi(\mathbf{s},\mathbf{a})\mathrm{d}\mathbf{a}$, the Lagrangian can be re-shaped in the more convenient form

$$L(p,\eta,V,\lambda) = \lambda - \mathbb{E}_{p_\pi(\mathbf{s},\mathbf{a})}\left[V(\mathbf{s})\right] + \eta\epsilon + \mathbb{E}_{p_\pi(\mathbf{s},\mathbf{a})}\left[\mathcal{R}_\mathbf{s}^\mathbf{a} - \lambda + \int_\mathcal{S} V(\mathbf{s}')\mathcal{P}_{\mathbf{ss}'}^\mathbf{a}\mathrm{d}\mathbf{s}' - \eta\log\frac{p_\pi(\mathbf{s},\mathbf{a})}{q(\mathbf{s},\mathbf{a})}\right].$$

To find the optimal p, we take the derivative of L with respect to p and set it to zero

$$\frac{\partial L}{\partial p_\pi(\mathbf{s},\mathbf{a})} = \mathcal{R}_\mathbf{s}^\mathbf{a} - \lambda + \int_\mathcal{S} V(\mathbf{s}')\mathcal{P}_{\mathbf{ss}'}^\mathbf{a} \mathrm{d}\mathbf{s}' - \eta \log \frac{p_\pi(\mathbf{s},\mathbf{a})}{q(\mathbf{s},\mathbf{a})} - \eta - V(\mathbf{s}) = 0.$$

Therefore, we obtain the new state-action distribution

$$p_\pi(\mathbf{s},\mathbf{a}) = q(\mathbf{s},\mathbf{a}) \exp\left(\frac{\mathcal{R}_\mathbf{s}^\mathbf{a} + \int_\mathcal{S} V(\mathbf{s}')\mathcal{P}_{\mathbf{ss}'}^\mathbf{a} \mathrm{d}\mathbf{s}' - V(\mathbf{s})}{\eta}\right) \exp\left(\frac{-\lambda-\eta}{\eta}\right). \tag{5}$$

The function $V(\mathbf{s})$ resembles a value function, such that $\delta(\mathbf{s},\mathbf{a},V) = \mathcal{R}_\mathbf{s}^\mathbf{a} + \int_\mathcal{S} V(\mathbf{s}')\mathcal{P}_{\mathbf{ss}'}^\mathbf{a} \mathrm{d}\mathbf{s}' - V(\mathbf{s})$ can be identified as a Bellman error. Since $p_\pi(\mathbf{s},\mathbf{a})$ is a probability distribution, we can identify $\exp(-\lambda/\eta - 1)$ to be a normalization factor

$$\exp(-\lambda/\eta - 1) = \frac{1}{\iint_{\mathcal{A}\times\mathcal{S}} q(\mathbf{s},\mathbf{a}) \exp(\delta(\mathbf{s},\mathbf{a},V)/\eta) \,\mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s}} = \frac{1}{\mathbb{E}_{q(\mathbf{s},\mathbf{a})} \exp(\delta(\mathbf{s},\mathbf{a},V)/\eta)},$$

which yields the policy

$$\pi(\mathbf{a}|\mathbf{s}) \propto q(\mathbf{s},\mathbf{a}) \exp\left(\frac{\delta(\mathbf{s},\mathbf{a},V)}{\eta}\right). \tag{6}$$

To obtain the dual function, we re-insert the state-action distribution $p_\pi(\mathbf{s},\mathbf{a})$ in the Lagrangian[2]

$$\begin{aligned} g(\eta,V,\lambda) =& \lambda + \eta\epsilon + \mathbb{E}_{p_\pi(\mathbf{s},\mathbf{a})}\left[\delta(\mathbf{s},\mathbf{a},V) - \lambda - \eta \log \frac{p_\pi(\mathbf{s},\mathbf{a})}{q(\mathbf{s},\mathbf{a})}\right] \\ =& \eta\epsilon + \eta \log\left(\mathbb{E}_{q(\mathbf{s},\mathbf{a})} \exp(\delta(\mathbf{s},\mathbf{a},V)/\eta)\right). \end{aligned}$$

We typically do not know the sampling distribution q, as it depends on the unknown system dynamics. However, the expected value over q can be approximated straightforwardly by taking the average of samples $1,\ldots,n$ taken from q. Note that λ and q do not appear in the final expression of the dual function

$$g(\eta,V) = \eta\epsilon + \eta \log\left(\frac{1}{n}\sum_{i=1}^n \exp(\delta(\mathbf{s}_i,\mathbf{a}_i,V)/\eta)\right). \tag{7}$$

To compute the Bellman error δ, the transition distribution is required. As this distribution is generally not known, δ needs to be approximated. The dual function (7) depends implicitly on reference distribution q through the samples.

2.2 Solving the Dual Problem

To solve Eqs. (1)–(4), we need to find the Lagrangian parameters that minimize the dual function in Eq. (7), i.e., $(\eta^*,V^*) = \arg\min g(\eta,V)$. V is a function with domain $\mathcal{S}$, hence, for continuous domains, we will surely over-fit without additional assumptions. One possibility would be to assume V a function linear in designed features, but good features are task-specific and often hard to define. We will therefore make the more general assumption that V^* is of the form

$$V^* = \sum_{\tilde{\mathbf{s}}\in\tilde{\mathcal{S}}} \alpha_{\tilde{\mathbf{s}}} k_\mathrm{s}(\tilde{\mathbf{s}},\cdot), \tag{8}$$

for some set of states $\tilde{\mathcal{S}}$ and scalars α, and a chosen reproducing kernel k_s. In other words, we assume $V^* \in \mathcal{F}$ for a reproducing kernel Hilbert space (RKHS) $\mathcal{F}$ with kernel k_s. The kernel k_s implicitly defines a (possibly infinite dimensional) feature map $\boldsymbol{\phi}(\mathbf{s}) = k_\mathrm{s}(\mathbf{s},\cdot)$ (Schölkopf et al., 1999). Such an implicit definition has the advantage that we do not need to explicitly compute a feature basis for V^* (Hofmann et al., 2008). Kernels are in most cases easier to choose than feature vectors as the complexity of V^* can grow with the amount of training data. In the final algorithm, we will only work with inner product of implicit features that can be computed using the kernel function, $\boldsymbol{\phi}(\mathbf{s})^T\boldsymbol{\phi}(\mathbf{s}') = k_\mathrm{s}(\mathbf{s},\mathbf{s}')$. Therefore, we do not need to explicitly represent the (possibly infinite dimensional) feature maps (Hofmann et al., 2008; Schölkopf et al., 1999).

2.2.1 Embedding the Transition Model.

Since the transition model $\mathcal{P}_{\mathbf{ss}'}^\mathbf{a}$ in Eq. (5) is unknown, we need to approximate $\mathbb{E}_{\mathbf{s}'}[V(\mathbf{s}')|\mathbf{s},\mathbf{a}]$. One way to do so would be to estimate the joint density $p(\mathbf{s},\mathbf{a},\mathbf{s}')$ using, e.g., kernel density estimation, and solving the integral $\int_\mathcal{S} V(\mathbf{s}')p(\mathbf{s},\mathbf{a},\mathbf{s}')d\mathbf{s}'$. However, kernel density estimation in $\mathcal{S}\times\mathcal{A}\times\mathcal{S}$ requires many training samples, and solving the integral generally requires a time-consuming numerical procedure. Instead, as suggested by Song et al. (2009), we can directly estimate a conditional operator $\mathbf{C}_{S'|S,A}$ that maps features $\boldsymbol{\psi}(\mathbf{s},\mathbf{a})$ of state-action pairs to the expected value of implicit features $\boldsymbol{\phi}(\mathbf{s}') = k_\mathrm{s}(\mathbf{s}',\cdot)$. The resulting embedding $\boldsymbol{\mu}_{\mathbf{s}'|\mathbf{s},\mathbf{a}} = \mathbb{E}_{\mathbf{s}'}[\boldsymbol{\phi}(\mathbf{s}')|\mathbf{s},\mathbf{a}]$ can subsequently be used to approximate functions in $\mathcal{F}$, such as V. Using such embeddings avoids estimating the joint density and leads to good results even for high-dimensional data, and renders calculations of expected values over a function in $\mathcal{F}$ straightforward without numerical integration (Song et al., 2013).

In order to learn the conditional operator, we will use a kernel over the state-action space to implicitly define state-action features $\boldsymbol{\psi}(\mathbf{s},\mathbf{a}) = k_\mathrm{s}(\mathbf{s},\cdot)k_\mathrm{a}(\mathbf{a},\cdot)$. Given a sample $\{(\mathbf{s}_1,\mathbf{a}_1,\mathbf{s}_1'),\ldots,(\mathbf{s}_n,\mathbf{a}_n,\mathbf{s}_n')\}$, the empirical conditional embedding is defined as

$$\hat{\boldsymbol{\mu}}_{S'|s,a} = \hat{\mathbf{C}}_{S'|S,A}\boldsymbol{\psi}(\mathbf{s},\mathbf{a}) = \boldsymbol{\Phi}\boldsymbol{\beta}(\mathbf{s},\mathbf{a}), \tag{9}$$

$$\hat{\mathbf{C}}_{S'|S,A} = \boldsymbol{\Phi}(\mathbf{K}_\mathrm{sa} + l_C\mathbf{I})^{-1}\boldsymbol{\Psi}^T, \tag{10}$$

where $\hat{\mathbf{C}}_{S'|S,A}$ is a learned conditional operator that allows the computation of embedding strengths $\boldsymbol{\beta}(\mathbf{s},\mathbf{a}) = (\mathbf{K}_\mathrm{sa} + l_C\mathbf{I})^{-1}\mathbf{k}_\mathrm{sa}(\mathbf{s},\mathbf{a})$, as suggested by Grünewälder et al. (2012a,b). In this equation, l_C is a regularization coefficient, the matrices $\boldsymbol{\Psi} = [\boldsymbol{\psi}(\mathbf{s}_1,\mathbf{a}_1),\ldots,\boldsymbol{\psi}(\mathbf{s}_n,\mathbf{a}_n)]$ and $\boldsymbol{\Phi} = [\boldsymbol{\phi}(\mathbf{s}_1'),\ldots,\boldsymbol{\phi}(\mathbf{s}_n')]$ consist of implicit feature factors, whereas matrix $\mathbf{K}_\mathrm{sa} = \boldsymbol{\Psi}^T\boldsymbol{\Psi}$ and vector $\mathbf{k}_\mathrm{sa}(\mathbf{s},\mathbf{a}) = \boldsymbol{\Psi}^T\boldsymbol{\psi}(\mathbf{s},\mathbf{a})$ contain kernel function evaluations between pairs of data points.[3] The mean embedding $\hat{\boldsymbol{\mu}}_{S'|s,a}$ can be seen as a vector-valued kernel ridge regressor that maps $\mathbf{s},\mathbf{a}$ pairs to the expected function $k_\mathrm{s}(\mathbf{s}',\cdot)$ (Grünewälder et al., 2012a).

2. A detailed derivation is given in Appendix A.

3. This means $[\mathbf{K}_\mathrm{sa}]_{ij} = k_\mathrm{s}(\mathbf{s}_i,\mathbf{s}_j)k_\mathrm{a}(\mathbf{a}_i,\mathbf{a}_j)$, $[\mathbf{k}_\mathrm{sa}(\mathbf{s},\mathbf{a})]_i = k_\mathrm{s}(\mathbf{s}_i,\mathbf{s})k_\mathrm{a}(\mathbf{a}_i,\mathbf{a})$.

2.2.2 Evaluation of V.

In the next step, we want to use the conditional embedding to evaluate V. Conditional embeddings in reproducing kernel Hilbert spaces have the property that conditional expectations of functions in $\mathcal{F}$ can be calculated as inner product in the Hilbert space (Song et al., 2009). Since $V \in \mathcal{F}$, i.e., is of the form (8), the expected value of V can be approximated using the embedded distribution (Song et al., 2009; Grünewälder et al., 2012a,b), i.e.,

$$\mathbb{E}_{\mathbf{s}'}[V(\mathbf{s}')|\mathbf{s},\mathbf{a}] = \left\langle V, \hat{\boldsymbol{\mu}}_{S'|s,a} \right\rangle_{\mathcal{F}} = \sum_{i=1}^{n} \beta_i(\mathbf{s},\mathbf{a}) V(\mathbf{s}'_i).$$

In the dual function g from Eq. (7), V is now only evaluated at sampled states $\mathbf{s}_i$ and $\mathbf{s}'_i$. As we assumed $V \in \mathcal{F}$, the generalized representer theorem (Schölkopf et al., 2001) tells us that there is at least one optimum of the form (8) with $\tilde{\mathcal{S}}$ the set of all sampled states.[4] Consequently, $\mathbb{E}_{\mathbf{s}'}[V(\mathbf{s}')|\mathbf{s},\mathbf{a}] - V(\mathbf{s}) = \boldsymbol{\alpha}^T \tilde{\mathbf{K}}_{\mathrm{s}} \boldsymbol{\beta}(\mathbf{s},\mathbf{a}) - \boldsymbol{\alpha}^T \mathbf{k}_{\mathrm{s}}(\mathbf{s})$, where $\tilde{\mathbf{K}}_{\mathrm{s}}$ is the Gram matrix with entries $[\tilde{\mathbf{K}}_{\mathrm{s}}]_{ji} = k_{\mathrm{s}}(\tilde{\mathbf{s}}_j, \mathbf{s}'_i)$, and $[\mathbf{k}_s(\mathbf{s})]_j = k_{\mathrm{s}}(\tilde{\mathbf{s}}_j, \mathbf{s})$.

2.2.3 Finding a Numerical Solution.

The dual problem can now be restated in terms of η and $\boldsymbol{\alpha}$, as

$$\min_{\eta,\boldsymbol{\alpha}} g(\eta,\boldsymbol{\alpha}) = \eta\epsilon + \eta \log\left(\sum_{i=1}^{n} \frac{1}{n} \exp\left(\frac{\delta(\mathbf{s}_i,\mathbf{a}_i,\boldsymbol{\alpha})}{\eta}\right)\right), \quad \text{s.t. } \eta \geq 0, \tag{11}$$

$$\delta(\mathbf{s},\mathbf{a},\boldsymbol{\alpha}) = \mathcal{R}^{\mathbf{a}}_{\mathbf{s}} + \boldsymbol{\alpha}^T \left(\tilde{\mathbf{K}}_{\mathrm{s}} \boldsymbol{\beta}(\mathbf{s},\mathbf{a}) - \mathbf{k}_{\mathrm{s}}(\mathbf{s})\right). \tag{12}$$

This objective is convex, and since the analytic gradient and Hessian for this objective are straightforward to obtain[5], we employ second order optimization methods to find the optimal η and $\boldsymbol{\alpha}$. Jointly optimizing for η and $\boldsymbol{\alpha}$ is rather slow because of the need of a constrained optimization method. As proposed by Lioutikov et al. (2014), we use a coordinate-descent-like approach where, iteratively, η and $\boldsymbol{\alpha}$ are minimized separately, keeping the value of the other variable fixed. Thus, a fast unconstrained convex optimization algorithm can be used to minimize $\boldsymbol{\alpha}$, whereas a more expensive constrained minimizer can be used for η. Minimizations are run for a fixed number of updates (instead of until convergence), and every iteration is initialized at the result of the last optimization, to speed up the optimization. Iterations continue until the constraints are fulfilled within an acceptable tolerance.

If we choose kernels $\mathbf{k}_{\mathrm{s}}(\mathbf{s}_i,\mathbf{s}_j) = \tilde{\phi}(\mathbf{s}_i)^T \tilde{\phi}(\mathbf{s}_j)$ and $\mathbf{k}_{\mathrm{sa}}((\mathbf{s}_i,\mathbf{a}_i),(\mathbf{s}_j,\mathbf{a}_j)) = \mathbb{1}_{\{(\mathbf{s}_i,\mathbf{a}_i)\}}((\mathbf{s}_j,\mathbf{a}_j))$, we obtain the original REPS formulation (Peters et al., 2010) as a special case, with corresponding Bellman error

$$\delta(\mathbf{s},\mathbf{a},\boldsymbol{\alpha}) = \mathcal{R}^{\mathbf{a}}_{\mathbf{s}} + \boldsymbol{\alpha}^T(\tilde{\phi}(\mathbf{s}') - \tilde{\phi}(\mathbf{s})). \tag{13}$$

In these equations, $\mathbb{1}$ is the indicator function, $\tilde{\phi}$ is a set of hand-crafted features, and $\mathbf{s}'$ is the observed outcome of applying action $\mathbf{a}$ in state $\mathbf{s}$. Our generalization allows the selection of widely applicable kernels that do not depend on hand-crafted features. Furthermore, avoiding the identity kernel function over the state-action space allows efficient learning in stochastic systems.

4. A sketch of the proof following Schölkopf et al. (2001) is given in Appendix B.
5. The dual and its partial derivatives and Hessians are given in Appendix A, together with a proof of convexity of the dual function.

2.3 Ensuring a Stationary Distribution

The REPS formulation (Eqs. 1–4) assumes the existence of a stationary distribution. However, not all MDPs have a stationary distribution for every policy π. For systems that do have a stationary distribution, steady-state behavior might not be realizable for real systems that need to be started and stopped. Furthermore, transient behavior, such as the swing-up of a pendulum, might be of greater interest than steady-state behavior.

We can ensure the system has a stationary distribution that includes such transients by resetting the system with a probability $1-\gamma$ at each time step. The system is subsequently set to a state from the initial state distribution $p_1(\mathbf{s})$. In this case, the expected value of V at the next time step is given by

$$\begin{aligned}\mathbb{E}[V(\mathbf{s}')|\mathbf{s},\mathbf{a}] &= \int_{\mathcal{S}} \gamma \mathcal{P}^{\mathbf{a}}_{\mathbf{ss}'} V(\mathbf{s}') + (1-\gamma) p_1(\mathbf{s}') V(\mathbf{s}') \mathrm{d}\mathbf{s}', \\ &= \gamma \boldsymbol{\alpha}^T \tilde{\mathbf{K}}_{\mathrm{s}} \boldsymbol{\beta}(\mathbf{s},\mathbf{a}) + (1-\gamma) \boldsymbol{\alpha}^T \hat{\boldsymbol{\mu}}_{S1}, \end{aligned} \tag{14}$$

where $\hat{\boldsymbol{\mu}}_{S1}$ is the empirical (observed) embedding of the initial state distribution and $\mathcal{P}^{\mathbf{a}}_{\mathbf{ss}'}$ are the transition probabilities of the original MDP. Such a reset procedure enables learning by removing the impracticable requirement of infinite roll-out length. In this way, we obtain a discounted setting similar to that used in RL methods that optimize the accumulated discounted reward.

2.4 Generalization of the Sample-Based Policy

The parameters resulting from the optimization, η and $\boldsymbol{\alpha}$, can be inserted back in Eqs. (12) and (6) to yield the desired probabilities $\{p_\pi(\mathbf{s}_1,\mathbf{a}_1),\ldots,p_\pi(\mathbf{s}_n,\mathbf{a}_n)\}$ at the sampled $(\mathbf{s},\mathbf{a})$ pairs (where $p_\pi(\mathbf{s},\mathbf{a}) = \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})$ as before). Conditioning on the current state yields the policy to be followed in the next iteration. However, since states and actions are continuous, we need to generalize from these weighted samples to nearby data points. To this end, we want to find a generalizing stochastic policy $\tilde{\pi}(\mathbf{a}|\mathbf{s})$ conditioned on the observed policy samples.

We first consider parametric policies $\tilde{\pi}(\mathbf{a}|\mathbf{s};\boldsymbol{\theta})$ linear in features $\boldsymbol{\phi}(s)$ with parameters $\boldsymbol{\theta}$. Later, we will generalize our results to non-parametric policies. We place a Gaussian prior over the unknown policy parameters, and choose a Gaussian noise model for the conditional over actions. Consequently, a Bayesian model is specified that will allow us to find a posterior over parameters $\boldsymbol{\theta}$

$$\begin{aligned} p(\boldsymbol{\theta}) &= \mathcal{N}(0, \alpha^{-1}\mathbf{I}), \\ \tilde{\pi}(\mathbf{a}|\mathbf{s};\boldsymbol{\theta}) &= \mathcal{N}(\boldsymbol{\theta}^T\boldsymbol{\phi}(\mathbf{s}), \beta^{-1}\mathbf{I}). \end{aligned}$$

By conditioning on the sampled state-action pairs, we obtain the familiar update equation

$$p(\boldsymbol{\theta}|\mathbf{a}_1,\ldots,\mathbf{a}_n,\mathbf{s}_1,\ldots,\mathbf{s}_n) = z^{-1} p(\boldsymbol{\theta}) \prod_{i=1}^{n} \tilde{\pi}(\mathbf{a}_i|\mathbf{s}_i;\boldsymbol{\theta}) \quad (\mathbf{s}_i,\mathbf{a}_i) \sim p_\pi(\mathbf{a},\mathbf{s}).$$

Algorithm 1 Policy iteration with relative entropy policy search (REPS)

repeat
- generate roll-outs according to $\tilde{\pi}_{i-1}$
- minimize dual — $\eta^*, \boldsymbol{\alpha}^* \leftarrow \arg\min g(\eta, \boldsymbol{\alpha})$ — Eq. 11
- calculate Bellman errors for each sample — $\delta_j \leftarrow \mathcal{R}_j + \boldsymbol{\alpha}^{*T}\left(\tilde{\phi}(\mathbf{s}'_j) - \tilde{\phi}(\mathbf{s}_j)\right)$ — Eq. 13
- calculate the sample weights — $w_j \leftarrow \exp(\delta_j/\eta^*)$ — Sec. 2.4
- fit a generalizing policy — $\tilde{\pi}_i(\mathbf{a}|\mathbf{s}) = \mathcal{N}(\mu(\mathbf{s}), \sigma^2(\mathbf{s}))$ — Sec. 2.4

until convergence

Here, z^{-1} is a normalization factor. However, our samples are drawn from $q(\mathbf{a}, \mathbf{s})$ and not $p_\pi(\mathbf{a}, \mathbf{s})$. From a log transformation of the update equation, we see that we can use importance sampling to estimate $\boldsymbol{\theta}$ using the approximation

$$\begin{aligned}
\log p(\boldsymbol{\theta}|\mathbf{a}_1, \mathbf{s}_1, \ldots, \mathbf{a}_n, \mathbf{s}_n) &= \log p(\boldsymbol{\theta}) + \sum_{i=1}^{n} \log \tilde{\pi}(\mathbf{a}|\mathbf{s}; \boldsymbol{\theta}) + \text{const.} && (\mathbf{s}_i, \mathbf{a}_i) \sim p_\pi(\mathbf{a}, \mathbf{s}), \\
&\approx \log p(\boldsymbol{\theta}) + \sum_{i=1}^{n} \frac{p_\pi(\mathbf{a}, \mathbf{s})}{q(\mathbf{a}, \mathbf{s})} \log \tilde{\pi}(\mathbf{a}|\mathbf{s}; \boldsymbol{\theta}) + \text{const.} && (\mathbf{s}_i, \mathbf{a}_i) \sim q(\mathbf{a}, \mathbf{s}), \\
&= \log p(\boldsymbol{\theta}) + \sum_{i=1}^{n} \log \tilde{\pi}(\mathbf{a}|\mathbf{s}; \boldsymbol{\theta})^{\frac{p_\pi(\mathbf{a}, \mathbf{s})}{q(\mathbf{a}, \mathbf{s})}} + \text{const.} && (\mathbf{s}_i, \mathbf{a}_i) \sim q(\mathbf{a}, \mathbf{s}),
\end{aligned}$$

As the new state-action distribution p_π is of the form given in (6), we can write the importance weights

$$w_i = \frac{p_\pi(\mathbf{s}_i, \mathbf{a}_i)}{q(\mathbf{s}_i, \mathbf{a}_i)} = \exp\left(\frac{\delta(\mathbf{s}_i, \mathbf{a}_i, V^*)}{\eta^*}\right).$$

Since $\tilde{\pi}(\mathbf{a}|\mathbf{s}; \boldsymbol{\theta})$ is Gaussian in our model, raising to the power of w_i simply scales the variance by $1/w_i$. By exponentiating both sides again, and using the familiar procedure for weighted Bayesian linear regression (Gelman et al., 2004), we find the predictive distribution

$$\tilde{\pi}(\mathbf{a}|\mathbf{s}) = \mathcal{N}(\beta \boldsymbol{\phi}(\mathbf{s})^T \mathbf{S}_n \boldsymbol{\Phi}^T \mathbf{D}^{-1} \mathbf{A}, \boldsymbol{\phi}(\mathbf{s})^T \mathbf{S}_n \boldsymbol{\phi}(\mathbf{s}) + \beta^{-1}), \quad S_n = (\beta \boldsymbol{\Phi}^T \mathbf{D}^{-1} \boldsymbol{\Phi} + \alpha \mathbf{I})^{-1}, \quad (15)$$

where $\mathbf{D}$ is a diagonal weighting matrix with $\mathbf{D}_{ii} = 1/w_i$ and $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_n]^T$.

The policy mean is of the form of the lower bound introduced by Dayan and Hinton (1997), and the policy that maximizes this lower bound can be found through a weighted linear regression (Peters and Schaal, 2007), although that framework does not employ a Bayesian formulation and therefore cannot represent uncertainty in the parameters. Note that instead of basing the weights on a transformation of the reward function, our approach uses a transformation of the Bellman error, which takes the long-term expected rewards into account. The critical step of choosing the transformation was done by manual design in earlier work (Peters and Schaal, 2007; Dayan and Hinton, 1997)), while here the transformation directly results from the optimization problem (1)–(4). Algorithm 1 shows how the different steps of our approach fit together in the special case of linear value functions and policy, when sampled outcomes are used to approximate the Bellman error. This form of the algorithm was introduced in our earlier work (Peters et al., 2010).

Algorithm 2 Policy iteration with non-parametric REPS (NP-REPS)

repeat
- generate roll-outs according to $\tilde{\pi}_{i-1}$
- calculate kernel embedding strengths — $\boldsymbol{\beta}_j \leftarrow (\mathbf{K}_{\text{sa}} + l_C \mathbf{I})^{-1} \mathbf{k}_{\text{sa}}(\mathbf{s}_j, \mathbf{a}_j)$ — Sec. 2.2
- minimize kernel-based dual — $\eta^*, \boldsymbol{\alpha}^* \leftarrow \arg\min g(\eta, \boldsymbol{\alpha})$ — Eq. 11
- calculate kernel-based Bellman errors — $\delta_j \leftarrow \mathcal{R}_j + \boldsymbol{\alpha}^{*T}\left(\tilde{\mathbf{K}}_{\text{s}} \boldsymbol{\beta}_j - \mathbf{k}_{\text{s}}(\mathbf{s}_j)\right)$ — Eq. 12
- calculate the sample weights — $w_j \leftarrow \exp(\delta_j/\eta^*)$ — Sec. 2.4
- fit a generalizing non-parametric policy — $\tilde{\pi}_i(\mathbf{a}|\mathbf{s}) = \mathcal{N}(\mu(\mathbf{s}), \sigma^2(\mathbf{s}))$ — Sec. 2.5

until convergence

2.5 Non-Parametric Generalizing Policies

For non-parametric policies, Eq. (15) can be kernelized straightforwardly, to yield

$$\tilde{\pi}(\mathbf{a}|\mathbf{s}) = \mathcal{N}(\mu(\mathbf{s}), \sigma^2(\mathbf{s})), \;\; \mu(s) = \mathbf{k}_{\text{s}}(\mathbf{s})^T (\mathbf{K}_{\text{s}} + l\mathbf{D})^{-1} \mathbf{A}, \quad (16)$$

$$\sigma^2(\mathbf{s}) = k(\mathbf{s}, \mathbf{s}) + l - \mathbf{k}_{\text{s}}(\mathbf{s})^T (\mathbf{K}_{\text{s}} + l\mathbf{D})^{-1} \mathbf{k}_{\text{s}}(\mathbf{s}), \quad (17)$$

where kernel vector $\mathbf{k}_{\text{s}}(\mathbf{s}) = \boldsymbol{\phi}(\mathbf{s})^T \boldsymbol{\Phi}$, kernel matrix $\mathbf{K}_{\text{s}} = \boldsymbol{\Phi}^T \boldsymbol{\Phi}$, and l is a free regularization hyper-parameter. Together with other hyper-parameters, such as the kernel bandwidth, l can be set by performing cross-validation on a maximum marginal likelihood objective.

Kober et al. (2011) derived an EM-based policy search approach that uses similar cost-sensitive Gaussian processes (GPs). The regularization term in these GPs is modulated with the inverse of the weight at each data point. However, there are some notable differences. First of all, in our case the weights w are found by transforming the advantage function rather than the reward function, allowing decision making in longer-horizon problems. Furthermore, that approach used a maximum likelihood perspective which allows derivation of the mean, but not the variance. In contrast, we derive our policy update using importance sampling from a Bayesian perspective that allows a principled derivation of the covariance update. Algorithm 2 shows how the different steps of our approach, non-parametric relative entropy policy search (NP-REPS), fit together.

2.6 Hyper-parameter Optimization

The conditional operator has open hyper-parameters, namely, the hyper-parameters of the kernels over $\mathbf{s}$ and $\mathbf{a}$ as well as the regularization parameter l_C. We set l_C and the hyper-parameters of kernel k_{a} through two-fold cross-validation on the objective

$$\sum_{i=1}^{n} \left\| \boldsymbol{\phi}(\tilde{\mathbf{s}}_i)^T \boldsymbol{\phi}(\mathbf{s}'_i) - \boldsymbol{\phi}(\tilde{\mathbf{s}}_i)^T \hat{\mathbf{C}}_{S'|S,A} \boldsymbol{\psi}(\mathbf{s}_i, \mathbf{a}_i) \right\|^2,$$

which minimizes the difference between actual embedding strength and the embedding strength predicted using the conditional operator $\hat{\mathbf{C}}_{S'|S,A}$ introduced in Eq. (10). This objective is based on the cross-validation objective proposed by Grünewälder et al. (2012a), but exploits the fact that the embedding will only be evaluated at known functions $\boldsymbol{\phi}(\tilde{\mathbf{s}})$.

The hyper-parameters of k_s, the kernel for the predicted variable $\mathbf{s}'$, cannot be tuned this way as trivial solutions exist. For example, essentially constant features $\boldsymbol{\phi}(\mathbf{s}')$ with very high bandwidth minimize the prediction error. Instead, we set the hyper-parameters of k_s through minimization of the mean squared TD error in a two-fold cross-validation procedure. We choose this objective since the (residual) TD error is a common objective for feature selection (Parr et al., 2008) in reinforcement learning.

We optimize the hyper-parameters of the Gaussian process policy separately from finding the hyper-parameters of the conditional operator. The employed optimization objective is the weighted marginal likelihood, with weights w_i as discussed in Section 2.5. This objective is maximized in a cross-validation procedure where every roll-out is used as separate fold.

2.7 Efficient Approximations for Large Data Sets

The proposed algorithm requires inverting several $n \times n$ matrices, where n is the number of samples. If open hyper-parameters (such as regularization parameters or kernel bandwidths) need to be optimized, this inversion happens inside the optimization loop. As a consequence, learning becomes slow if more than approximately 5000 samples are used (on current computing hardware). Especially for complex problems and problems requiring a high control frequency, such a soft limit can be prohibitive. In order to scale our method to larger problems, approximate methods with high time efficiency have to be considered. In this section, we will discuss two families of such approximation methods: sparsification of the kernel matrix, and approximation of the kernel function using stochastic features. Related work on efficient calculations for non-parametric reinforcement learning are discussed in Section 4.4 on page 33.

2.7.1 Sparsification Approaches.

One way of scaling up kernel methods is to consider sparsifications, where a small number of pseudo-inputs are used rather than the full data set. Multiple sparsification schemes have been proposed in the context of supervised learning, notably the likelihood approximation used in the projected latent variables (PLV) approach (Seeger et al., 2003) and the Bayesian derivation of sparse pseudo-input Gaussian processes (SPGPs) by Snelson and Ghahramani (2006).

Such sparsifications have been used in a number of RL algorithms (Engel et al., 2003; Xu et al., 2014; Jung and Polani, 2007; Xu et al., 2007; Lever and Stafford, 2015). In many of these algorithms, a quadratic program is optimized to learn the embedding strength of the data points in the active set m. This approach typically results in an approximation $k(\mathbf{x}, \mathbf{x}') \approx \mathbf{k}_m(\mathbf{x})^T \mathbf{K}_{mm}^{-1} \mathbf{k}_m(\mathbf{x}')$. In this equation, $\mathbf{k}_m(\mathbf{x})$ and $\mathbf{K}_{mm}$ are a vector and a matrix, respectively, of similarities to the active set of m data-points (Jung and Polani, 2007; Xu et al., 2007). The same effective kernel is used in the PLV approach (Seeger et al., 2003). Note that, effectively, a non-stationary kernel is obtained, that is parametrized by the data points in the active set m (Jung and Polani, 2007; Xu et al., 2007).

The SPGP approach uses a very similar kernel, but includes a state-dependent regularization term. This term proves to be helpful in gradient-based hyper-parameter optimization (Snelson and Ghahramani, 2006). The covariance of output points is then given by $\mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{mn} + l\mathbf{I} + \boldsymbol{\Lambda}$. In this equation, the n^{th} element on the diagonal of $\boldsymbol{\Lambda}$ is given by

$k(\mathbf{x}_n, \mathbf{x}_n) - \mathbf{k}_m(\mathbf{x}_n)^T \mathbf{K}_{mm}^{-1} \mathbf{k}_m(\mathbf{x}_n)$. The matrix $\mathbf{K}_{nm}$ denotes a Gram matrix between all n input points and the active subset of m data points, while k_m and $\mathbf{k}_m$ denote a scalar and vector of corresponding kernel values.

To derive a sparsification with the same type of effective kernel using the reward-dependent regularization terms introduced in Section 2.5, we can consider the regular update equation for the non-parametric policy mean in Eq. (16),

$$\mu(\mathbf{x}) = \mathbf{k}_n(\mathbf{x})^T (\mathbf{K}_{nn} + l\mathbf{D})^{-1} \mathbf{y},$$

where $\mathbf{y}$ is the vector of all training outputs. We replace the occurrences of the kernel with the effective kernel

$$\mu(\mathbf{x}) = \mathbf{k}_m(\mathbf{x})^T \mathbf{K}_{mm}^{-1} \mathbf{K}_{mn} (\mathbf{K}_{nm}\mathbf{K}_{mm}^{-1}\mathbf{K}_{mn} + l\mathbf{D} + \boldsymbol{\Lambda})^{-1} \mathbf{y}.$$

Now, we can apply the Woodbury identity to obtain the update equation

$$\mu(\mathbf{x}) = \mathbf{k}_m(\mathbf{x})^T (\mathbf{K}_{mm} + \mathbf{K}_{mn}(l\mathbf{D} + \boldsymbol{\Lambda})^{-1}\mathbf{K}_{nm})^{-1} \mathbf{K}_{mn} (l\mathbf{D} + \boldsymbol{\Lambda})^{-1} \mathbf{y}.$$

The empirical conditional embedding (Eq. 10) can similarly be approximated using this effective kernel. Considering the regular update equation for the non-parametric policy covariance in Eq. (17), in similar a fashion we can derive the predictive variance

$$\sigma^2(\mathbf{x}) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}_m(\mathbf{x})^T (\mathbf{K}_{mm}^{-1} + (\mathbf{K}_{mm} + \mathbf{K}_{mn}(l\mathbf{D} + \boldsymbol{\Lambda})^{-1}\mathbf{K}_{nm})^{-1}) \mathbf{k}_m(\mathbf{x}) + l.$$

Analogously, a cost-regularized version of the PLV approach can be obtained in a similar form, but omitting the $\boldsymbol{\Lambda}$ terms. As the proposed sparsification scheme depends on the inverse of $\mathbf{K}_{mm}$, numerical problems can potentially ensue if the active subset m is poorly chosen (such that two almost-equal data points are present). To address this issue, we regularize $\mathbf{K}_{mm}$ where necessary.

Alternatively, we consider fitting a regular Gaussian process to a set of M inducing inputs, with pseudo-targets given by weighted linear regression. For pseudo-outputs $\tilde{\mathbf{y}}$ of M sparse inputs, a Gaussian process would predict $\hat{\mathbf{y}} = \mathbf{K}_{nm}(\mathbf{K}_{mm} + l\mathbf{I})^{-1}\tilde{\mathbf{y}}$ at all (active and passive) inputs. Since we know the true outputs $\mathbf{y}$ for the training data, a maximum likelihood solution can be found using weighted linear regression, considering $\mathbf{K}_{nm}(\mathbf{K}_{mm} + l\mathbf{I})^{-1}$ as design matrix. This approach yields the update equations for the mean

$$\mu(\mathbf{x}) = \mathbf{k}_m \tilde{\mathbf{K}}_{mm}^{-1} \tilde{\mathbf{y}},$$

where $\tilde{\mathbf{K}}_{mm}^{-1}$ is the regularized inverse $(\mathbf{K}_{mm} + l\mathbf{I})^{-1}$. The inducing output values $\tilde{\mathbf{y}}$ are

$$\tilde{\mathbf{y}} = (\tilde{\mathbf{K}}_{mm}^{-1}\mathbf{K}_{mn}\mathbf{D}\mathbf{K}_{nm}\tilde{\mathbf{K}}_{mm}^{-1} + l_2\mathbf{I})^{-1} \tilde{\mathbf{K}}_{mm}^{-1}\mathbf{K}_{mn}\mathbf{D}\mathbf{y},$$

where l_2 is an additional regularization parameter. As the covariance does not depend on the inducing output at the training points, the standard update equation for Gaussian processes can be used in this case. The update equation for the mean corresponds to that of a Gaussian process with an effective kernel $k(x, x') = \mathbf{k}_m(x)^T \tilde{\mathbf{K}}_{mm}^{-1} \tilde{\mathbf{K}}_{mm}^{-1} \mathbf{k}_m(x')$.

2.7.2 Random Fourier Features.

Instead of sparsification, where the exact kernel function is used at a subset of data points, we might instead use an approximation of the kernel function at all data points. One such approach is proposed by Rahimi and Recht (2007). They define a distribution $p(z)$ over mapping functions z such that the inner products in sampled feature spaces are unbiased estimates of the kernel evaluation $k(\mathbf{x},\mathbf{y}) = \mathbb{E}[z(\mathbf{x})^T z(\mathbf{y})] \approx \mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y})$, where $\mathbf{z}_i(\cdot) \sim p(z)$. They propose two kinds of random features that obey this criterion: Fourier features and binning features. As the binning features are suitable only for kernels that solely rely on the L1 norm between data points, we will focus on the Fourier features in this section. These features have been used successfully in various classification and regression tasks (Rahimi and Recht, 2007; Hernández-Lobato et al., 2014; Lu et al., 2016).

For the Fourier features, we require the Fourier transform $\alpha p(\boldsymbol{\omega})$ of a stationary (shift-invariant) kernel $k(\mathbf{x},\mathbf{y})$, where α is a normalization factor that ensures $p(\boldsymbol{\omega})$ is a probability distribution. The inverse Fourier transform is thus

$$k(\mathbf{x},\mathbf{y}) = \alpha \int_{\mathcal{R}^d} p(\boldsymbol{\omega}) e^{i\boldsymbol{\omega}^T(\mathbf{x}-\mathbf{y})} \mathrm{d}\boldsymbol{\omega} = 2\alpha \mathbb{E}[\cos(\boldsymbol{\omega}^T\mathbf{x} + b)\cos(\boldsymbol{\omega}^T\mathbf{y} + b)], \quad b \sim \mathcal{U}(0, 2\pi).$$

We can approximate this integral using samples $(\boldsymbol{\omega}_i, b_i)$ with $i = 1, \ldots, L$ and obtain

$$k(\mathbf{x},\mathbf{y}) \approx 2\alpha \frac{1}{L} \sum_{i=1}^{L} \cos(\boldsymbol{\omega}_i^T\mathbf{x} + b_i)\cos(\boldsymbol{\omega}_i^T\mathbf{y} + b_i) = \mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y}),$$

with $\mathbf{z}_i(\mathbf{x}) = \sqrt{2\alpha L^{-1}}\cos(\boldsymbol{\omega}_i^T\mathbf{x} + b_i)$. As the chosen number of samples L gets smaller, the approximation gets coarser but computations will get faster, as we will need to invert $L \times L$ covariance matrices.

In this article, we will use the squared-exponential (or Gaussian) kernel $k_{\mathrm{es}}(\mathbf{x},\mathbf{y}) = \alpha \exp(-0.5||\mathbf{x}-\mathbf{y}||_{\Sigma^{-1}})$ with diagonal covariance Σ. This kernel has a corresponding Fourier transform $p_{\mathrm{es}}(\boldsymbol{\omega}) = \mathcal{N}(0,\Sigma)$. Furthermore, for periodic data we will use a periodic kernel $k_{\mathrm{p}}(\mathbf{x},\mathbf{y}) = \alpha \exp(-2\sin^2(0.5|\mathbf{x}-\mathbf{x}'|)/\sigma^2)$. This kernel is equivalent to the Gaussian kernel on periodic features

$$k_{\mathrm{p}}(\mathbf{x},\mathbf{y}) = k_{\mathrm{es}}([\cos(\mathbf{x}), \sin(\mathbf{x})]^T, [\cos(\mathbf{y}), \sin(\mathbf{y})]^T).$$

Therefore, suitable random features can be generated as

$$\mathbf{z}_i(x) = \sqrt{2\alpha L^{-1}}\cos([\cos(\mathbf{x}), \sin(\mathbf{x})]\boldsymbol{\omega}_i + b_i).$$

Products of Gaussian and periodic kernels can likewise be written as a single multivariate Gaussian kernel on appropriate features of the inputs and handled in a similar way. As a result, all matrix inversions are now performed on $L \times L$ rather than $n \times n$ matrices.

3. Experiments

We first evaluate our method first on a reaching task, and the under-powered pendulum swing-up task. Then, we will consider two tasks with redundant inputs: we compare different methods on a variant of the puddle-world task with a 400-dimensional input representation, and show the ability of the proposed method to learn a variant of the swing-up task on a real robot that has access only to camera images rather than joint angles. In this section, we will first discuss elements of our set-up that are the same across tasks. Subsequently, we discuss implementation specifics and results for each task separately.

3.1 Experimental Set-up

We assume a realistic exploration setup in which the agent cannot choose arbitrary state-action pairs. Instead, as shown in Alg. 2, from an initial state distribution our agent explores using its stochastic policy. After every 10 roll-outs, the model learner and the policy of the agents are updated. To bootstrap the model and the policy, the agent is given 30 roll-outs using a random exploratory policy initially. To avoid excessive computations, we include a simple forgetting mechanism that only keeps the latest 30 roll-outs at any time.[6] As each roll-out contains 49 time steps on average in the larger tasks (as episodes are reset with a constant probability after each step), most computations are performed on approximately 1500×1500 matrices.

After each update, in simulated runs, the learning progress is evaluated by running the learned policy on 100 roll-outs with a fixed random seed. This data is not used for learning. For every method, we performed 10 trials, each consisting of 20 iterations so that 220 roll-outs were performed per trial (30 initial roll-outs plus 10 per iteration). In real-robot runs we evaluate the learned policy on the training samples, and performed 6 trials of 10 iterations each. The model and policy are refined incrementally in every iteration.

3.2 Compared Methods

We compared learning progress of the proposed approach to that of various other approaches. On the one hand, we consider the non-parametric value-function based methods introduced by Grünewälder et al. (2012b) and Pazis and Parr (2011). On the other hand, we will compare to the performance of neural-network based methods introduced by Lillicrap et al. (2016) and Schulman et al. (2015) that are well-suited for redundant input data. We also compare to versions of REPS with the sample-based model approximation used by Peters et al. (2010) and Daniel et al. (2016), and one that uses a fixed feature set. The relationship of the proposed approach to earlier non-parametric reinforcement learning methods will be discussed in Section 4.3 on page 32.

3.2.1 Sample Based Model.

REPS only needs $\mathbb{E}_{\mathbf{s}'}[V(\mathbf{s}')|\mathbf{s},\mathbf{a}]$ at observed state-action pairs $(\mathbf{s}_i,\mathbf{a}_i)$. Therefore, if the system is deterministic, this expectation is simply $V(\mathbf{s}_i')$ at the observed values for $(\mathbf{s}_i,\mathbf{a}_i)$. In stochastic systems, this sample-based method is used as approximation.

3.2.2 Feature Based REPS with Fixed Basis Functions.

Instead of the non-parametric form of V assumed in this paper, we can follow earlier work and define a fixed feature basis (Peters et al., 2010; Daniel et al., 2016). We choose to use a

6. As a consequence, the reference distribution q is a mixture of the previous three state-action distributions in our experiments.

similar number of the same radial basis functions used in the non-parametric method, but distribute these according to a regular grid over the state-action space.

3.2.3 Approximate Value Iteration.

In this approach by Grünewälder et al. (2012b), the value function is assumed to be an element of the chosen RKHS. The maximization of the Q function requires discretizing $\mathbf{a}$, for which we choose 25 uniform bins in the allowable range. A deterministic policy selects $\mathbf{a}^* = \arg\max Q(\mathbf{s}_i, \cdot)$, but this policy does not explore, yielding bad performance in on-policy learning. To obtain an exploration-exploitation trade-off we replace the maximum by the soft-max operator $a^* \propto \exp(Q(\mathbf{s}_i, \cdot)c/\text{stdev}(Q(\mathbf{s}_i, \cdot)))$. The free parameter c specifies the greediness of the exploration/exploitation trade-off on normalized Q values. In an **on-policy** scheme, the new policy is used to obtain samples for the next iteration.

As a comparison to this on-policy scheme, we also compare using a **grid** of state-action pairs as training data. For a dense grid, this method has a richer input than all other methods, as those other methods start with uninformed roll-outs from the initial-state distribution. In a real system, obtaining such a grid is often unrealistic as the state cannot be set arbitrarily.

3.2.4 Non-parametric Approximate Linear Programming.

Pazis and Parr (2011) introduced a non-parametric method, NPALP, that assumes the value function is Lipschitz. This assumption allows the RL problem to be formalized as a linear program. This method assumes the dynamics are deterministic. A greedy policy is obtained that is optimal if all state-action pairs have been visited. Since visiting all state-action pairs is infeasible in continuous systems, we include exploration by adding Gaussian distributed noise to the action in a fraction ϵ of the selected actions.

3.2.5 Trust Region Policy Optimization.

This method, introduced by Schulman et al. (2015), uses a bound on the expected Kullback-Leibler divergence between successive policies, inspired by a theoretical policy iteration algorithm that guarantees non-decreasing expected returns. The method, TRPO, requires a parametric policy to be defined, but has been shown to work well with deep neural-network policies that can be used in many different tasks, thanks to their flexibility. We use the reference implementation provided in the RLlab framework (Duan et al., 2016). This TRPO implementation requires a fixed time horizon, so we evaluate this model on a modified version of the tasks where the episodes are of fixed length, equal to the expected episode length used for other methods.

3.2.6 Deep Deterministic Policy Gradients.

Lillicrap et al. (2016) introduced a policy gradient algorithm, DDPG, that works with deterministic policies. DDPG is an actor-critic method that uses neural networks for both the Q function and the policy. After each time step, the new experience sample is stored in an 'experience replay' buffer, a randomly sampled batch from this buffer is then used to calculate the policy gradient. This technique diminishes the problem that successive

samples tend to be highly correlated. We again use the RLlab implementation (Duan et al., 2016) with fixed episode length.

3.2.7 Linear Least Squares Policy Iteration

As an example of a linear method, we use LSPI (Lagoudakis and Parr, 2003). This method learns a Q function by minimizing temporal differences. We only include this method to show that linear methods in general do not suffice on our task with pixel inputs. As suggested by Lagoudakis and Parr (2003), we use this method off-line, in contrast to the other methods. Thus, all data used in this method has been gathered using the initial exploration policy.

3.3 Approximation Methods

We compare the approximation methods described in Section 2.7 to each other, as well as to a naive baseline. This comparison evaluates the trade-off between a better approximation quality and a faster computation time across a range of different numbers of features and inducing inputs.

3.3.1 Sub-sampling as Baseline.

For the sub-sampling baseline, we simply train the method as described in Section 2.2 with a subset of the data points, ignoring the points that are not in the random subset.

3.3.2 Sparse Pseudo-input Gaussian Processes (SPGP).

We use the method proposed by Snelson and Ghahramani (2006) to approximate the Gaussian process policy. To incorporate the desirability of data points, we use the modifications proposed in Section 2.7. There is no straightforward extension of this method to learn the Bellman error terms (Eq. 12). Therefore, in these steps of the algorithm we interpret the matrix $\mathbf{K}_{nm}$ introduced in Section 2.7 as a feature matrix. Subsequently, we calculate the Bellman error terms using the feature-based formulation introduced by Peters et al. (2010). In contrast to that work, we do not use single sample roll-outs, but calculate the embedding strengths $\boldsymbol{\beta}$ (Section 2.2) using a linear kernel with the pseudo-features as input. Snelson and Ghahramani (2006) suggest choosing the active subset by maximizing the marginal likelihood. However, this maximization is a computationally intensive process, and for the value function approximation, there is no criterion available equivalent to the maximum likelihood criterion for the supervised learning set-up. Therefore, we choose random subsets when applying this method.

3.3.3 Projected Latent Variables (LPV).

Additionally, we compare to the sparsification method proposed by Seeger et al. (2003). As in the previous method, we interpret the $\mathbf{K}_{nm}$ matrix as features and use these to calculate embedding strengths and Bellman error terms.

3.3.4 Regression-based Sparse GPs.

We also consider the regression-based sparse Gaussian Processes proposed in Section 2.7. Again, the $\mathbf{K}_{nm}$ matrix is interpreted as design matrix for calculation of the embedding strengths and Bellman error terms.

3.3.5 Fourier Transform Based Approximation

The last method we consider is based on the work of Rahimi and Recht (2007), with the extension to desirability-weighted samples as described in Section 2.7. As this method approximates the policy using features, we can use these features to calculate embedding strengths and Bellman error terms in the feature-based formulation introduced by Peters et al. (2010).

3.4 Reaching Task Experiment

In the reaching task, we simulate a simple two-link planar robot. In this task, the agent's actions directly set the accelerations of the two joints. Each link is of unit length and mass, and the system is completely deterministic. The agent gets negative reinforcement $r(\mathbf{s}, \mathbf{a}) = -10^{-4}||\mathbf{a}||_2^2 - ||\mathbf{x} - \mathbf{x}_{\text{des}}||_2^2$ according to the square of the applied action and of the distance of its end-effector to the Cartesian position $\mathbf{x}_{\text{des}} = [0.5,\ 0]$. Note that actions are two dimensional and states are four dimensional (joint positions and velocities). The robot starts stretched-out with the end-effector at $\mathbf{x} = [0,\ 2]$. The maximum applied acceleration is 50ms^2. We use $\gamma = 0.96$, which resets roll-outs after 24 samples, on average.

We use the commonly used squared-exponential (or Gaussian) kernel for angular velocities $\dot{\boldsymbol{\theta}}$ and actions. This kernel is defined as $k_{\text{se}}(\mathbf{x}_i, \mathbf{x}_j) = \exp(-(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{D}(\mathbf{x}_i - \mathbf{x}_j))$ (with $\mathbf{D}$ a diagonal matrix containing free bandwidth parameters). However, for the angles $\boldsymbol{\theta}$ we need a kernel that represents its periodicity. We use the kernel $k_{\text{p}}(x_i, x_j) = \exp(-\sum_d \sin((x_i^{(d)} - x_j^{(d)})/(2\pi))^2/\mathbf{l}_d^2)$, where $\mathbf{l}$ is a vector of free bandwidth parameters. Consequently, we obtain a complete kernel

$$k((\boldsymbol{\theta}_i, \dot{\boldsymbol{\theta}}_i, a_i), (\boldsymbol{\theta}_j, \dot{\boldsymbol{\theta}}_j, a_j)) = k_{\text{p}}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) k_{\text{se}}(\dot{\boldsymbol{\theta}}_i, \dot{\boldsymbol{\theta}}_j) k_{\text{se}}(\mathbf{a}_i, \mathbf{a}_j),$$

composed of three simpler kernel functions.

Feature-based REPS needs a grid over the complete state-action space. Due to difficulties with the six-dimensional state-action space, we omitted this method. The exploration parameter ϵ of the NPALP method was set to 0.1, with the standard deviation of Gaussian noise set to 30Nm^2. The Lipschitz constant was set to 1 with the velocity dimensions scaled by 1/5 for calculating distances. The exploration parameter c of the approximate value iteration method was set to 1.5. These values were manually tuned to yield fast and consistent learning progress. For REPS, we use a KL bound ϵ of 0.5 in our experiments, as this value empirically yielded acceptably fast learning progress while keeping updates smooth enough on a range of learning problems.

3.4.1 Results of the Reaching Task.

The results of the reaching task are shown in Figure 1. As this task is deterministic, the sample-based model is optimal and provides an upper bound to the performance we can

expect to get. Since non-parametric REPS with model learning needs to iteratively learn the transition distribution, its convergence is slower. After inspection of individual trials, the wide variance seems to be caused by occasional failures to find good hyper-parameters for the state-action kernel.

The baseline methods NPALP and value iteration using RKHS embeddings obtain good performance after the couple of iterations. However, if we iterate performing roll-outs and policy updates after those first steps, these methods fail to improve the policy consistently. The grid based value iteration scheme fails in this case: due to memory limitations the maximum grid size we could use was $[5 \times 5 \times 5 \times 5 \times 2 \times 2]$ in the 6-dimensional space, which appears to be insufficient to learn the task.

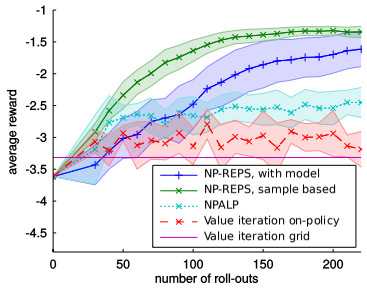
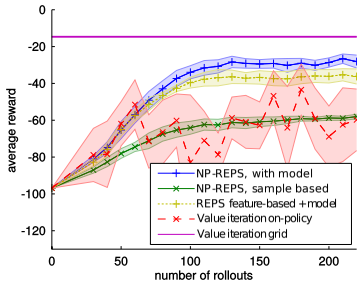


Figure 1: Comparison of learning progress of different methods on the reaching task. As this environment is deterministic, sample-based approximations of the transition functions are optimal. Error bars show twice the standard error of the mean. The value-iteration method does not depend on roll-outs, its performance is shown for comparison.

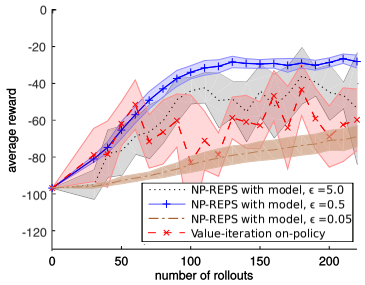
3.5 Low-Dimensional Swing-up Experiment

In this experiment, we simulate a pendulum with a length $l = 0.5\text{m}$ and a mass $m = 10\text{kg}$ distributed along its length. A torque a can be applied at the pivot. The pendulum is modeled by the dynamics equation $\ddot{\theta} = (glm\sin\theta + a - k\dot{\theta})/(ml^2/3)$, where $k = 0.25\text{Ns}$ is a friction coefficient and $g = 9.81$ is the gravitational constant. The controller's sampling frequency is 20 Hz, i.e., every 0.05s the agent gets a reward and chooses a new action. The maximum admissible torque is 30Nm, which prevents a direct swing-up from the downwards position. Additive noise with a variance of $1/dt$ disturbs the controls, resulting in a standard deviation of about 4.5Nm per time step. The reward function was set to $r(s, a) = -10\theta^2 - 0.1\dot{\theta}^2 - 10^{-3}a^2$, where θ is mapped to $[-0.5\pi, 1.5\pi)$ to differentiate the rewards of clockwise and counter-clockwise swing-ups. We use a reset probability of 0.02 ($\gamma = 0.98$).

The algorithms directly access the state variable encoding the angle θ and the angular velocity $\dot{\theta}$, i.e., the state is defined as $\mathbf{s} = [\theta, \dot{\theta}]^T$. The same kernels are used as in the reaching task experiment (Section 3.4). The NPALP method was not designed for stochastic systems and is consequently omitted. We set the grid size for feature-based REPS to $[10 \times 10 \times 10]$ and for the value-iteration method to $[19 \times 11 \times 11]$. The greediness parameter c for on-policy value-iteration was set to 2 after manual tuning. We compare the KL bound ϵ used in the previous experiment, 0.5, to two extreme settings $\epsilon = 5$ and $\epsilon = 0.05$ to illustrate the effect of this bound.



(a) Comparison of learning progress of different methods on the swing-up task.



(b) Comparison of learning progress under different bounds ϵ on the KL-divergence.

Figure 2: Results of the low-dimensional swing-up experiment. Our non-parametric relative entropy method outperforms the other on-policy learners. Error bars show twice the standard error of the mean.

3.5.1 Results of the Low-Dimensional Swing-up.

We show a comparison of the discussed methods in Figure 2a. The value iteration method starts out competitively, but fails to keep improving the policy. Large variance indicates the learning process is unstable. The bounded policy update in REPS makes the learning progress smooth by limiting information loss, limiting exploitation to retain more exploration in early stages of the learning process.

The sample-based baseline model performs considerably worse in this experiment, as it cannot account for stochastic transitions. The variant with fixed features performs well initially, but in later iterations, the non-parametric method is able to focus its representative power on frequently visited parts of the state space, resulting in an improved performance.

We show a comparison for different values of the KL bound ϵ in Figure 2b. We plotted two rather extreme values for ϵ. For a large value of $\epsilon = 5$, we observe behavior that is similar to the greedy value iteration scheme. Initially, performance increases, but learning progress has a high variance and is not smooth. Sub-optimal performance to both the proposed value $\epsilon = 0.5$ and the greedy value-iteration is caused by

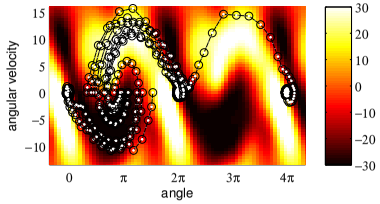


Figure 3: Mean of the learned stochastic policy. Superimposed are 15 trajectories starting at the x-axis between 0 and 2π. Most roll-outs reach the desired inverted pose, possibly after one swing back and forth. One roll-out overshoots and makes a full rotation before stabilizing the pendulum.

instability of the policy fitting described in Section 2.5 when the sample weights span several orders of magnitude. For a low value of $\epsilon = 0.05$, learning progress is very stable but slow. We found that setting $\epsilon = 0.5$ yielded good performance in all tasks we tried.

In contrast to the previous experiment, here, the grid-based value iteration method worked well, as shown in Figure 2a. The policy learned by NP-REPS, shown in Figure 3, sometimes overshoots the inverted position, which the grid based value iteration avoids. However, to reach this performance, a grid of training samples covering the full state-action space was needed. Providing such a grid is only feasible in simulation, as without an existing controller, it is generally not possible to start the dynamical system with arbitrary position and velocity. Furthermore, on higher-dimensional tasks, a grid of sufficient resolution would require impracticably many samples.

3.6 Image-based Puddle World Experiment

In this experiment, we investigated a modified version of the classical puddle-world task with a 400-dimensional redundant representation. In the puddle world task, the agent has to navigate to one corner of a two-dimensional world while avoiding high-cost areas ('puddles'). The aim of this experiment is to confirm the ability of our proposed method to handle such representation, and to compare our method to other method that have been shown to work well with raw sensory representations. We mostly follow the description for the puddle world by Sutton (1996). As size of the goal region we use the area above the line $x + y = 1.8$. We represent the state by a simple rendered pixel image, where the agent is represented by a circle of 8% of the image size (the agent does not 'see' the puddles, but has to learn where they are from experience). We found that for the policy search methods we studied, a relatively low discount factor ($\gamma = 0.96$) or corresponding reset probability (0.04) provided lower-variance results. Such a discount factor makes long-term rewards less salient, thus we modified the per time step reward to -10. To make the task sensible in an average-reward setting, we do not restart when the goal area is reached, rather requiring the agent to learn to maintain this desired position.

A Gaussian kernel directly on pixel images would be very sensitive to even small shifts. A better choice is a kernel exponential in the squared difference between low-pass filtered images. To this end, we convolve the image with a Gaussian kernel of 20% of the image size. The same representation is also used for the other methods. Note, that methods based on neural networks would be able to learn this and other invariances, but this additional expressive power comes at the cost of many parameters that have to be learned. The image is finally down-scaled to 20×20 pixels to be able to reduce the size of a batch of images in working memory. This does not change much for non-parametric methods (as relative distance between images should stay roughly constant), and reduces the number of weights in neural network based methods. We again performed 10 separate trials for each method, using 15 iterations for REPS and an equivalent amount of training experiments for the other methods. In each iteration, 20 roll-outs were performed, retaining the roll-outs from the last 3 iterations in memory as before.

The methods under comparison were tuned as follows. For NP-REPS, we set the bandwidth of the value function to half the maximal pixel intensity and added a $l2$ regularizer $10^{-9}\boldsymbol{\alpha}^T\boldsymbol{\alpha}$ to Eq. (11) to avoid over-fitting as this yielded good empirical performance.

Other hyper-parameters were optimized as explained previously, with the bandwidth of all pixels for the model and policy tied together to speed up optimization. The KL-bound ϵ was again set to 0.5. For LSPI, we discretized the actions in horizontal and vertical directions to three values: the maximum in either direction or 0.

For the neural-network based methods, we started from the default values in the RLlab implementation (Duan et al., 2016). We adapted the batch size / epoch length to the 20 roll-outs performed per iteration. We manually tuned other parameters to maximize empirical performance. This procedure resulted in a KL bound of 0.05 for TRPO (much higher than the default, which is tuned for situations where a lot of data is available but the task is harder). For DDPG, we obtained a learning rate of 10^{-4} for both the Q function and policy, a minimum replay memory size of 1500 and a maximum memory size of 5000 and a batch size of 64 (the relatively high batch size made the method more sample efficient). For DDPG, we used a Gaussian exploration strategy with a decay period of 5000, as well as multi-layer perceptron for the Q function. We tried several convolutional and three-layer feed-forward architectures, and got best performance for two layers with 4 hidden units each for TRPO and two layers with 8 hidden units for DDPG.

3.6.1 Results of the Image-based Puddle World Experiment.

The performance of the compared methods is shown in Figure 4. The poor performance of the LSPI method directly on raw pixels indicates that the problem is non-linear. TRPO with a neural network policy learns relatively slowly (we tried higher learning rates, however, these made learning more unstable but did not yield higher average returns). In our experiments, TRPO needed at least 1000 roll-outs to converge to a good solution. DDPG with a neural network policy and Q function learns about as fast as NP-REPS, but yields high-variance results that tend to be slightly worse by the end of the experiment.

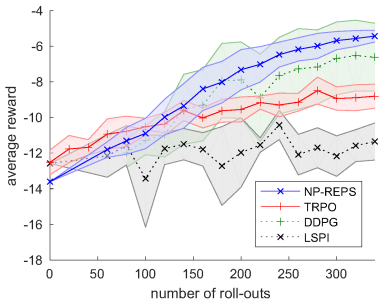


Figure 4: Results of the image-based puddle-world experiment. The RLlab version of DDPG only reports performance once the replay memory is initialized, so this graph starts after 80 roll-outs. Error bars show twice the standard error of the mean.

We want to stress two potential limitations of this experiment. First, difference between the compared methods might be due to either the representation of the policy and/or value function or the algorithm itself. Where NP-REPS uses kernel methods to represent the V function and the policy, TRPO and DDPG use neural networks. LSPI used a value function linear in the raw pixels with a greedy policy. Where in this experiment a non-parametric representation seems beneficial, in domains with visual distractors neural networks might do better. Secondly, this experiment evaluates how well TRPO and DDPG do in a situation with a low number of training roll-outs, whereas these methods are commonly used to work with a much higher number of roll-outs where they tend to do well.

3.7 Approximation Experiment on the Swing-up Task

We evaluated different schemes for approximating the kernel matrix on the pendulum swing-up task described in Section 3.5. We do this by training policies using the various methods discussed in Section 3.3. The trade-off between accuracy and computational speed is defined by the number of inducing inputs for the sparse kernel methods, or by the number of random features used for the Fourier-transform based approximation. The number of features (respectively, inducing inputs) was varied to investigate the trade-off between approximation accuracy and computation speed. We measured the learning progress using the average reward, and additionally logged the time needed per iteration for each condition. We report the average time needed over 10 trials of 10 iterations.

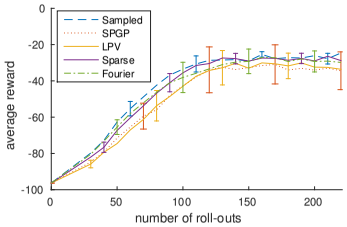
3.7.1 Results and Discussion of the Approximation Experiment.

A comparison of our results using different numbers of features is shown in Figure 5. In Figure 5a, all available inputs were used, and hence no approximation is needed. Thus, the sub-sampling method reduces to the original methods, and performs best. However, if the number of available bases is reduced to 500 (about one third of the available training points), just training on a sampled subset performs very badly, as shown in Figure 5b. In this set-up, the approximation based on a Fourier transform of the kernel seems the most suitable, both in terms of sample efficiency and of asymptotic average reward.

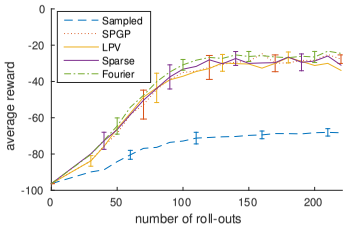
In Figure 5c, the available number of bases is reduced even more, to about 7% of the available training data. In this case, the regression-based sparsification method allows faster learning than the Fourier-based methods and both other sparsification methods, possibly because we used induced inputs $\tilde{y}$ that were optimal in a least-square sense. Considering that only 7% of the basis is available, the drop in performance relative to the full availability is modest.

Overall, the SPGP and LPV sparsification methods perform very similarly and yield similar asymptotic values as the regression-based sparsification methods. The main difference between these methods is the additional regularization term based on the approximation accuracy that Snelson and Ghahramani (2006) introduced. One of the main benefits of this additional regularization is a better behavior of the marginal likelihood for the purpose of optimizing the inducing input points (Snelson and Ghahramani, 2006). As we did not optimize for these points (as that would be too computationally costly to do inside the reinforcement-learning loop), our results seems plausible in this respect.

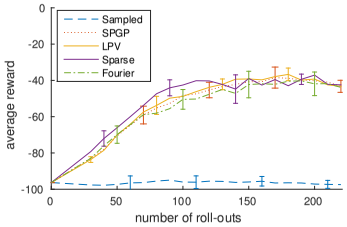
An indication of the time requirement of the different methods is given in Figure 5d. The implementations of the algorithms are not directly comparable, so drawing hard conclusions about the algorithms is not possible on the basis of this data. For example, cross-validation can be implemented using a fast decremental update of the matrix inverse only for the sub-sampled and Fourier-based methods. For the other sparsification methods, there is no straightforward way to implement this speed-up so these were trained using a 2-fold cross-validation setup (rather than using one fold for each trajectory, which would have been



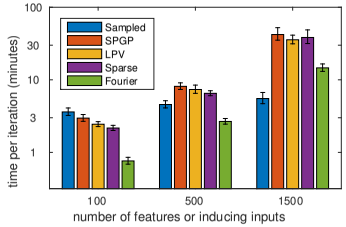
(a) Different approximation methods using 1500 features or all available (on average 1500) inducing inputs.



(b) Different approximation methods using 500 features or inducing inputs.



(c) Different approximation methods using 100 features or inducing inputs.



(d) Time requirement of different approximation methods on a 2.7 GHz processor running in single threads, log scale.

Figure 5: Results of the evaluation of different approximation methods using different numbers of features. In all graphs, error bars indicate twice the standard error of the mean. The learning curves show error bars only for every fifth iteration to keep the figures interpretable. At a medium number of features (b), the Fourier-based random features deliver performance that is almost equal to the original algorithm, while requiring fewer computational resources (d). More computational resources can be saved by using only 100 features or inducing inputs, at the cost of a larger performance gap to the original algorithm (c).

even more expensive computationally). Therefore, these three methods ('SPGP', 'LPV', and 'Sparse') are slowest in cases with many available inducing inputs.

When all data is available, applying any approximation takes more time than the baseline 'sampling' method. However, the 'Fourier' method tends to be much faster when the number of features is reduced—one of the reasons for this effect is that in our implementation, calculated features are cached. This caching benefits feature-based approximations, but not kernel-based ones. Compared to using all available bases without approximating, the 'Fourier' method with 500 bases is about twice as fast at a similar level of performance.

3.8 Real-Robot Swing-up Experiment from Vision Data

In this last experiment, we aim to validate our method on a real-robot task. We consider two scenarios: the learner either has access to the joint angles and velocity, or only to a redundant image-based representations of the state obtained through a camera pointed at the robot. Considering the time it takes to perform robot evaluations, and the wear to the robotic system, we provide a proof-of-concept of the proposed method without comparing to other methods.

3.8.1 Robot Set-up and System Dynamics.

To the end-effector of a Kuka light-weight robot arm, we attached a wooden rod with a mass of roughly 1kg and a length of roughly 80 cm. Red cardboard was used to make the pendulum visually salient. We aim to swing this rod up using the last degree-of-freedom, and limit the torque of the corresponding motor such that the pendulum cannot be swung up directly from the downwards position. This set-up is shown in Figure 6a.

A couple of characteristics of the robot system make the task different from the simulated pendulum swing-up task in the previous section. The robot link is limited in its range of motion to less than 2π and in its velocity to less than about $4.2/s$, whereas the simulation had no such limitation. To prevent the robot from moving hard into these limits, we add a feedback signal that stops the robot if it gets close to those limits for safety.[7] The joint limits are illustrated in Figure 6b.

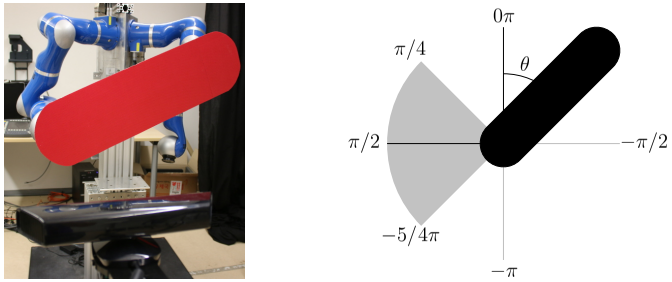
To perform exploration on the real robot, another issue to address is that high jerk motions might damage the gearboxes. Therefore, instead of controlling the torque as in the previous experiments, we will control an increment to the torque. This increment will smoothly be added to the previous torque over the course of one time step. Controlling the increment, rather than directly controlling the torque, has the additional benefit that the applied torque tends to be more consistent over time, preventing 'washing out' of high-frequency control signals on the robot system. To preserve Markov properties, the previous torque has to be appended to the state vector. To provide comparable results over the different robot trials, instead of resetting the system randomly, we reset the system every 50 time steps to a set of ten different starting angles. The starting velocity and acceleration are set to zero.

3.8.2 Camera Set-up and Image Processing.

The camera provides video frames at a rate of about 30Hz, but this rate can fluctuate slightly during the experiments. To prevent synchronization issues between the camera and the control system, we let the arrival of camera images govern the time step length. Since we want to control the robot at about ten Hertz, we choose a new action whenever every third camera image is received. Consequently, not all torques are applied for the same duration, providing a source of transition noise.

To keep the amount of storage space and processing time limited, we down-sample the images to 15×20 pixels and reduce the image to a single channel (by subtracting the average brightness from the red channel value, since the color of the pendulum is red). The

7. The exact feedback signals used are given in Appendix C.



(a) (b)

Figure 6: The set-up of the real-robot swing-up experiment with redundant representation. (a) The robot set-up. One of the robot arms holds a pendulum which has a mass of about 1 kg and a length of about 80 cm. The Kinect camera in the foreground is used to provide feedback to the robot (proprioceptive joint information is not available to the robot). (b) An illustration of the set-up that illustrates the coordinate system used. As the image is shown from the camera's point of view, the coordinate system is inverted (clockwise θ). In the shaded area, an additional torque is applied to keep the robot from running into the joint limit at $\pi/2$.

image is blurred slightly with a low-pass filter to smooth out sensor noise, using a Gaussian kernel with a bandwidth of 6% of the image width. To provide the learner with a notion of velocity, the 15×20 pixel image and the 15×20 difference image to the previous time step were given to the robot as a concatenated 600-dimensional feature vector, as shown in Figure 7.

3.8.3 Employed Reward Function and Kernel.

In contrast to the earlier experiment, the real robot system is not periodic as it cannot turn the joint more than 2π. Consequently, we need a reward function that punishes the robot for getting too close to the joint limit. We use the reward function

$$r(\mathbf{s}, a) = \exp\left(-(1.5\theta)^2\right) - 2(\theta - 0.25)\,\mathbb{1}_{\{x \in \mathbb{R}: x > 0.25\}}(\theta) - 0.0005a^2,$$

where the first term rewards the robot for being close to the upright position and the second term penalizes the robot for being too close to the joint limit. All else being equal, we prefer solutions that do not needlessly change the torque, as jerky motion can damage the robots. Therefore, the third term penalizes large control actions.

As kernel on the state variables, we used a Gaussian kernel with three separate bandwidth parameters: one for all pixels of the current image, one for all pixels of the difference image, and one for the previously applied torque. The bandwidth parameters for all pixels of each image were tied together in this way to keep the hyper-parameter optimization manageable. For the learned transition model, the kernel on the action (torque increment) is again a squared exponential.

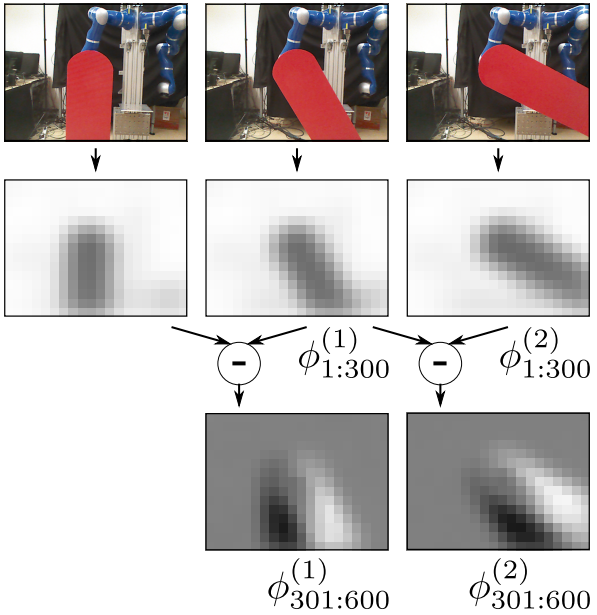


Figure 7: Sensor representation used by the robot. The Kinect camera image is downsampled into 15 by 20 pixels and converted to a single-channel image by subtracting the average intensity from the red channel values. From the resulting sequence of images, the current image and the difference between the current image and the previous image are concatenated with the previously applied torque, yielding a 601-dimensional feature vector.

In this experiment, the heuristic for setting the bandwidth for the value function approximation according to the TD-error did not always work. Therefore, these three bandwidth parameters were set by hand to 0.5, 1.4, and 4.0, respectively, as these values allowed good prediction performance while being able to represent the value function with enough accuracy. The bandwidth parameters and other kernel parameters for the learned transition model and the policy were set automatically by maximizing the same cross-validation objectives used in the previous experiments. The bound on the KL divergence was set to $\epsilon = 0.8$.

We used the Fourier feature approximation to the full kernel matrix, as the simulation experiments showed these features to yield good performance with an intermediate number of bases while also requiring relatively little computation time. Furthermore, computing the features and evaluating the linear Bayesian policy is straightforward to implement on a robotic system and can run in real-time as it has relatively low computational demands. We used 1000 random basis features, as the system is intrinsically higher dimensional compared to the simulated pendulum swing-up experiment (as we appended the previously applied torque to the state vector).

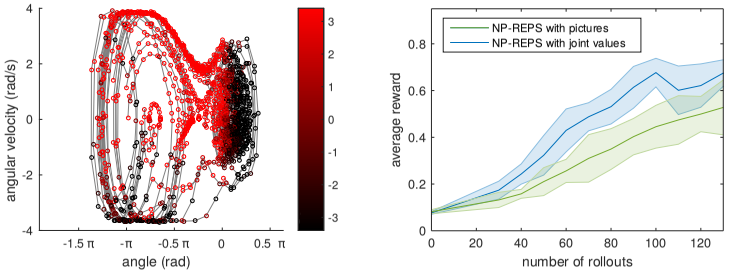
Sometimes, a feasible solution to the dual optimization problem could not be found. We considered this effect could be due to over-fitting to the sensor noise in the camera that made the system partially observable. We addressed this issue by projecting the 1000-dimensional feature vectors on all principal components with principal values at least 5% of the maximum principal value for purpose of optimization of the dual function (Eq. 11) only. This procedure yielded between 100 and 200 components on our data set and addressed the problem satisfactorily. The dimension reduction also sped up the optimization of the dual function. Models and policies were still learned in the original 1000-dimensional spaces, as these steps contain regularization terms that make them robust to such over-fitting.

3.8.4 Results for the Real-robot Swing-up from Vision Data.

The results of the robot swing-up task are shown in Figure 8.[8] Of the six trials we performed, five resulted in policies that successfully swing-up and balance the pendulum. An example of the phase-space is shown in Figure 8a. It is apparent that the pendulum oscillates near the target position which occasionally causes the pendulum to drop. The pendulum-camera system has some system delay which could cause such oscillations, and which also prevents a finer time discretization.

Figure 8b shows the average learning progress for the task. The system learns from an initial uninformed policy that hardly obtains high rewards. Our result shows that non-parametric methods are a promising approach to dealing with redundant state representations such as images, since the task is successfully learned. Nevertheless, knowing a good representation, such as joint values, still resulted in better learning performance. Generally, many different representations could be used as long as a kernel yields appropriate similarity values can be defined that. In the end, the algorithm only performs operations based on those similarity values. Some pixels will never change their value. Such pixels are not problematic, as stationary kernels such as the squared exponential are not influenced by them.

8. A video illustrating the experiment and the results is available at `http://youtu.be/y00HJu2qcO0`.



(a) Phase-space trajectories with torque. (b) Learning progress over six trials.

Figure 8: Results of the real-robot experiment from vision data. (a) Phase-space trajectories of one of the learned policies. Color indicates the applied torque (Nm). In most trajectories, the robot manages to swing up the pendulum and balance it around the upright position ($\theta = 0$). Near this position, the pendulum tends to oscillate as a consequence of time discretization and system delays. (b) Learning progress with input from either joint angles or pictures only. The REPS algorithm would converge to a locally optimum solution regardless of input modality, but seems to need more training time to reach such a local optimum using visual input. The graph shows the average reward over six independent trials. Error bars show twice the standard error.

4. Related Work

Over the years, many different reinforcement learning (RL) algorithms have been proposed, as reviewed by, among others, Bertsekas and Tsitsiklis (1996); Szepesvári (2010); Sutton and Barto (1998); Busoniu et al. (2010); Powell (2007); Bartlett (2003); Kaelbling et al. (1996); Wiering and Otterlo (2012); and Deisenroth et al. (2013). Many of the most well-known algorithms are value-function methods. However, there is no notion of the sampled data or sampling policy in the value function, making it impossible to limit the loss of information as the policy is updated (Peters et al., 2010). Furthermore, in continuous state-action spaces, such methods need to be approximated, which means that policy iteration does not necessarily improve the policy (Kakade and Langford, 2002; Bartlett, 2003).

Policy search methods are a complementary class of reinforcement learning approaches, which explicitly represent the sampling policy (Deisenroth et al., 2013). Such methods might additionally represent a value function, e.g. (Williams, 1992; Peters and Schaal, 2008; Sutton and Barto, 1998), in which case they are referred to as actor-critic algorithms. Policy search methods allow the learning agent to take the sampled data or the sampling policy into account. Policy search methods offer other advantages: policies might be easier to represent than value functions and convergent algorithms for policy search are known (Bagnell and Schneider, 2003a). Furthermore, for certain policy parameterizations, stability guarantees can be given (Deisenroth et al., 2013). If prior knowledge or task demonstrations are available, these can usually be integrated straightforwardly in policy search methods (Deisenroth et al., 2013; Peters and Schaal, 2006). For these reasons, policy search methods have in practice proven to work well on real (e.g. robotic) systems (Deisenroth et al., 2013).

In high-dimensional domains, both value-function and policy search methods have often relied on hand-crafted feature representations (Kaelbling et al., 1996; Kober et al., 2013; Bartlett, 2003). This hand-tuning can largely be avoided by learning a suitable representation (Jonschkowski and Brock, 2015; Böhmer et al., 2013). Such an approach, however, often requires a lot of training data and usually relies on non-convex optimization. Defining a representation can also be side-stepped by using non-parametric methods (Ormoneit and Sen, 2002; Rasmussen and Kuss, 2003), that implicitly use very rich representations that can adapt to the complexity of the data. Such methods, however, have the disadvantage that they usually rely on inverting matrices that grow with the data set. As a solution, efficient approximations can be used (Seeger et al., 2003; Snelson and Ghahramani, 2006; Rahimi and Recht, 2007).

We discuss these issues in more detail in the remainder of this section. First, we discuss various RL methods that provide stable policy updates. Subsequently, we give an overview of different studies addressing RL with high-dimensional states, followed by a discussion of non-parametric RL techniques. Finally, we discuss various methods for efficiently approximating non-parametric methods that allow such methods to be applied to large data sets.

4.1 Policy Updates with Limited Information Loss.

Perhaps the most straightforward way to limit the information loss of a policy is to stay close to a previous policy. Policy gradient methods (Williams, 1992; Sutton et al., 1999), for example, limit the policy update $\delta\boldsymbol{\theta}$ to a step in the gradient direction of fixed Euclidean

length $\delta\boldsymbol{\theta}^T\delta\boldsymbol{\theta} = \epsilon$. However, this metric is not invariant to re-parametrization of the policy. This problem can be addressed by instead using the Fisher information metric $\mathbf{F}$ in the constraint $\delta\boldsymbol{\theta}^T\mathbf{F}(\boldsymbol{\theta})\delta\boldsymbol{\theta} = \varepsilon$, as suggested by Kakade (2002). This constraint can be interpreted as the second-order Taylor expansion of the loss of information (relative entropy) between the path distributions of the original and the updated policy (Bagnell and Schneider, 2003a).

There are two main approaches to specifying such an information constraint. As suggested by Bagnell and Schneider (2003a), the loss of information between successive path distributions might be bounded. Such a formulation is equivalent to bounding the expected loss of information between successive policies, since the transition dynamics are the same under both path distributions. In our previous work (Peters et al., 2010), we instead proposed to bound the information loss between successive state-action distributions.

The first formulation has led to various algorithms that provide stable policy updates. For example, the dynamic policy programming algorithm (Azar et al., 2011) uses the relative entropy between successive policies as an additional cost term in the value function. Similar update equations have been derived from two points of view. Firstly, from the point of view of maximizing the cumulative reward under constraints on the communication bandwidth or data processing capacity (Tishby and Polani, 2011), or, equivalently, minimizing the policy complexity under a constraint on the policy's value (Still and Precup, 2012). Another derivation minimizes the relative entropy from the trajectory distribution conditioned on obtaining maximal rewards to the proposed policy (Rawlik et al., 2013b). In these approaches, the trade-off between greedy exploitation and maintaining stable updates—analogous to the (inverse) temperature of a Boltzmann distribution—is a free parameter. To obtain convergence, an update schedule that decays the temperature over time has to be designed. In contrast to adding the relative entropy as a cost-term, Levine and Abbeel (2014) employ a bound on the relative entropy between successive policies.

Another line of work, that is related to bounding the divergence between successive policies, has focused on guaranteeing improvement of the policy by limiting the policy update. In conservative policy iteration (Kakade and Langford, 2002) and safe policy iteration (Pirotta et al., 2013), the updated policy is a mixture of the old policy and a greedy policy that maximized a lower bound on the policy improvement. Trust region policy optimization (Schulman et al., 2015) similarly guarantees policy improvement, using the relative entropy between successive policies in the lower bound.

Kober et al. (2013) observed that "in practice, reinforcement learning algorithms tend to work best for real systems when they are constrained to make modest changes to the distribution over states while learning". However, depending on the system dynamics, a small change in the policy might cause a large change in the distribution of states visited by the policy. Thus, it may be advantageous to limit the relative entropy between successive state-action distributions rather than between successive policies or trajectory distributions. Empirically, a bound on the state-action distribution has been shown to outperform a bound on the policy for relatively large step-sizes (Lioutikov et al., 2014). Moreover, bounding the information loss between subsequent state-action distributions has been shown to have optimal regret in an adversarial Markov decision process (MDP) settings for discrete finite horizon problems (Zimin and Neu, 2013).

In our previous work (Peters et al., 2010), the Relative Entropy Policy Search (REPS) algorithm was derived based on such a bound on the relative entropy between successive state-action distributions. Policies learned using REPS re-weight the previous state-action distribution using a soft-max of the advantage function. This soft-max policy is reminiscent of the common ad-hoc exploration-exploitation trade-off using Boltzmann exploration (Kaelbling et al., 1996; Sutton, 1990; Lin, 1993), expectation-maximization based updates (Peters and Schaal, 2007; Kober et al., 2011), and the previously discussed algorithms by Azar et al. (2011) and Rawlik et al. (2013b). The advantage of REPS is, that the 'temperature' parameter that governs exploration is set directly by the algorithm. As a result, the algorithm is invariant to re-scaling of the reward function, and the temperature automatically decreases as the policy converges.

Alternative techniques search an optimal policy within the space of policies of similar exponential form (Lever and Stafford, 2015; Bagnell and Schneider, 2003b). In REPS, the exponential form does not result from an imposed search-space of the policy, but is a direct consequence of solving the bounded optimization problem. Under specific assumptions on the environment and the reward functions, exponential transformations of the value functions are also used in the definition of the optimal policy in the non-parametric method by Rawlik et al. (2013a). The embedding of the exponentiated value function, however, suffers from numerical problems where the value function is low.

Earlier approaches based on the REPS formulation (Lioutikov et al., 2014; Kupcsik et al., 2013; Peters et al., 2010; Daniel et al., 2016) have shown to be successful on a variety of problems. In all these approaches, a function-valued Lagrangian multiplier V emerges from the resulting optimization problem, which can be seen as a value function (Peters et al., 2010). However, these approaches assume the Lagrangian multiplier V is linear in manually-defined features, making it hard to apply the algorithm to domains with high-dimensional non-linear sensor representations. We relaxed this assumption, by requiring V to be a member of a non-linear RKHS, allowing implicit infinite feature representations. In contrast to earlier approaches based on the REPS optimization problem, our method can naturally handle non-parametric policies, such as Gaussian process policies.

REPS requires the estimation of the Bellman error. This estimation can, for example, be performed using a learned transition model. Thus far, work on learned transition models for REPS has been limited. The transition dynamics have been approximated by deterministic single-sample outcomes (Daniel et al., 2016; Peters et al., 2010) which only works well for deterministic environments, or by time-dependent linear models (Lioutikov et al., 2014). Gaussian process models have been used in the bandit setting (Kupcsik et al., 2013) to learn a simulator that predicts the outcome of new roll-outs. Instead, similar to previous value function methods (Grünewälder et al., 2012b; Boots et al., 2013; Nishiyama et al., 2012), we will employ empirical conditional RKHS embeddings to obtain non-linear models for step-based reinforcement learning.

4.2 Reinforcement Learning for High-dimensional State Representations.

Recently, several researchers have started to address the problem of reinforcement learning with high-dimensional state representations such as camera images. In many of these works, learning consists of two fully separate steps: learning a representation and, subsequently,

learning a policy or value function. One possible approach is to view the problem as reducing the dimensionality of the high-dimensional states while losing as little information as possible. For such a purpose, deep neural networks such as deep auto-encoders have become popular. For example, Lange et al. (2012) and Mattner et al. (2012) used such networks to learn state representations that were subsequently exploited to learn visual tasks using fitted Q-iteration or non-parametric approximate dynamic programming, respectively. Finn et al. (2016) used spatial auto-encoders, that exploit the spatial coherence of images, to learn visual pushing tasks using guided policy search.

The discussed methods do not explicitly make use of the structure of reinforcement learning problems. In domains where salient sensory distractors occur, taking such structure into account can help avoid representing those distractors. Jonschkowski and Brock (2015) proposed a method that takes observed transitions and rewards into account using a set of robotic priors, to learn representations for, among others, a visual navigation task. Another way to take the dynamics into account is to use slow-feature analysis, which encodes diffusion distances based on the transition kernel (Böhmer et al., 2013). The learned representations were used in an LSPI approach to learn visual navigation. A visual navigation task was also addressed by Boots et al. (2011), who select features based on their ability to predict characteristics of possible future events. Forcing the dynamics to be linear in the latent space is another way of enforcing a task-relevant representation. Watter et al. (2015) showed this principle to be successful at learning representations for model-predictive control of several dynamical systems in an off-policy setting.

The goal of finding a state representation that takes the task structure into account can also be reached by directly learning neural network policies that map from high-dimensional sensory information to control actions (Levine et al., 2016). In this approach, the hidden neurons act as feature representation. As the network is trained to reproduce correct actions, those neurons are optimized to provide a representation that helps the learning agent succeed at its task. This method requires having a low-dimensional task representation available at train time. It guides the neural network training using locally optimized trajectories. Mnih et al. (2015) instead directly used a neural network as an approximate Q function, as such an approach is able to scale to large networks and data sets. Their method obtained superb performance on playing Atari games, but is limited to problems with a discrete set of actions. Other methods have used neural networks that represent policies and have achieved impressive results on simulated dynamical systems, including bipedal locomotion and a driving simulator (Schulman et al., 2015; Mnih et al., 2016; Lillicrap et al., 2016). These methods, however, tend to require data sets with a million transitions or more to achieve these results. Such large data sets are impractical to obtain for real physical systems, e.g., robots. Furthermore, all neural-network based learning methods rely on non-convex optimization of all the network weights.

4.3 Non-parametric Reinforcement Learning Methods.

Non-parametric kernel methods use an implicit representation of the data, and therefore avoid the explicit choice of a feature representation. Many different approaches in reinforcement learning have been re-formulated to use non-parametric methods. For example, non-parametric value iteration methods have been proposed for (partially observable) MDPs

(Grünewälder et al., 2012b; Nishiyama et al., 2012), and non-parametric approximate dynamic programming method have been proposed by Ormoneit and Sen (2002), Taylor and Parr (2009), Deisenroth et al. (2009), Kroemer and Peters (2011) and Xu et al. (2014). Engel et al. (2003) proposed a Gaussian process (GP) temporal difference algorithm, and furthermore, there are examples of non-parametric policy iteration schemes such as kernelized least-squares policy evaluation (Jung and Polani, 2007) and least-squares policy iteration (Xu et al., 2007; Rasmussen and Kuss, 2003). The approximate linear programming algorithm, proposed by Pazis and Parr (2011), does not use kernels. Instead, this model-free method assumes the value function is Lipschitz, and assumes deterministic dynamics. Such value function methods use greedy maximization with respect to approximated value functions. Consequentially, these methods use deterministic actions. If exploration of the state space is required, heuristics such as an ϵ-greedy or a soft-max policy can be used. Furthermore, Grünewälder et al. (2012b) and Nishiyama et al. (2012) consider only discrete action sets and assume the state-action space can be sampled uniformly. For robotic systems, this can generally only be done in simulation.

Policy-search methods can be applied to address these shortcomings, by iteratively improving a policy. Kober et al. (2011) introduced cost-regularized kernel regression, which finds non-parametric policies for contextual bandits. In contrast, other approaches have focused on step-based decision making. For example, Bagnell and Schneider (2003b) developed a policy gradient method embedding a desirability function that defines a policy in a RKHS. However, their approach is restricted to discrete actions, and as a model-free method, cannot exploit learnable system dynamics. More recently, Lever and Stafford (2015) introduced another policy gradient approach, which searches for the mean function of a Gaussian process policy within a RKHS. Although this method cannot represent non-Gaussian policies, unlike the method of Bagnell and Schneider (2003b), it can represent policies that are close to deterministic more easily. A disadvantage of this method is that a schedule to decay the co-variance of the Gaussian towards zero has to be manually defined. Vien et al. (2016) introduced a non-parametric variant of the natural policy gradient, together with a natural actor-critic algorithm and expectation-maximization based updates.

Alternative non-parametric methods were proposed by Rawlik et al. (2013a) and Deisenroth and Rasmussen (2011). The method by Rawlik et al. (2013a) considers continuous-time systems with continuous actions. This method assumes the environments injects observable control noise and that the system is control-affine. Deisenroth and Rasmussen (2011) describe a model-based iterative method. They explicitly marginalize the uncertain non-parametric model to avoid over-greedy optimization. However, their method requires the reward function to be known and to be of squared exponential form. Additionally, it selects actions greedily and so it does not address the exploration problem.

4.4 Efficient Approximation for Non-parametric RL Methods.

Kernel-based non-parametric methods usually require inverting Gram matrices, which are of dimension $n \times n$, where n is the number of data points. This step is a bottleneck for scaling up these methods to large data sets (1000 - 10000 samples being the upper limit where most of these methods start to be prohibitively slow). Therefore, non-parametric approaches benefit from efficient approximations for large data sets.

Multiple non-parametric RL algorithms handle large data sets by selecting a subset of data points, and then finding coefficients for each of the kernels using a quadratic programming problem (Engel et al., 2003; Xu et al., 2014; Jung and Polani, 2007; Xu et al., 2007; Lever and Stafford, 2015). In most cases, this sparsification approach results in an approximation of the kernel function by a non-stationary kernel function parametrized by a subset of active data-points, as pointed out by Jung and Polani (2007) and Xu et al. (2007). If all data-points were chosen to be active, this approximation would be exact.

A sparsification approach for the method of Ormoneit and Sen (2002) has been proposed by Barreto et al. (2011). The proposed stochastic factorization approach, however, works only on stochastic kernel matrices, such as used in Nadaraya-Watson kernel regression. Another approach for applying cost-regularized kernel regression to large data sets is to apply a learning algorithm to separate subsets of the data and combine the results (Macedo et al., 2014). However, as the authors state, this approximation is not mathematically equivalent to the original problem, and if, as suggested, data from multiple iterations is combined, the on-policy assumption of the algorithm would be violated.

Yet another method takes an approach complimentary to sparsification approaches. Whereas sparse Gaussian process approaches essentially find coefficients for the true kernel matrix at a subset of data points, Rahimi and Recht (2007) propose an approach that evaluates an approximation to the kernel function at all training data points. To the best of our knowledge, this type of approximations has, so far, not been explored in the context of reinforcement learning. This approach approximates the kernel function as an inner product of random Fourier features. As such, it is reminiscent of the work by Fard et al. (2013) and Ghavamzadeh et al. (2010), who use random projections for parametric value function methods. However, these studies aim at reducing the dimensionality of sparse or redundant features using random projections, the method of Rahimi and Recht (2007) is used to project low-dimensional vectors into high-dimensional feature spaces so that the function approximation problems become linear. Konidaris et al. (2011) employ Fourier basis features for value function approximation. The parameters of the basis features were predetermined whereas in our approach the parameters are drawn from a distribution based on the kernel bandwidth, which can be optimized using standard techniques.

5. Conclusion and Future Work

In this paper, we have developed a policy search method with smooth, robust updates to solve continuous MDPs. Our method uses learned non-parametric models and allows the use of non-parametric policies, avoiding hand-crafted features. By taking the sampling distribution into account during policy updates, stable learning progress was obtained even with relatively small batches of data. By embedding the conditional transition distribution, expectations over functions of the next state can be computed without density estimation. The resulting predictions are robust even with state representations with hundreds of dimensions.

We show that the resulting algorithm is able to outperform other non-parametric algorithms on a reaching task and a pendulum swing-up task with control noise in on-policy settings. We also show good performance of the proposed algorithm relative to neural-

network based methods on a variant of the puddle-world task with a redundant input representation.

A limiting factor in applying the method to larger problems is the computational cost of inverting the Gram matrix. To address this issue, we evaluated different approaches to approximate this matrix that allow the inverse to be calculated efficiently. We found the approximation of the kernel function using random Fourier features to have desirable properties: they are computationally fast, easy to implement, and yielded good performance for moderate numbers of basis features. Sparsification yielded better performance on a smaller set of basis functions.

We evaluated the applicability of the algorithm to a real-robot under-powered swing-up task, with 600-dimensional visual representations of the pendulum's state. Here, we found that our method could successfully learn policies that swing up and balance the pendulum, from high-dimensional data.

Many tasks concerning sensory data are similar to the real-robot under-powered swing-up task, in that they have a high extrinsic dimensionality, but are intrinsically low-dimensional. Kernel-based algorithms perform all operations on kernel values, so they are invariant to the extrinsic dimensionality. Our kernel-based RL algorithm can, thus, be applied to such tasks without a separate dimension reduction. However, underlying distractor dimensions would negatively impact kernel-based algorithms unless the kernel parameters would be set (by design or using learning techniques) to ignore those effects.

The proposed algorithm is designed with data scarcity in mind, and shows good empirical performance in such cases. However, even with the proposed approximation of the Gram matrix, the run-time of the algorithm can still be a limiting factor. As the complexity of learning problems grows, we need more data and more approximating features, increasing the computational requirements. Thus, the proposed algorithm is likely to be a good choice when obtaining samples is costly or restricted, but might not be the best choice when samples are cheap relative to computation time.

In future work, we want to investigate synergies between the optimization problem and the generalizing policy. We will also address hyper-parameter optimization in systems with multi-dimensional controls, as our current objective does not always yield suitable hyper-parameters. We plan to apply our algorithm on real-world robotic tasks, including tasks with high-dimensional visual or tactile sensors, and investigate how to exploit structural knowledge about dynamical systems in such tasks. To avoid the need to enhance visual salience, learning task-relevant kernels will be essential.

Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007–2013) under grant agreements #610967 (TACMAN) and #270327 (CompLACS). Additionally, this project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement #645582 (RoMaNS). Part of this research has been made possible by the provision of compute time on the Lichtenberg cluster of the TU Darmstadt.

Appendix A. The Dual and its Derivatives

To obtain the dual function, we re-insert the state-action probabilities $p_\pi = \pi(\mathbf{a}|\mathbf{s})\mu_\pi(\mathbf{s})$ in the Lagrangian to obtain the dual

$$\begin{aligned}
g(\eta, V, \lambda) =& \lambda + \eta\epsilon + \mathbb{E}_{p_\pi(\mathbf{s},\mathbf{a})}\left[\delta(\mathbf{s},\mathbf{a},V) - \lambda - \eta \log \frac{p_\pi(\mathbf{s},\mathbf{a})}{q(\mathbf{s},\mathbf{a})}\right], \\
=& \lambda + \eta\epsilon + \mathbb{E}_{p_\pi(\mathbf{s},\mathbf{a})}\left[-\lambda + \lambda\right] + \mathbb{E}_{p_\pi(\mathbf{s},\mathbf{a})}\left[\delta(\mathbf{s},\mathbf{a},V) - \delta(\mathbf{s},\mathbf{a},V) + \eta\right], \\
=& \lambda + \eta\epsilon + \mathbb{E}_{p_\pi(\mathbf{s},\mathbf{a})}\eta \; \mathrm{d}\mathbf{a}\mathrm{d}\mathbf{s} = \lambda + \eta\epsilon + \eta = \eta\epsilon + \eta\log(Z), \\
=& \eta\epsilon + \eta \log\left(\mathbb{E}_{q(\mathbf{s},\mathbf{a})} \exp\left(\delta(\mathbf{s},\mathbf{a},V)/\eta\right)\right),
\end{aligned}$$

where we used that $\exp\left(-\lambda/\eta - 1\right) = Z^{-1}$, so $\lambda + \eta = \eta\log(Z)$. The expected value over q can straightforwardly be approximated by taking the average of samples $1, \ldots, n$ taken from q. Note that λ and q do not appear in the final expression.

$$g(\eta, V) = \eta\epsilon + \eta \log\left(\frac{1}{n}\sum_{i=1}^{n} \exp\left(\delta(\mathbf{s}_i,\mathbf{a}_i,V)/\eta\right)\right).$$

When employing the kernel embedding, the Bellman error is written as

$$\delta(\mathbf{s}_i,\mathbf{a}_i,\boldsymbol{\alpha}) = \mathcal{R}_{\mathbf{s}_i}^{\mathbf{a}_i} + \boldsymbol{\alpha}^T(\mathbf{K}\beta(\mathbf{s}_i,\mathbf{a}_i) - \mathbf{k}_\mathrm{s}(\mathbf{s}_i)).$$

We define

$$w_i = \frac{\exp\left(\delta(\mathbf{s}_i,\mathbf{a}_i,\boldsymbol{\alpha})/\eta\right)}{\sum_{i=j}^{n} \exp\left(\delta(\mathbf{s}_j,\mathbf{a}_j,\boldsymbol{\alpha})/\eta\right)}$$

to keep equations brief and readable. The partial derivatives can be written as:

$$\begin{aligned}
\frac{\partial g(\eta,\boldsymbol{\alpha})}{\partial \eta} =& -\frac{1}{\eta}\sum_{i=1}^{n} w_i\delta(\mathbf{s}_i,\mathbf{a}_i,\boldsymbol{\alpha}) + \epsilon + \log\left(\frac{1}{n}\sum_{i=1}^{n}\exp\left(\delta(\mathbf{s}_i,\mathbf{a}_i,\boldsymbol{\alpha})/\eta\right)\right), \\
\frac{\partial g(\eta,\boldsymbol{\alpha})}{\partial \boldsymbol{\alpha}} =& \sum_{i=1}^{n} w_i\left(\mathbf{K}\beta(\mathbf{s}_i,\mathbf{a}_i) - \mathbf{k}_\mathrm{s}(\mathbf{s}_i)\right),
\end{aligned}$$

and furthermore, we obtain the Hessian:

$$\begin{aligned}
\mathbf{H}(g(\eta,\boldsymbol{\theta}) =& \frac{1}{\eta}\sum_{i=1}^{n}\left(w_i\begin{bmatrix} -\delta(\mathbf{s}_i,\mathbf{a}_i,\boldsymbol{\alpha}) \\ \mathbf{K}\beta(\mathbf{s}_i,\mathbf{a}_i) - \mathbf{k}_\mathrm{s}(\mathbf{s}_i)\end{bmatrix}\begin{bmatrix} -\delta(\mathbf{s}_i,\mathbf{a}_i,\boldsymbol{\alpha}) \\ \mathbf{K}\beta(\mathbf{s}_i,\mathbf{a}_i) - \mathbf{k}_\mathrm{s}(\mathbf{s}_i)\end{bmatrix}^T\right) \\
& - \frac{1}{\eta}\sum_{i=1}^{n}\left(w_i\begin{bmatrix} -\delta(\mathbf{s}_i,\mathbf{a}_i,\boldsymbol{\alpha}) \\ \mathbf{K}\beta(\mathbf{s}_i,\mathbf{a}_i) - \mathbf{k}_\mathrm{s}(\mathbf{s}_i)\end{bmatrix}\right)\sum_{i=1}^{n}\left(w_i\begin{bmatrix} -\delta(\mathbf{s}_i,\mathbf{a}_i,\boldsymbol{\alpha}) \\ \mathbf{K}\beta(\mathbf{s}_i,\mathbf{a}_i) - \mathbf{k}_\mathrm{s}(\mathbf{s}_i)\end{bmatrix}\right)^T.
\end{aligned}$$

The Hessian of a function is positive semi-definite on a set if and only if the function is convex on that set. Since η is the Lagrange multiplier for an inequality constraint, we have $\eta \geq 0$.To show that $\mathbf{H}(g(\eta,\boldsymbol{\theta}))$ is positive semi-definite on $(\eta,\theta) \in [0,+\infty) \times \mathbb{R}^n$ we introduce the following abbreviation:

$$\mathbf{u}_i = \begin{bmatrix} -\delta(\mathbf{s}_i,\mathbf{a}_i,\boldsymbol{\alpha}) \\ \mathbf{K}\beta(\mathbf{s}_i,\mathbf{a}_i) - \mathbf{k}_\mathrm{s}(\mathbf{s}_i)\end{bmatrix}^T.$$

Now, we can re-write the Hessian

$$\begin{aligned}
\mathbf{H}(g(\eta,\boldsymbol{\alpha})) =& \frac{1}{\eta}\sum_{i=1}^{n}\left(w_i\mathbf{u}_i\mathbf{u}_i^T\right) - \frac{1}{\eta}\sum_{i=1}^{n}\left(w_i\mathbf{u}_i\right)\sum_{i=1}^{n}\left(w_i\mathbf{u}_i\right)^T \\
=& \frac{1}{\eta}\sum_{i=1}^{n}\left(w_i\mathbf{u}_i\mathbf{u}_i^T\right) - \frac{2}{\eta}\sum_{i=1}^{n}\left(w_i\mathbf{u}_i\right)\sum_{j=1}^{n}\left(w_j\mathbf{u}_j\right)^T \\
&+ \frac{1}{\eta}\sum_{i=1}^{n}\left(w_i\right)\sum_{j=1}^{n}\left(w_j\mathbf{u}_j\right)\sum_{j=1}^{n}\left(w_j\mathbf{u}_j\right)^T \\
=& \frac{1}{\eta}\sum_{i=1}^{n} w_i\left(\mathbf{u}_i\mathbf{u}_i^T - 2\mathbf{u}_i\sum_{j=1}^{n}\left(w_j\mathbf{u}_j\right)^T + \sum_{j=1}^{n}\left(w_j\mathbf{u}_j\right)\sum_{j=1}^{n}\left(w_j\mathbf{u}_j\right)^T\right) \\
=& \frac{1}{\eta}\sum_{i=1}^{n}\left(w_i\left(\mathbf{u}_i - \sum_{j=1}^{n} w_j\mathbf{u}_j\right)\left(\mathbf{u}_i - \sum_{j=1}^{n} w_j\mathbf{u}_j\right)^T\right).
\end{aligned}$$

Note that we have used the property that $\sum_{i=1}^{n} w_i = 1$. $\mathbf{H}$ is positive semi-definite if $\mathbf{a}^T\mathbf{H}\mathbf{a} \geq 0$ for any $\mathbf{a}$.

$$\begin{aligned}
\mathbf{a}^T\mathbf{H}\mathbf{a} =& \mathbf{a}^T\left(\frac{1}{\eta}\sum_{i=1}^{n}\left(w_i\left(\mathbf{u}_i - \sum_{j=1}^{n} w_j\mathbf{u}_j\right)\left(\mathbf{u}_i - \sum_{j=1}^{n} w_j\mathbf{u}_j\right)^T\right)\right)\mathbf{a} \\
=& \frac{1}{\eta}\sum_{i=1}^{n}\left(w_i\mathbf{a}^T\left(\mathbf{u}_i - \sum_{j=1}^{n} w_j\mathbf{u}_j\right)\left(\mathbf{u}_i - \sum_{j=1}^{n} w_j\mathbf{u}_j\right)^T\mathbf{a}\right) \\
=& \frac{1}{\eta}\sum_{i=1}^{n}\left(w_i\left(\mathbf{a}^T\mathbf{u}_i - \sum_{j=1}^{n} w_j\mathbf{a}^T\mathbf{u}_j\right)^2\right).
\end{aligned}$$

The factors $\left(\mathbf{a}^T\mathbf{u}_i - \sum_{j=1}^{n} w_j\mathbf{a}^T\mathbf{u}_j\right)^2$ are squares of real numbers and thus non-negative. Since the w_i are the result of a soft-max operation and thus non-negative, this holds for the weighted sum, too. Thus, for any $\eta \geq 0$, $\mathbf{a}^T\mathbf{H}\mathbf{a} \geq 0$ and g is convex.

Appendix B. Optimization with Respect to V

We want to show that at least one of the functions V minimizing the dual functional g can be represented using a weighted sum of kernel functions centered at the sample states, i.e., that

$$V^* = \sum_{\tilde{\mathbf{s}}\in\tilde{\mathcal{S}}} \alpha_{\tilde{\mathbf{s}}} k_{\mathrm{s}}(\tilde{\mathbf{s}},\cdot), \tag{18}$$

where $\tilde{\mathcal{S}}$ is the set of states samples during the roll-outs. We follow some steps in the proof of Schölkopf et al. (2001). They consider arbitrary objective functions c mapping to $\mathbb{R}\cup\{\infty\}$ of the form

$$c((\mathbf{s}_1, y_1, V(\mathbf{s}_1)), \ldots, (\mathbf{s}_m, y_m, V(\mathbf{s}_m))), \tag{19}$$

which can represent an error function between the value of a function $V(\mathbf{s})$ at the samples $\mathbf{s}_i$ and the corresponding desired outputs y_i. In our case, we do not have desired output values y_i for our objective function. This is inconsequential as c can be arbitrary, and so can be independent of all y values.

Any function V can be written as $V = \sum_{\tilde{\mathbf{s}}\in\tilde{\mathcal{S}}} \alpha_{\tilde{\mathbf{s}}} k_{\mathrm{s}}(\tilde{\mathbf{s}},\cdot) + v(\mathbf{s})$, where $v(\mathbf{s})$ is an additional bias term. If V is constrained to be in the Hilbert space defined by k, Schölkopf et al. [2001] show that c is independent of the bias term $v(\mathbf{s})$. This means that for any optimal V' that is not of the proposed form, there is a V^* of the proposed form that has the same objective value which is obtained by subtracting $v(\mathbf{s})$ from V'.

As the dual function g satisfies the conditions to cost function c, for us this means that there is at least one V^* optimizing g of the proposed form. Note that it is inconsequential that the dual g also depends on Lagrangian parameter η. For any optimum $(\eta^*, V^{*\prime})$, if $V^{*\prime}$ is not of the proposed form, projecting $V^{*\prime}$ on the proposed basis yields another function V^* that satisfies $g(\eta^*, V^{*\prime}) = g(\eta^*, V^*)$, so (η^*, V^*) must be an optimum as well.

Therefore, there is always at least one minimum of any such function c of the proposed form.

Appendix C. Feedback signals to avoid joint angle and velocity limits

In our real robot set-up, the control action selected by the algorithm is the increment in the torque to be applied. However, to avoid running into joint angle and velocity limits, the resulting torque is modified when these limits are approached. Therefore, the torque applied at time step t

$$\tau^{(t+1)} = \max\left(\min\left(\tau_t + u + \tau_{\mathrm{s}}^{(t)} + \tau_{\mathrm{d}}^{(t)}, \tau_{\max}^{(t)}\right), \tau_{\min}^{(t)}\right),$$

where $\tau_{\mathrm{s}}^{(t)}$ and $\tau_{\mathrm{s}}^{(d)}$ terms that act as spring and damper when the robot gets close to the joint limit, and $\tau_{\max}^{(t)}$ and $\tau_{\min}^{(t)}$ are the maximum and minimum torque, respectively, that can be applied without breaking the velocity limit. The definition of θ, as well as the joint limit and the area where the feedback terms are applied, are illustrated in Figure 6b.

The feedback terms are defined as follows. The spring-like term

$$\tau_{\mathrm{s}}^{(t)} = \begin{cases} 0 & \text{if } -5/4\pi < \theta < 1/4\pi, \\ 15(-5/4\pi - \theta) & \text{if } \theta < -5/4\pi, \\ 15(1/4\pi - \theta) & \text{if } 1/4\pi < \theta, \end{cases}$$

is applied whenever the joint gets close to the joint limit at $1/2\pi = -3/2\pi$. The damper-like term

$$\tau_{\mathrm{d}}^{(t)} = \begin{cases} 0.3(-5/4\pi - \theta)\dot{\theta} & \text{if } \theta < -5/4\pi \text{ and } \dot{\theta} > 0, \\ 0.3(1/4\pi - \theta)\dot{\theta} & \text{if } 1/4\pi < \theta \text{ and } \dot{\theta} < 0, \\ 0 & \text{otherwise}, \end{cases}$$

is applied in the same region, as the feedback has to be higher when the pendulum has a high velocity. To prevent the velocity from exceeding the velocity limit, additionally, the minimum and maximum torques

$$\tau_{\max}^{(t)} = \min\left(\tau_{\max}^{(t-1)} + 0.05, 20(3.85 - \dot{\theta}) + 4.5\cos(\theta)\right),$$

$$\tau_{\min}^{(t)} = \max\left(\tau_{\min}^{(t-1)} - 0.05, 20(-3.7 - \dot{\theta}) + 4.5\cos(\theta)\right),$$

are applied. The cosine term is a rough compensation for the torque induced by gravity. The term linear in $\dot{\theta}$ is a linear damping term. Furthermore, the system has some delays which tended to induce oscillations close to the maximum torque. Therefore, the maximum (respectively minimum) can only be increased (decreased) by a small amount relative to the previous value.

References

M. G. Azar, V. Gómez, and B. Kappen. Dynamic policy programming with function approximation. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AIstats)*, pages 119–127, 2011.

J.A. Bagnell and J. Schneider. Covariant policy search. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2003a.

J.A. Bagnell and J. Schneider. Policy search in reproducing kernel Hilbert space. Technical Report RI-TR-03-45, CMU, 2003b.

A. S. Barreto, D. Precup, and J. Pineau. Reinforcement learning using kernel-based stochastic factorization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 720–728, 2011.

P. L. Bartlett. An introduction to reinforcement learning theory: Value function methods. In *Advanced Lectures on Machine Learning*, pages 184–202. Springer Berlin Heidelberg, 2003.

D. P. Bertsekas and J. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

W. Böhmer, S. Grünewälder, Y. Shen, M. Musial, and K. Obermayer. Construction of approximation spaces for reinforcement learning. *The Journal of Machine Learning Research*, 14(1):2067–2118, 2013.

B. Boots, S. Siddiqi, and G. Gordon. Closing the learning planning loop with predictive state representations. *International Journal of Robotics Research*, 30:954–956, 2011.

B. Boots, A. Gretton, and G. J. Gordon. Hilbert space embeddings of predictive state representations. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, 2013.

L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst. *Reinforcement Learning and Dynamic Programming using Function Approximators*, volume 39. CRC press, 2010.

C. Daniel, G. Neumann, O. Kroemer, and J. Peters. Hierarchical relative entropy policy search. *Journal of Machine Learning Research*, 2016.

P. Dayan and G.E. Hinton. Using expectation-maximization for reinforcement learning. *Neural Computation*, 9(2):271–278, 1997.

M. P. Deisenroth and C. E. Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 465–472, 2011.

M. P. Deisenroth, C. E. Rasmussen, and J. Peters. Gaussian process dynamic programming. *Neurocomputing*, 72(7):1508–1524, 2009.

M. P. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, pages 388–403, 2013.

Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2016.

Y. Engel, S. Mannor, and R. Meir. Bayes meets Bellman: The Gaussian process approach to temporal difference learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 20, pages 154–161, 2003.

M. M. Fard, Y. Grinberg, A.-M. Farahmand, J. Pineau, and D. Precup. Bellman error based feature generation using random projections on sparse spaces. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3030–3038, 2013.

C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel. Deep spatial autoencoders for visuomotor learning. In *Proceedings of the IEEE International Conference on Robotics and Automations (ICRA)*, 2016.

A. Gelman, J.B. Carlin, H.S. Stern, and D.B. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC, 2 edition, 2004.

M. Ghavamzadeh, A. Lazaric, O. Maillard, and R. Munos. LSTD with random projections. In *Advances in Neural Information Processing Systems (NIPS)*, pages 721–729, 2010.

S. Grünewälder, G. Lever, L. Baldassarre, S. Patterson, A. Gretton, and M. Pontil. Conditional mean embeddings as regressors. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1823–1830, 2012a.

S. Grünewälder, G. Lever, L. Baldassarre, M. Pontil, and A. Gretton. Modelling transition dynamics in MDPs with RKHS embeddings. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 535–542, 2012b.

J. M. Hernández-Lobato, M. W. Hoffman, and Z. Ghahramani. Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems (NIPS)*, pages 918–926, 2014.

T. Hofmann, B. Schölkopf, and A. J. Smola. Kernel methods in machine learning. *The Annals of Statistics*, pages 1171–1220, 2008.

R. Jonschkowski and O. Brock. Learning state representations with robotic priors. *Autonomous Robots*, 39(3):407–428, 2015.

T. Jung and D. Polani. Kernelizing LSPE(λ). In *Proceedings of the IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning*, pages 338–345, 2007.

L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

S. A. Kakade. Natural policy gradient. In *Advanced in Neural Information Processing Systems (NIPS)*, 2002.

S. A. Kakade and J. Langford. Approximately optimal approximate reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 2, pages 267–274, 2002.

J. Kober, E. Oztop, and J. Peters. Reinforcement learning to adjust robot movements to new situations. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2650–2655, 2011.

J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 11(32):1238–1274, 2013.

G. D. Konidaris, Osentoski S., and Thomas P. S. Value function approximation in reinforcement learning using the Fourier basis. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 380–385, 2011.

O. Kroemer and J. Peters. A non-parametric approach to dynamic programming. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1719–1727, 2011.

A. G. Kupcsik, M. P. Deisenroth, J. Peters, and G. Neumann. Data-efficient generalization of robot skills with contextual policy search. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2013.

M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *Journal of Machine Learning Research*, 4:1107–1149, December 2003.

S. Lange, M. Riedmiller, and A. Voigtländer. Autonomous reinforcement learning on raw visual input data in a real world application. In *International Joint Conference on Neural Networks*, 2012.

G. Lever and R. Stafford. Modelling policies in MDPs in reproducing kernel Hilbert space. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AIstats)*, pages 590–598, 2015.

S. Levine and P. Abbeel. Learning neural network policies with guided policy search under unknown dynamics. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1071–1079, 2014.

S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2016.

L.-J. Lin. *Reinforcement Learning for Robots using Neural Networks*. PhD thesis, Carnegie Mellon University, 1993.

R. Lioutikov, A. Paraschos, J. Peters, and G. Neumann. Generalizing movements with information theoretic stochastic optimal control. *Journal of Aerospace Information Systems*, (9):579–595, 2014.

Z. Lu, D. Guo, A. Bagheri Garakani, K. Liu, A. May, A. Bellet, L. Fan, M. Collins, B. Kingsbury, M. Picheny, and F. Sha. A comparison between deep neural nets and kernel acoustic models for speech recognition. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 2016.

J. Macedo, C. Santos, and L. Costa. Using cost-regularized kernel regression with a high number of samples. In *Proceedings of the IEEE International Conference on Autonomous Robot Systems and Competitions*, pages 199–204, 2014.

S. Mannor, D. Simester, P. Sun, and J. N. Tsitsiklis. Biases and variance in value function estimates. *Management Science*, 53(2):308–322, February 2007.

J. Mattner, S. Lange, and M. Riedmiller. Learn to swing up and balance a real pole based on raw visual input data. In *International Conference on Neural Information Processing*, pages 126–133, 2012.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1928–1937, 2016.

Y. Nishiyama, A. Boularias, A. Gretton, and K. Fukumizu. Hilbert space embeddings of POMDPs. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 644–653, 2012.

D. Ormoneit and Ś. Sen. Kernel-based reinforcement learning. *Machine Learning*, 49(2-3): 161–178, 2002.

R. Parr, L. Li, G. Taylor, C. Painter-Wakefield, and M. L. Littman. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 752–759, 2008.

J. Pazis and R. Parr. Non-parametric approximate linear programming for MDPs. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 459–464, 2011.

J. Peters and S. Schaal. Policy gradient methods for robotics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2219–2225, 2006.

J. Peters and S. Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 745–750, 2007.

J. Peters and S. Schaal. Natural actor critic. *Neurocomputing*, 71(7-9):1180–1190, 2008.

J. Peters, J. Kober, and D. Nguyen-Tuong. Policy learning – a unified perspective with applications in robotics. In *Proceedings of the European Workshop on Reinforcement Learning (EWRL)*, 2008.

J. Peters, K. Mülling, and Y. Altün. Relative entropy policy search. In *Proceedings of the National Conference on Artificial Intelligence (AAAI), Physically Grounded AI Track*, pages 1607–1612, 2010.

M. Pirotta, M. Restelli, A. Pecorino, and D. Calandriello. Safe policy iteration. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 307–315, 2013.

W. B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1177–1184, 2007.

C. E. Rasmussen and M. Kuss. Gaussian processes in reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS)*, volume 4, page 1, 2003.

K. Rawlik, M. Toussaint, and S. Vijayakumar. Path integral control by reproducing kernel Hilbert space embedding. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2013a.

K. Rawlik, M. Toussaint, and S. Vijayakumar. On stochastic optimal control and reinforcement learning by approximate inference. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3052–3056. AAAI Press, 2013b.

B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. J. Smola. Input space versus feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, 1999.

B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In David Helmbold and Bob Williamson, editors, *Computational Learning Theory*, volume 2111 of *Lecture Notes in Computer Science*, pages 416–426. Springer Berlin Heidelberg, 2001.

J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1889–1897, 2015.

M. Seeger, C.K.I. Williams, and N.D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In *Workshop on Artificial Intelligence and Statistics*, 2003.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1257–1264, 2006.

L. Song, J. Huang, A. Smola, and K. Fukumizu. Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 961–968, 2009.

L. Song, K. Fukumizu, and A. Gretton. Kernel embeddings of conditional distributions: A unified kernel framework for nonparametric inference in graphical models. *Signal Processing Magazine, IEEE*, 30(4):98–111, 2013.

S. Still and D. Precup. An information-theoretic approach to curiosity-driven reinforcement learning. *Theory in Biosciences*, 131(3):139–148, 2012.

R. S. Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 216–224, 1990.

R. S. Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1038–1044, 1996.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 1998.

R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1057–1063, 1999.

C. Szepesvári. Algorithms for reinforcement learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 4(1):1–103, 2010.

G. Taylor and R. Parr. Kernelized value function approximation for reinforcement learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1017–1024, 2009.

N. Tishby and D. Polani. Information theory of decisions and actions. In *Perception-Action Cycle*, pages 601–636. Springer, 2011.

H. van Hoof, T. Hermans, G. Neumann, and J. Peters. Learning robot in-hand manipulation with tactile features. In *Proceedings of the IEEE International Conference on Humanoid Robots (Humanoids)*, pages 121–127, 2015a.

H. van Hoof, J. Peters, and G. Neumann. Learning of non-parametric control policies with high-dimensional state features. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AIstats)*, pages 995–1003, 2015b.

N.A. Vien, P. Englert, and M. Toussaint. Policy search in reproducing kernel Hilbert space. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

M. Watter, J. T. Springenberg, J. Boedecker, and M. Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2728–2736, 2015.

M. Wiering and M. Otterlo. *Reinforcement learning: State-of-the-art*. Springer Berline Heidelberg, 2012.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

X. Xu, D. Hu, and X. Lu. Kernel-based least squares policy iteration for reinforcement learning. *IEEE Transactions on Neural Networks*, 18(4):973–992, 2007.

X. Xu, C. Lian, L. Zuo, and H. He. Kernel-based approximate dynamic programming for real-time online learning control: An experimental study. *IEEE Transactions on Control Systems Technology*, 22(1):146–156, 2014.

A. Zimin and G. Neu. Online learning in episodic Markovian decision processes by relative entropy policy search. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1583–1591, 2013.